SỞ GIÁO DỤC VÀ ĐÀO TẠO HÀ NỘI

GIÁO TRÌNH

Cơ sở dữ liệu quan hệ

DÙNG TRONG CÁC TRƯỜNG
TRUNG HỌC CHUYÊN NGHIỆP

NHÀ XUẤT BẢN HÀ NỘI

SỞ GIÁO DỤC VÀ ĐÀO TẠO HÀ NỘI

PHẠM ĐỨC NHIỆM

GIÁO TRÌNH
CƠ SỞ DỮ LIỆU QUAN HỆ

(Dùng trong các trường THCN)

NHÀ XUẤT BẢN HÀ NỘI - 2005

Lời giới thiệu

Nước ta đang bước vào thời kỳ công nghiệp hóa, hiện đại hóa nhằm đưa Việt Nam trở thành nước công nghiệp văn minh, hiện đại.

Trong sự nghiệp cách mạng to lớn đó, công tác đào tạo nhân lực luôn giữ vai trò quan trọng. Báo cáo Chính trị của Ban Chấp hành Trung ương Đảng Cộng sản Việt Nam tại Đại hội Đảng toàn quốc lần thứ IX đã chỉ rõ: "Phát triển giáo dục và đào tạo là một trong những động lực quan trọng thúc đẩy sự nghiệp *công nghiệp hóa, hiện đại hóa*, là điều kiện để phát triển nguồn lực con người - yếu tố cơ bản để phát triển xã hội, tăng trưởng kinh tế nhanh và bền vững".

Quán triệt chủ trương, Nghị quyết của Đảng và Nhà nước và nhận thức đúng đắn về tầm quan trọng của chương trình, giáo trình đối với việc nâng cao chất lượng đào tạo, theo đề nghị của Sở Giáo dục và Đào tạo Hà Nội, ngày 23/9/2003, Ủyban nhân dân thành phố Hà Nội đã ra Quyết định số 5620/QĐ-UB cho phép Sở Giáo dục và Đào tạo thực hiện đề án biên soạn chương trình, giáo trình trong các trường Trung học chuyên nghiệp (THCN) Hà Nội. Quyết định này thể hiện sự quan tâm sâu sắc của Thành ủy, UBND thành phố trong việc nâng cao chất lượng đào tạo và phát triển nguồn nhân lực Thủ đô.

Trên cơ sở chương trình khung của Bộ Giáo dục và Đào tạo ban hành và những kinh nghiệm rút ra từ thực tế đào tạo, Sở Giáo dục và Đào tạo đã chỉ đạo các trường THCN tổ chức biên soạn chương trình, giáo trình một cách khoa học, hệ

thống và cập nhật những kiến thức thực tiễn phù hợp với đối tượng học sinh THCN Hà Nội.

Bộ giáo trình này là tài liệu giảng dạy và học tập trong các trường THCN ở Hà Nội, đồng thời là tài liệu tham khảo hữu ích cho các trường có đào tạo các ngành kỹ thuật - nghiệp vụ và đông đảo bạn đọc quan tâm đến vấn đề hướng nghiệp, dạy nghề.

Việc tổ chức biên soạn bộ chương trình, giáo trình này là một trong nhiều hoạt động thiết thực của ngành giáo dục và đào tạo Thủ đô để kỷ niệm "50 năm giải phóng Thủ đô", "50 năm thành lập ngành" và hướng tới kỷ niệm "1000 năm Thăng Long - Hà Nội".

Sở Giáo dục và Đào tạo Hà Nội chân thành cảm ơn Thành ủy, UBND, các sở, ban, ngành của Thành phố, Vụ Giáo dục chuyên nghiệp Bộ Giáo dục và Đào tạo, các nhà khoa học, các chuyên gia đầu ngành, các giảng viên, các nhà quản lý, các nhà doanh nghiệp đã tạo điều kiện giúp đỡ, đóng góp ý kiến, tham gia Hội đồng phản biện, Hội đồng thẩm định và Hội đồng nghiệm thu các chương trình, giáo trình.

Đây là lần đầu tiên Sở Giáo dục và Đào tạo Hà Nội tổ chức biên soạn chương trình, giáo trình. Dù đã hết sức cố gắng nhưng chắc chắn không tránh khỏi thiếu sót, bất cập. Chúng tôi mong nhận được những ý kiến đóng góp của bạn đọc để từng bước hoàn thiện bộ giáo trình trong các lần tái bản sau.

GIÁM ĐỐC SỞ GIÁO DỤC VÀ ĐÀO TẠO

Lời nói đầu

Trong thực tế việc, ứng dụng công nghệ thông tin (CNTT) hiện nay để quản lý các thông tin của các đối tượng cần quan tâm là một đòi hỏi cấp thiết. Việc xây dựng các hệ thống quản lý các thông tin trên máy tính được sử dụng rộng rãi trên nhiều lĩnh vực như: kinh tế, xã hội, quốc phòng, an ninh, v.v. Một trong những vấn đề quan trọng trong quản lý thông tin là xây dựng cơ sở dữ liệu sao cho các phép toán xử lý trên chúng có hiệu quả cao nhất. Chính vậy, cơ sở dữ liệu (CSDL- Database) là một lĩnh vực phát triển mạnh của công nghệ thông tin. Cùng với sự phát triển của CNTT ở nước ta, việc sử dụng các kiến thức về cơ sở dữ liệu (CSDL) ngày càng trở nên cấp thiết. Ở Việt Nam, các tài liệu về CSDL bằng tiếng Việt còn rất ít ỏi. Hiện nay chúng ta chưa có một tài liệu chung dùng làm giáo trình chuẩn, đầy đủ, cơ bản để giảng dạy cho sinh viên, học sinh. Trong giáo trình này, tôi muốn trình bày một số kiến thức cơ bản nhất về mô hình CSDL quan hệ giúp cho học sinh có thể ứng dụng các kiến thức về CSDL vào thực tiễn, tiếp tục nghiên cứu sâu về lý thuyết CSDL cũng như các môn tin học khác, đồng thời làm tài liệu tham khảo cho các bạn đọc trong công việc nghiên cứu, giảng dạy và học tập.

Giáo trình được chia làm 4 chương:

Chương 1: Sơ lược về cơ sở dữ liệu

Nội dung chương 1 là một số khái niệm chung về cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu, sơ lược về một số mô hình và có trình bày một số kiến thức quan trọng về mô hình thực thể - liên hệ, làm nền tảng cho việc thiết kế các lược đồ cơ sở dữ liệu khái niệm trong các mô hình khác.

Chương 2: Mô hình cơ sở dữ liệu quan hệ

Chương này đi sâu vào trình bày các khái niệm quan trọng cũng như một

số phép toán cơ bản của mô hình CSDL quan hệ. Chương 2 còn đề cập đến cách chuyển sơ đồ thực thể - liên hệ về các lược đồ CSDL quan hệ cùng với các kiến thức nhằm giúp đơn giản hoá và tinh chỉnh các lược đồ này.

Chương 3: Ngôn ngữ con dữ liệu SQL

Chương 3 trình bày SQL như một ngôn ngữ đầy đủ, vừa là ngôn ngữ định nghĩa dữ liệu (DDL), vừa là ngôn ngữ thao tác dữ liệu (DML). Các ví dụ trong chương này là các ví dụ đã chạy thật bằng các câu lệnh SQL chuẩn ISO trên môi trường VISUAL FOXPRO.

Chương 4: Lý thuyết thiết kế cơ sở dữ liệu quan hệ

Trọng tâm của chương này là các kiến thức về phụ thuộc hàm, áp dụng các kiến thức này vào việc tìm khoá và chuẩn hoá các lược đồ CSDL quan hệ.

Nhân đây tôi xin chân thành cảm ơn các thầy và các cán bộ nghiên cứu khoa học PGS.TS. Nguyễn Thiện Luận; TS. Đỗ Năng Toàn; ThS. Lê Văn Năng; ThS. Nguyễn Duy Phương đã đóng góp những ý kiến quý báu cho giáo trình.

Mặc dù đã cố gắng tham khảo các tài liệu trong nước, ngoài nước cũng như nhiều ý kiến tham gia của các thầy, các bạn đồng nghiệp đã dạy và nghiên cứu môn CSDL, song cuốn giáo trình vẫn không tránh khỏi những thiếu sót. Rất mong nhận được sự đóng góp ý kiến của bạn đọc và các em học sinh.

TÁC GIẢ

Chương 1

SƠ LƯỢC VỀ CƠ SỞ DỮ LIỆU

Mục tiêu:

Học sinh hiểu được các khái niệm chung về cơ sở dữ liệu, một số mô hình cơ sở dữ liệu và các kiến thức quan trọng về mô hình thực thể - liên hệ, làm nền tảng cho việc thiết kế các lược đồ cơ sở dữ liệu khái niệm trong các mô hình khác.

Nội dung tóm tắt:

Nội dung chương 1 trình bày một số khái niệm cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu, sơ lược về một số mô hình cơ sở dữ liệu và trình bày các kiến thức quan trọng đối với mô hình thực thể - liên hệ.

NỘI DUNG

I. CÁC KHÁI NIỆM CƠ BẢN

1. Cơ sở dữ liệu

Cơ sở dữ liệu (Database, viết tắt là CSDL) là một lĩnh vực nghiên cứu các mô hình, nguyên lý, phương pháp tổ chức dữ liệu trên các vật mang tin.

Để dễ dàng cho việc giải thích các khái niệm, trước hết ta thử xem xét hệ thống bán vé máy bay bằng máy tính. Dữ liệu lưu trữ trong máy tính bao gồm thông tin về hành khách, chuyến bay, đường bay v.v. Mọi thông tin về mối quan hệ này được biểu diễn trong máy thông qua việc đặt chỗ của khách hàng. Vậy làm thế nào để biểu diễn được dữ liệu đó và đảm bảo cho hành khách đi đúng chuyến? Dữ liệu nêu trên được lưu trữ trong máy theo một quy định nào đó và được gọi là cơ sở dữ liệu.

Như vậy, CSDL là tập hợp các thông tin có quan hệ với nhau được lưu trữ

trong máy tính theo một quy định nhất định nhằm phục vụ cho một mục đích nào đó.

2. Hệ quản trị cơ sở dữ liệu (Database Management System - HQTCSDL)

- Hệ quản trị cơ sở dữ liệu là hệ thống các chương trình nhằm tạo lập, xử lý, thay đổi, quản lý và khai thác các CSDL.

- Một số HQTCSDL thông dụng hiện nay: Foxpro, Access, Oracale,... với các phiên bản khác nhau.

- Có hai đặc điểm để phân biệt một HQTCSDL với các hệ thống lập trình khác:

(1) Khả năng quản lý những dữ liệu cố định.

(2) Khả năng truy xuất có hiệu quả một số lượng lớn dữ liệu.

Điểm (1) khẳng định rằng có một CSDL tồn tại thường xuyên và nội dung của CSDL này là những dữ liệu được HQTCSDL truy xuất và quản lý. Điểm (2) phân biệt một HQTCSDL với các hệ thống quản lý tập tin cùng quản lý dữ liệu cố định nhưng nói chung không cho phép truy xuất nhanh chóng các thành phần tuỳ ý của dữ liệu. Ngoài ra, còn có một số khả năng khác thường gặp trong các HQTCSDL trên thị trường:

+ HQTCSDL hỗ trợ ít nhất một mô hình dữ liệu, nhờ đó người sử dụng có thể xem được dữ liệu.

+ HQTCSDL hỗ trợ một số ngôn ngữ bậc cao cho phép người sử dụng định nghĩa các cấu trúc dữ liệu, truy xuất dữ liệu và thao tác dữ liệu.

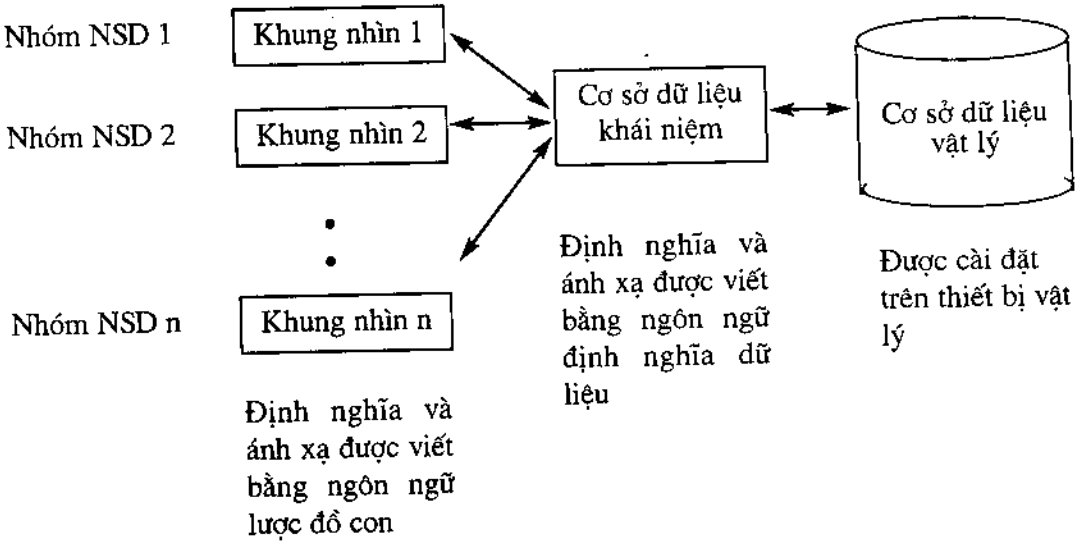
+ HQTCSDL quản lý các giao dịch, nghĩa là cho phép nhiều người sử dụng truy xuất đồng thời và chính xác đến một CSDL.

+ HQTCSDL điều khiển quá trình truy xuất, là khả năng giới hạn các quá trình truy xuất dữ liệu của những người không được phép và khả năng kiểm tra độ tin cậy của dữ liệu.

+ HQTCSDL có khả năng tự thích ứng là khả năng phục hồi lại dữ liệu do sự cố của hệ thống mà làm mất dữ liệu.

3. Kiến trúc một hệ cơ sở dữ liệu

Một CSDL được phân thành 3 mức trừu tượng khác nhau: Mức CSDL vật lý, mức CSDL khái niệm và mức khung nhìn.



Hình 1.1

* *Mức CSDL vật lý*

Một tập hợp các tệp dữ liệu, các chỉ mục hoặc những cấu trúc lưu trữ khác dùng để truy xuất dữ liệu một cách có hiệu quả gọi là CSDL vật lý. CSDL vật lý tồn tại thường xuyên trong thiết bị lưu trữ như đĩa từ, nhiều CSDL có thể được quản lý bởi cùng một HQTCSDL.

* *Mức CSDL khái niệm*

CSDL khái niệm là một sự trừu tượng hoá của thế giới thực khi nó được gắn với người sử dụng. HQTCSDL cung cấp ngôn ngữ định nghĩa dữ liệu (Data Definition Language viết tắt là DDL) để mô tả lược đồ khái niệm (Conceptual Scheme) và nó sẽ được cài đặt bằng lược đồ vật lý (Physical Scheme). DDL cho phép ta mô tả CSDL khái niệm nhờ các thuật ngữ của "mô hình dữ liệu". Ví dụ: Chẳng hạn trong mô hình quan hệ, dữ liệu được trình bày qua các bảng với các cột là các thuộc tính, các hàng là các bộ.

CSDL vật lý là sự cài đặt cụ thể của CSDL mức khái niệm.

** Mức CSDL khung nhìn*

Khung nhìn (view) hay lược đồ con (SubScheme) là một phần của CSDL khái niệm hoặc là sự trừu tượng hoá một phần CSDL khái niệm.

Hay nói một cách khác, View là cách nhìn của người sử dụng đối với CSDL, là một đối tượng tưởng tượng được xây dựng từ CSDL khái niệm nhưng thực sự không tồn tại trong CSDL đó.

Hầu hết các HQTCSDL đều cung cấp những phương tiện để khai báo khung nhìn được gọi là ngôn ngữ định nghĩa dữ liệu lược đồ con (SubScheme Data Definition Language) và các phương tiện để diễn đạt các câu vấn tin và thao tác trên khung nhìn gọi là ngôn ngữ thao tác dữ liệu lược đồ con (SubScheme Data Manipulation Language). Theo một nghĩa nào đó, việc xây dựng khung nhìn ngược lại với quá trình tích hợp CSDL. Đối với một tập hợp dữ liệu tham gia vào CSDL khái niệm, ta có thể xây dựng một khung nhìn chứa dữ liệu này. Các khung nhìn có vai trò quan trọng trong việc bảo mật hệ thống CSDL, chỉ những người có nhu cầu và quyền hạn mới có thể đọc được các tệp dữ liệu dành cho họ.

Ví dụ 1.1: Để phân tích sự khác nhau giữa các mức vật lý, mức khái niệm và mức khung nhìn nhờ tính tương tự của chúng đối với các ngôn ngữ lập trình cụ thể. Hãy xét một mảng hai chiều có kích thước m x n.

Ở mức khái niệm có thể khai báo mảng như sau:

A: array[1...m,1...n] of Integer;

Ở mức vật lý mảng A được lưu trữ trong một vùng nhớ liên tục nhờ quy tắc: A[i, j] sẽ ở vị trí $a_0 + 2(n(i-1) + j-1)$.

Một khung nhìn của mảng A có thể tạo ra bằng cách khai báo một hàm f(i) là tổng của A[i, j] với ($i = \overline{1,m}$, $j = \overline{1,n}$). Trong khung nhìn này không chỉ nhìn thấy A vừa có liên hệ vừa tách biệt, là hàm chứ không phải là mảng, mà còn che dấu được một số thông tin, bởi vì chỉ có thể thấy được tổng của các hàng mà không phải bản thân các hàng này.

Ví dụ 1.2: Có thể trừu tượng hoá các sinh viên qua các thuộc tính: MaSV, TenSV, NgaySinh, Gioi, Quequan. Điểm của các sinh viên và mối liên hệ giữa các sinh viên và điểm của họ qua các thuộc tính: MaSV, Mon1, Mon2, Mon3. Chẳng hạn dùng HQTCSDL Visual Foxpro thì:

- Ở mức CSDL khái niệm có cấu trúc của bảng SV và bảng DIEM, có thể khai báo như sau:

+ Create table SV(MaSV char (10) not null, TenSV char (40), NgaySinh date, Gioi logical, Quequan char (50)).

+ Create table Diem (MaSV char (10), Mon1 Number (5,2), Mon2 Number (5,2), Mon3 Number (5,2)).

- Ở mức CSDL vật lý có các bảng SV.dbf và Diem.dbf trên đĩa từ.

- Ở mức khung nhìn muốn có kết quả tổng hợp (một view) của các sinh viên bao gồm các thông tin: TenSV, Tuoi, Mon1, Mon2, Mon3 thì khai báo như sau: Create View KQ(TenSV, Tuoi, Mon1, Mon2, Mon3) as (Select TenSV, year(date())-year(NgaySinh), Mon1, Mon2, Mon3 from SV, Diem where SV.MaSV = Diem.MaSV).

4. Lược đồ (Scheme) và Thể hiện (Instance)

Ngoài việc phân chia các mức trừu tượng như trên, còn có một cách hiểu khác về tính hai mặt của CSDL đó là lược đồ và thể hiện. Khi thiết kế thì quan tâm đến những hoạch định trên CSDL, đó chính là lược đồ của CSDL, nhưng khi sử dụng thì lại quan tâm đến dữ liệu thực sự tồn tại trong CSDL, đó là thể hiện của CSDL. Lược đồ thường không thay đổi trong khi đó các thể hiện lại thường xuyên thay đổi.

* *Lược đồ (Scheme)*

Lược đồ là bộ khung hay cấu trúc của CSDL, nó thường bao gồm một số danh mục, chỉ tiêu hoặc một số kiểu của thực thể trong CSDL.

* *Thể hiện (Instance)*

Thể hiện của CSDL là dữ liệu hiện có trong CSDL.

Tương ứng với mức CSDL vật lý có lược đồ vật lý, tương ứng với mức CSDL khái niệm có lược đồ khái niệm và tương ứng với khung nhìn có lược đồ con.

Ví dụ 1.3: Mô tả mảng và hàm được nêu ra trong ví dụ 1.2, đó thực sự là một thông tin của lược đồ.

1- Lược đồ vật lý chính là khẳng định mảng A được lưu trong bộ nhớ bắt đầu từ vị trí a_0 và giá trị A[i, j] được lưu trong $a_0 + 2(n(i-1) + j-1)$.

2- Lược đồ khái niệm là khai báo:

A : array[1...m,1...n] of Integer;

3- Lược đồ con là hàm f(i) được định nghĩa như sau:

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

Để đưa ra ví dụ về thể hiện của lược đồ khái niệm này, ta cho m = n = 3 và cho A là một ma phương:

8	1	6
3	5	7
4	9	2

Thế thì thể hiện vật lý sẽ là 9 từ máy liên tiếp bắt đầu từ vị trí a_0 theo thứ tự chứa 8, 1, 6, 3, 5, 7, 4, 9, 2. Cuối cùng thể hiện khung nhìn là hàm f(1) = f(2) = f(3) = 15.

** Tính độc lập dữ liệu*

Theo hình 1.1, đi từ khung hình qua CSDL khái niệm đến CSDL vật lý cho thấy có hai mức "độc lập dữ liệu". Thứ nhất: lược đồ vật lý có thể thay đổi do người quản trị CSDL mà không cần thay đổi lược đồ khái niệm hay phải định nghĩa lại lược đồ con. Ví dụ, ta có thể tham chiếu đến mảng A trong ví dụ 1.1 và ví dụ 1.3 vẫn sẽ đúng bất kể việc cài đặt vật lý là theo hàng hay theo cột. Việc tổ chức lại CSDL vật lý (thay đổi các tổ chức, cấu trúc dữ liệu trên các thiết bị nhớ thứ cấp) có thể làm thay đổi hiệu quả tính toán của các chương trình ứng dụng nhưng không đòi hỏi phải viết lại các chương trình đó. Tính độc lập này gọi là độc lập dữ liệu mức vật lý.

Mối quan hệ giữa các khung nhìn và lược đồ khái niệm cho thêm một loại độc lập nữa, gọi là độc lập dữ liệu lôgic. Khi sử dụng một CSDL, có thể cần thiết phải thay đổi lược đồ khái niệm như thêm thông tin về các loại thực thể hay bổ sung hoặc xoá bớt các thông tin về các thực thể đang tồn tại trong CSDL. Việc thay đổi lược đồ khái niệm không làm ảnh hưởng tới các lược đồ con đang tồn tại, do đó không cần thiết phải thay đổi các chương trình ứng dụng.

Vì thế, tính độc lập dữ liệu là mục tiêu chủ yếu của các hệ CSDL. Có thể định nghĩa tính độc lập dữ liệu là "tính bất biến của các hệ ứng dụng đối với những thay đổi trong cấu trúc lưu trữ và chiến lược truy nhập" (Date).

5. Các ngôn ngữ CSDL

Trong các ngôn ngữ lập trình, thông thường tất cả các khai báo và các câu lệnh khả thi đều là thành phần của ngôn ngữ. Trong thế giới CSDL thường có hai chức năng riêng rẽ là khai báo và tính toán, chúng được tách thành hai ngôn ngữ khác nhau. Vấn đề là ở chỗ, đối với các chương trình thông thường, dữ liệu chỉ tồn tại khi chương trình đang thực hiện, còn trong hệ thống CSDL dữ liệu luôn hiện hữu và có thể định nghĩa một lần duy nhất. Vì vậy, nếu có một phương tiện riêng để định nghĩa dữ liệu sẽ có ý nghĩa hơn.

** Ngôn ngữ định nghĩa dữ liệu (Data Definition Language - DDL)*

Như ta đã biết ở trên, lược đồ khái niệm được đặc tả bằng một ngôn ngữ, được cung cấp như là một thành phần của HQTCSDL và được gọi là ngôn ngữ định nghĩa dữ liệu. Đây không phải là một ngôn ngữ thủ tục mà thực ra là một hệ thống ký hiệu để mô tả các kiểu thực thể và mối liên hệ giữa chúng theo một mô hình dữ liệu cụ thể nào đó. Ví dụ: Có thể định nghĩa kiểu thực thể sinh viên trong mô hình quan hệ trên (bảng SV) như sau:

Create table SV(MaSV char(10) not null, TenSV char(40), NgaySinh date, Gioi logical, Quequan char(50))

Đây là một ví dụ về ngôn ngữ định nghĩa dữ liệu SQL.

Ngôn ngữ định nghĩa dữ liệu được sử dụng khi thiết kế CSDL và cả khi thiết kế này cần sửa đổi. Nó không được dùng để lấy dữ liệu hay sửa đổi dữ liệu. Ngôn ngữ định nghĩa dữ liệu có những câu lệnh để mô tả cấu hình vật lý (Physical layout) theo những thuật ngữ trừu tượng như ví dụ trên. Thiết kế chi tiết cho CSDL vật lý được thực hiện bởi các thủ tục của HQTCSDL, chúng sẽ biên dịch các câu lệnh trong ngôn ngữ định nghĩa dữ liệu.

Mô tả các lược đồ con và tính tương ứng của chúng đối với lược đồ khái niệm phải sử dụng đến ngôn ngữ định nghĩa dữ liệu lược đồ con (Subscheme Data Definition Language), thường thì ngôn ngữ này chính là ngôn ngữ định nghĩa dữ liệu.

** Ngôn ngữ thao tác dữ liệu (Data Manipulation Language - DML)*

Các thao tác trên CSDL đòi hỏi phải có một ngôn ngữ đặc biệt gọi là ngôn ngữ thao tác dữ liệu hay còn gọi là ngôn ngữ vấn tin (Query language) để diễn tả các câu lệnh như:

- Đưa ra thông tin của các sinh viên nữ có quê quán tại Hà Nội.

- Đưa ra thông tin của các sinh viên gồm mã sinh viên, tên sinh viên, tuổi.

- Thêm sinh viên có mã sinh viên là 123, tên là Nguyễn Hải Đăng, sinh ngày 18 tháng 9 năm 1966, giới tính là nam và quê ở Hải Phòng.

- Xóa đi sinh viên có mã sinh viên là 12.

- Sửa lại tên sinh viên là Phạm Hữu Hải cho sinh viên có mã sinh viên là 14.

Chúng dùng để truy xuất, tìm kiếm, cập nhật thông tin đối với CSDL. Thuật ngữ "ngôn ngữ vấn tin" được dùng như một từ đồng nghĩa với thuật ngữ DML, nhưng chỉ có một số câu lệnh của DML là vấn tin. (Chúng rút ra thông tin từ CSDL mà không sửa đổi gì, những câu lệnh khác có thể sửa đổi CSDL nên không phải là những câu vấn tin mặc dù chúng có thể được diễn tả trong ngôn ngữ vấn tin).

* *Ngôn ngữ chủ (Host Language)*

Thông thường, việc thao tác trên CSDL được thực hiện bởi một chương trình ứng dụng đã được viết trước để thực hiện một nhiệm vụ nào đó. Đối với một chương trình ứng dụng cần làm được nhiều việc hơn là chỉ thao tác với CSDL, ví dụ thực hiện các tính toán phức tạp. Vì vậy, chương trình để thao tác CSDL thường được viết trong một ngôn ngữ chủ, là một ngôn ngữ lập trình thông thường chẳng hạn như: C hay Basic v.v.

Các lệnh của DML được kích hoạt bởi chương trình ngôn ngữ chủ theo hai cách tuỳ thuộc vào đặc tính của HQTCSDL.

- Các lệnh của DML được kích hoạt bằng việc gọi các thủ tục được cung cấp bởi HQTCSDL.

- Các lệnh là các câu lệnh trong một ngôn ngữ, được xem là phần mở rộng của ngôn ngữ chủ.

II. MỘT SỐ MÔ HÌNH CSDL

1. Mô hình CSDL (Database Model)

Mô hình CSDL là một hệ hình thức toán học gồm có hai phần:

- Một hệ thống ký hiệu để mô tả dữ liệu.

- Một tập hợp các phép toán thao tác trên dữ liệu đó.

2. Một số mô hình CSDL thông dụng

- Mô hình thực thể liên hệ (Entity Relationship model): là mô hình cho

phép mô tả các thực thể thông qua các thuộc tính và mối liên hệ giữa các thực thể. Một trong các cách biểu thị mô hình thực thể là dùng đồ thị, sơ đồ khối.

- Mô hình mạng (Network model): là mô hình thực thể liên hệ trong đó các mối liên hệ bị hạn chế trong kiểu nhị phân (hai thực thể) và nhiều - một hoặc một - một và được biểu diễn bởi một đồ thị có hướng.

- Mô hình phân cấp (Hierachical model): là mô hình mạng có nhiều cây trong đó tất cả các đường nối chỉ đi theo hướng từ con đến cha.

- Mô hình quan hệ (Relational model): là mô hình dựa vào ký hiệu là tập các tên và cơ sở toán học của nó là các phép toán tập hợp và ánh xạ. Nó là mô hình phổ biến hiện nay. Tập các phép toán trong mô hình này dựa trên hai hệ ký hiệu: hệ ký hiệu đại số và hệ ký hiệu logic.

- Mô hình hướng đối tượng (Object Oriented model): là mô hình cung cấp đặc tính nhận dạng đối tượng. Trong đó mỗi lớp đối tượng được đặc trưng bởi hai yếu tố:

+ Tập các thuộc tính (properties) để nhận dạng đối tượng.

+) Tập các phương thức (methods) để thao tác với đối tượng.

III. MÔ HÌNH THỰC THỂ - LIÊN HỆ

Mục đích của mô hình thực thể - liên hệ là cho phép mô tả lược đồ khái niệm của một tổ chức mà không cần chú ý đến tính hiệu quả hoặc thiết kế CSDL vật lý được mong đợi như ở phần lớn các mô hình khác. Người ta thừa nhận rằng: "Sơ đồ thực thể - liên hệ (Entity Relationship Diagram) có thể chuyển về lược đồ khái niệm ở các mô hình khác" (ví dụ mô hình quan hệ) mà trên đó các hệ thống CSDL thực sự được xây dựng một cách khá đơn giản.

1. Thực thể

Thuật ngữ thực thể (Entity) không có một định nghĩa hình thức. Thực thể là một sự vật tồn tại và phân biệt thực thể này với thực thể khác. Ví dụ mỗi con người là một thực thể, mỗi chiếc xe máy là một thực thể. Khái niệm về "tính phân biệt được" rất gần với "đặc tính nhận dạng đối tượng" vì thế mô hình thực thể liên hệ được xem như là mô hình hướng đối tượng.

2. Tập thực thể

Một nhóm bao gồm tất cả các thực thể "tương tự" tạo ra một tập thực thể.

Ví dụ 1.4: Các tập thực thể:

+ Tất cả những người trong một cơ quan.

+ Tất cả những người có tóc đỏ.

+ Tất cả những người có xe gắn máy.

3. Thuộc tính và khoá

- Thuộc tính: Các đặc tính của tập thực thể gọi là các thuộc tính. Mỗi thuộc tính của tập thực thể lấy giá trị trên một miền dành cho thuộc tính đó. Thường thì miền giá trị đối với mỗi thuộc tính là một tập số nguyên, tập các số thực hoặc chuỗi ký tự nhưng cũng không loại trừ các kiểu giá trị khác.

- Khóa: Mỗi thuộc tính hoặc một tập các thuộc tính dùng để xác định một cách duy nhất mỗi thực thể trong một tập thực thể gọi là khóa đối với tập thực thể đó. Về nguyên tắc, mỗi thực thể có một khóa, bởi vì mỗi thực thể đều có thể phân biệt được với thực thể khác. Nếu không chọn được một tập các thuộc tính có chứa một khóa cho một tập thực thể thì không có khả năng phân biệt được thực thể này với thực thể kia trong tập thực thể đó. Trong trường hợp này thì các số đếm thường được gán làm thuộc tính khóa.

Ví dụ 1.5: Một tập thực thể chỉ bao gồm các công dân Việt Nam có thể dùng thuộc tính "Số chứng minh thư" (IDNO) làm khoá. Tuy nhiên, nếu muốn xác định một cách duy nhất các công dân của nhiều quốc gia thì không thể đảm bảo được hai quốc gia sẽ không dùng hai số chứng minh thư giống nhau. Vì vậy, một khoá thích hợp phải gồm một cặp thuộc tính IDNO và COUNTRY (nước).

4. Phân cấp ISA

Ta nói rằng A isa B (đọc là A là một B) nếu tập thực thể B là sự tổng quát hóa của tập thực thể A, hoặc tương đương A là một loại B đặc biệt. Mục đích chính của việc khai báo những mối liên hệ ISA giữa các tập thực thể A và B là: A có thể kế thừa các thuộc tính của B, nhưng A có thể có thêm những thuộc tính khác.

Ví dụ 1.6: Một cơ quan có một tập thực thể NHANVIEN với các thuộc tính là MANV, TENNV, LUONG. Nếu cơ quan này có một Đảng bộ thì các nhân viên là đảng viên sẽ có các thuộc tính quan trọng khác như: NGAY VD mà những nhân viên khác không có. Cách dễ dàng nhất để thiết kế lược đồ này là tạo ra một tập thực thể DANGVIEN có mối liên hệ DANGVIEN isa

NHANVIEN, các thuộc tính của NHANVIEN được DANGVIEN kế thừa nhưng chỉ có DANGVIEN mới có thuộc tính NGAYVD (ngày vào Đảng).

5. Mối liên hệ

Mối liên hệ (Relationship) giữa các tập thực thể là một danh sách có thứ tự của các tập thực thể. Một tập thực thể đặc biệt có thể xuất hiện nhiều lần trong danh sách. Danh sách các tập thực thể này là một khái niệm ở mức lược đồ của một mối liên hệ. Nếu có một mối liên hệ R giữa các tập thực thể E_1, E_2, E_3,..., E_k thì thể hiện của R là một tập các k - bộ, một tập như thế được gọi là một tập liên hệ. Mỗi k - bộ $(e_1,..., e_k)$ trong tập liên hệ R khẳng định rằng các thực thể e_1, e_2,..., e_k trong đó $e_i \in E_i$; i =1,..., k liên kết với nhau thành một nhóm trong mối liên hệ R. Trường hợp thông dụng nhất là k = 2.

Ví dụ 1.7: Có một tập thực thể PERSONS có mối liên hệ MOTHER_OF và danh sách các thực thể của nó là PERSONS1, PERSONS2. Tập liên hệ tương ứng với mối liên hệ MOTHER_OF gồm tất cả những cặp (P_1, P_2) sao cho cá thể P_2 là mẹ cá thể P_1.

Một cách khác để biểu diễn thông tin này là thừa nhận tồn tại tập thực thể MOTHERS và mối liên hệ MOTHERS isa PERSONS. Do vậy, mối liên hệ MOTHER_OF là danh sách các tập PERSONS, MOTHERS.

6. Thuộc tính khoá vay mượn (khoá ngoại – foreign key)

Trong mối liên hệ ISA nếu A isa B thì đương nhiên khoá của A sẽ là khoá của B và những thuộc tính này không xuất hiện như thuộc tính của tập A mà vay mượn của tập B.

Ví dụ 1.8: (Trong ví dụ 1.6), khoá của tập DANGVIEN sẽ là thuộc tính MANV của NHANVIEN. Vì vậy, một Đảng viên được xác định một cách duy nhất nhờ vào MANV của NHANVIEN.

Nhiều khi thuộc tính khoá của tập thực thể A là một thuộc tính của tập thực thể B nhờ mối liên hệ R không phải ISA. Điều này cần thiết để cung cấp cho mỗi thực thể a trong A một thực thể b trong B.

Ví dụ 1.9: Chẳng hạn, trong ví dụ 1.5 mỗi công dân đều có thuộc tính COUNTRY và cùng với IDNO để tạo ra một khoá cho mỗi công dân, điều đó có nghĩa là việc thiết kế CSDL đã xem các quốc gia như một kiểu thực thể khác và có một mối liên hệ COUNTRY_OF liên kết giữa các công dân với các quốc gia. Vì vậy, các công dân đã phải mượn thuộc tính COUNTRY để

cùng với IDNO làm khoá. Trong các ví dụ trên thuộc tính MANV, COUNTRY là các thuộc tính khoá vay mượn (foreign key).

7. Sơ đồ thực thể liên hệ

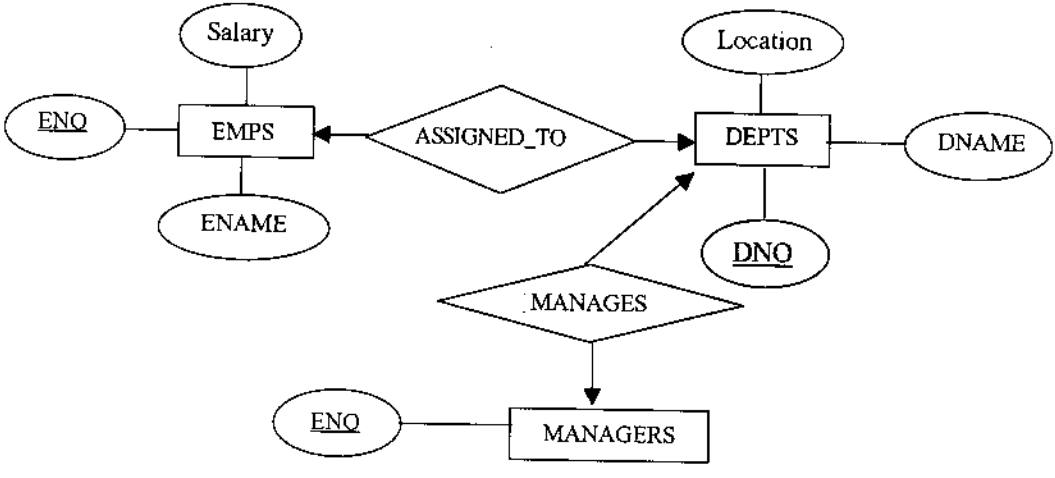
* *Quy ước:*

- Các hình chữ nhật biểu diễn các tập thực thể.

- Các vòng tròn biểu diễn các thuộc tính. Chúng được liên kết với các tập thực thể bằng các cạnh (vô hướng). Các thuộc tính là các thành phần của một khóa cho một tập thực thể sẽ được gạch dưới. Trường hợp đặc biệt, nếu một tập thực thể chỉ có một thuộc tính thì có thể gọi tập thực thể đó bằng tên thuộc tính của tập. Khi đó, tập thực thể sẽ là một vòng tròn chứ không phải là hình chữ nhật và nó gắn kết với các mối liên hệ mà tập đó hàm chứa.

- Các hình thoi biểu diễn các mối liên hệ. Chúng được liên kết với các tập thành viên bởi các cạnh vô hướng hoặc có hướng (các cung).

Ví dụ 1.10: Giả sử ta có 3 tập thực thể EMPS (Nhân viên) có các thuộc tính ENO, ENAME, Salary (ENO là khoá); DEPTS (Phòng) có các thuộc tính DNO, DNAME, Location (DNO là khoá); MANAGERS (Trưởng phòng) có duy nhất thuộc tính ENO (ENO là khoá). Hai tập đầu liên kết nhờ mối liên hệ ASSIGNED_TO (thuộc phòng) và hai tập sau nhờ liên hệ MANAGES (quản lý).



Hình 1.2: Ví dụ mô hình thực thể - liên hệ

8. Tính chất hàm của mối liên hệ

Để mô hình hóa đầy đủ thế giới thực cần phải phân loại các mối liên hệ theo số lượng các thực thể từ mỗi tập tham gia vào trong mối liên hệ.

8.1. Mối liên hệ một - một

Một mối liên hệ một - một (one to one relationship) là một mối liên hệ mà với mỗi thực thể trong một tập thực thể này chỉ có nhiều nhất một phần tử được liên kết trong tập thực thể kia.

Ví dụ 1.11: Mối liên hệ MANAGES giữa DEPTS và MANAGER trong ví dụ trên được khai báo là mối liên hệ một - một. Nếu như vậy, trong CSDL không bao giờ tìm được nhiều trưởng phòng cho một phòng và cũng không có người nào quản lý nhiều phòng. Có thể là tại một thời điểm, một phòng nào đó không có trưởng phòng, thậm chí cũng có thể có người có tên trong danh sách trưởng phòng lại không quản lý một phòng nào cả. Tính một - một của mối liên hệ này chỉ là một giả thiết về thế giới thực. Vì vậy, người thiết kế CSDL có thể tuỳ ý chọn lựa. Vẫn có thể cho rằng, một người lãnh đạo 2 phòng thậm chí 1 phòng có 2 trưởng phòng khi đó mối quan hệ trên là nhiều - nhiều. Tuy nhiên, nếu cho MANAGES là mối liên hệ một - một sẽ có ích hơn khi thiết kế CSDL vật lý.

8.2. Mối liên hệ nhiều - một

Một mối liên hệ giữa các tập thực thể E_1 và E_2 được gọi là mối liên hệ nhiều - một (many - one relationship) nếu mỗi thực thể trong tập E_2 có thể không liên kết với thực thể nào hoặc liên kết với một hay nhiều thực thể trong tập E_1, nhưng mỗi thực thể trong tập thực thể E_1 chỉ liên kết nhiều nhất với một thực thể trong tập thực thể E_2.

Ví dụ 1.12: Mối liên hệ ASSIGNED_TO giữa EMPS và DEPTS ở ví dụ trên là mối liên hệ nhiều - một, có nghĩa là mỗi nhân viên chỉ làm việc trong một phòng, một vài nhân viên như giám đốc chẳng hạn không gán cho một phòng nào cả và một phòng có nhiều nhân viên.

Khái niệm liên hệ nhiều - một tổng quát hóa thành mối liên hệ giữa ba tập trở lên. Nếu có một mối liên hệ R giữa các tập E_1, E_2,..., E_k và với các thực thể trong tất cả các tập thực thể, trừ E_i chỉ có nhiều nhất một thực thể của E_i có liên hệ với chúng thì ta gọi R là mối liên hệ nhiều - một từ E_1, E_2,..., E_k (trừ E_i) đến E_i.

8.3. Mối liên hệ nhiều - nhiều

Chúng ta cũng gặp mối liên hệ nhiều - nhiều, ở đó không có một hạn chế nào trên tập k - bộ của các thực thể khi xuất hiện trong tập liên hệ.

Một mối liên nhiều - nhiều (many - many relationship) là một mối liên hệ mà với mỗi thực thể trong một tập thực thể này có thể không liên kết với thực thể nào hoặc liên kết với một hay nhiều thực thể trong tập thực thể kia.

Ví dụ 1.13: Mối liên hệ SUPPLIES giữa tập thực thể SUPPLIERS gồm các thuộc tính SNO, SNAME, SADDR và tập thực thể PRODUCTS gồm các thuộc tính PNO, PNAME, COLOR, WEIGHT là mối liên hệ nhiều - nhiều.

Trong thực hành các mối liên hệ nhiều - nhiều thường hay gặp nên phải cẩn thận trong cách diễn đạt những mối liên hệ này trong lược đồ khái niệm của CSDL thực sự (thiết kế thực thể - liên hệ không phải là lược đồ khái niệm, đúng hơn chỉ là bảng phác thảo và cần chuyển đổi các tập thực thể và các mối liên hệ về mô hình dữ liệu mà DBMS hỗ trợ). Nhiều mô hình dữ liệu không cho phép biểu diễn trực tiếp các mối liên hệ nhiều - nhiều, yêu cầu phải phân chúng ra thành các mối liên hệ nhiều - một.

9. Biểu diễn tính chất hàm trong các sơ đồ thực thể - liên hệ

Các sơ đồ thực thể - liên hệ dùng các cung, đó là các cạnh có hướng chỉ ra bởi một mũi tên cho biết khi nào có mối liên hệ là nhiều - một hay một - một. Trong trường hợp mối liên hệ nhiều - một R từ A đến B thì đặt một cung từ hình thoi R đến hình chữ nhật B. Chẳng hạn, giả sử rằng mỗi nhân viên chỉ được phân công tối đa vào một phòng, điều đó giải thích mũi tên đi từ ASSIGNED_TO tới DEPTS như hình vẽ 1.2 ở trên.

Tổng quát, nếu mối liên hệ R gồm nhiều hơn hai tập và thuộc loại nhiều - một vào một tập A nào đó thì sẽ vẽ một cung từ R đến A và các cạnh vô hướng đến các tập khác.

Nếu R là mối liên hệ một - một giữa A và B thì vẽ một mũi tên từ R đến cả A và B. Giả sử rằng, các trưởng phòng chỉ quản lý một phòng và mỗi phòng chỉ có một trưởng phòng. Điều đó giải thích cho các cung đi từ MANAGES đến DEPTS và MANAGERS. Ngoại lệ, nếu A isa B thì chỉ vẽ một cung đến B.

Ví dụ 1.14: Một siêu thị nhỏ ST muốn quản lý công việc kinh doanh của mình bằng máy tính và có ý định thiết kế một hệ thống CSDL lưu trữ các

thông tin cần thiết cho hoạt động kinh doanh. Nhân viên quản lý CSDL hệ thống là SV A, một sinh viên cao đẳng CNTT đang làm việc cho siêu thị đã phân tích và triển khai một sơ đồ thực thể - liên hệ như sau:

- Một lĩnh vực quan trọng trong công việc kinh doanh của ST là làm việc với các nhà cung cấp hàng, nên SV A quyết định trong CSDL có một tập thực thể SUPPLIERS (các nhà cung cấp) gồm các thuộc tính SNO (số hiệu nhà cung cấp - khoá), SNAME (tên nhà cung cấp), SADDR(địa chỉ).

- Một vấn đề quan trọng liên quan đến các nhà cung cấp là tập các mặt hàng họ cung cấp, do đó SV A đã đưa ra một tập thực thể ITEMS (các mặt hàng) gồm các thuộc tính INO (số hiệu mặt hàng - khoá), INAME (tên mặt hàng), COLOR (màu sắc), WEIGHT (trọng lượng). Mối liên hệ giữa SUPPLIERS và ITEMS là mối liên hệ nhiều - nhiều SUPPLIES (cung cấp) với ý nghĩa là mỗi nhà cung cấp có thể cung cấp nhiều mặt hàng và mỗi mặt hàng có thể được cung cấp bởi nhiều nhà cung cấp. Tuy nhiên, một tập thực thể thứ ba PRICES (giá mặt hàng) cũng được đưa vào mối liên hệ này. Mỗi nhà cung cấp sẽ ấn định giá của mỗi mặt hàng họ cung cấp. Vì vậy, tốt hơn sẽ coi mối liên hệ SUPPLIES là mối liên hệ giữa ITEMS, SUPPLIERS và PRICES với ý nghĩa là nếu mối liên hệ SUPPLIES chứa bộ ba (i, s, p) thì có nghĩa là nhà cung cấp S bán mặt hàng i với giá p. Trong sơ đồ thực thể liên hệ PRICES được biểu diễn là một vòng tròn chứ không phải là hình chữ nhật. Lý do PRICES chỉ có một thuộc tính cũng chính là PRICES và ta vẽ PRICES như một thuộc tính của mối liên hệ SUPPLIES. Vì vậy, SUPPLIES được xem như là một cặp biểu diễn mặt hàng - nhà cung cấp và giá mặt hàng liên quan đến cặp đó. Mối liên hệ này là mối liên hệ nhiều - một từ ITEMS và SUPPLIERS đến PRICES và trong sơ đồ thực thể - liên hệ có một cung có hướng từ SUPPLIES đến PRICES và hai cung vô hướng đến ITEMS và SUPPLIERS. Như vậy, nếu cho trước một nhà cung cấp và một mặt hàng thì chỉ có một giá duy nhất cho mặt hàng đó. Ngoài ra, không thể chia SUPPLIES thành hai hoặc ba mối liên hệ hai ngôi. Ví dụ, nếu chúng ta có một mối liên hệ giữa SUPPLIERS và ITEMS và một mối liên hệ khác giữa SUPPLIERS và PRICES thì mỗi nhà cung cấp buộc phải bán tất cả các mặt hàng cùng một giá. Nếu có một mối liên hệ khác giữa ITEMS và PRICES thì mỗi mặt hàng, tất cả các nhà cung cấp buộc phải bán cùng một giá.

Siêu thị ST có nhiều gian hàng (Department), mỗi gian hàng đều có một gian hàng trưởng và các nhân viên (Employee), nên trong CSDL có hai thực

thể nữa là DEPTS với các thuộc tính DNO và DNAME và một trong chúng là khoá và EMPS với các thuộc tính ENO (khoá), ENAME, SALARY.

Mối liên hệ WORK_IN từ EMPS đến DEPTS là mối liên hệ nhiều - một, phản ánh quy định là mỗi nhân viên chỉ được làm việc tại một gian hàng.

Mỗi gian hàng chịu trách nhiệm bán một số mặt hàng và quy định của siêu thị là mỗi mặt hàng chỉ được bán ở một gian hàng. Vì vậy, có một mối liên hệ nhiều - một CARRIES từ ITEMS đến DEPTS. .

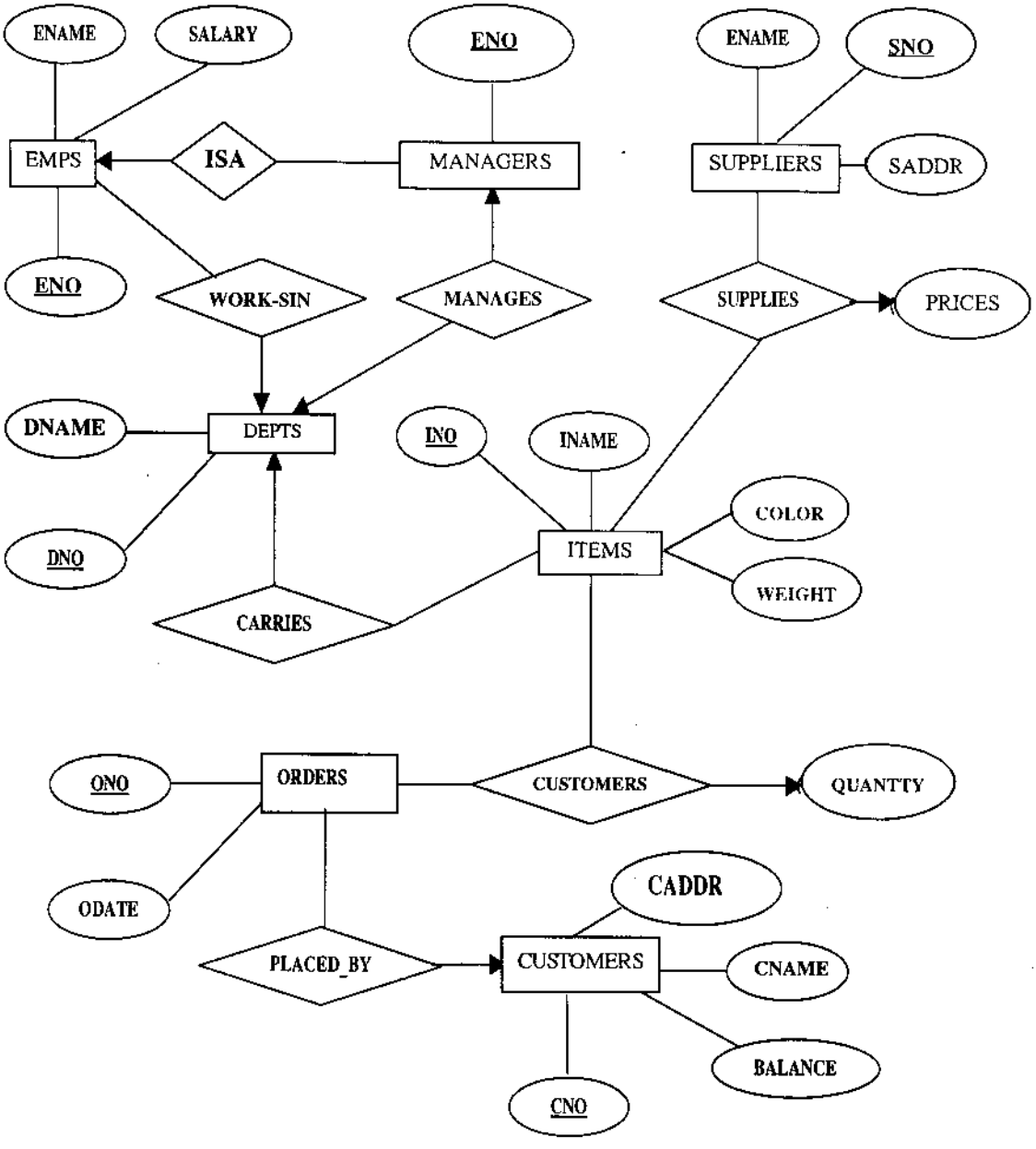
Các gian hàng trưởng được biểu diễn qua tập thực thể MANAGERS. Có một mối liên hệ một - một MANAGES giữa MANAGERS và DEPTS. Điều này cho thấy ở ST không có hai gian hàng trưởng cho cùng một gian hàng, hoặc một gian hàng trưởng quản lý hai gian hàng. Sau nữa, mỗi gian hàng trưởng cũng là một nhân viên nên có một mối liên hệ isa từ MANAGERS đến EMPS. Để truy xuất tên, lương của một gian hàng trưởng nào đó chỉ việc thông qua mối liên hệ isa này đến thực thể nhân viên là gian hàng trưởng cần tìm và tìm các thông tin đó trong các thuộc tính ENAME, SALARY của EMPS.

Một tập thực thể quan trọng khác là các khách hàng CUSTOMERS với các thuộc tính CNO, CNAME, CADDR, BALANCE. Thuộc tính số hiệu khách hàng CNO là khoá, các thuộc tính còn lại là tên của khách hàng, địa chỉ của khách hàng và số dư của khách hàng.

Các khách hàng đặt mua hàng qua các đơn đặt hàng. Mỗi đơn đặt hàng (order) gồm một danh sách các mặt hàng (item) và số lượng (quantity) theo yêu cầu khách hàng. Các thuộc tính của tập thực thể ORDERS là ONO (số hiệu đơn hàng - khoá), ODATE (ngày đặt hàng) nhưng nội dung thực sự của các đơn hàng được biểu diễn qua mối liên hệ INCLUDES giữa ORDERS, ITEMS và QUANTITY. Tập thực thể QUANTITY chỉ có duy nhất là giá trị của bản thân nó, vì vậy chỉ cần biểu diễn bằng một vòng tròn và gắn nó với mối liên hệ INCLUDES. Mối liên hệ này là nhiều - một từ ITEMS và ORDERS đến QUANTITY, bởi vì mỗi đơn đặt hàng chỉ có một số lượng duy nhất cho mỗi mặt hàng được đặt.

Cuối cùng, mối liên hệ nhiều - một PLACED_BY (đặt bởi) từ ORDERS đến CUSTOMERS cho biết khách hàng nào đã đặt những mặt hàng nào.

Ta có sơ đồ thực thể - liên hệ như hình vẽ sau (trang bên):



Hình 1.3. Sơ đồ thực thể - liên hệ cho CSDL ST

Câu hỏi và bài tập chương 1

1. Nêu khái niệm về CSDL.

2. Nêu khái niệm HQTCSDL, phân biệt nó với các hệ thống lập trình khác.

3. Nêu kiến trúc một hệ CSDL và phân biệt các mức của nó.

4. Nêu khái niệm lược đồ và thể hiện, cho ví dụ.

5. Nêu khái niệm mô hình dữ liệu.

6. Nêu các khái niệm thực thể và tập thực thể. Cho ví dụ.

7. Nêu các khái niệm khoá của tập thực thể và cho ví dụ.

8. Nêu các mối liên hệ giữa các tập thực thể và cho ví dụ.

9. Hãy sử dụng mô hình thực thể - liên hệ để mô tả dữ liệu cho hệ thống quản lý điểm của một lớp, sau đó vẽ sơ đồ thực thể - liên hệ cho CSDL đó.

10. Hãy sử dụng mô hình thực thể - liên hệ để mô tả dữ liệu cho hệ thống quản lý việc cho thuê truyện của một cửa hiệu cho thuê truyện, sau đó vẽ sơ đồ thực thể - liên hệ cho CSDL đó.

11. Hãy sử dụng mô hình thực thể - liên hệ để mô tả dữ liệu cho hệ thống quản lý việc mua bán vật tư của một cửa hàng vật tư, sau đó vẽ sơ đồ thực thể - liên hệ cho CSDL đó.

12. Hãy sử dụng mô hình thực thể - liên hệ để mô tả dữ liệu cho hệ thống quản lý nhân sự của một phòng tổ chức, sau đó vẽ sơ đồ thực thể - liên hệ cho CSDL đó.

13. Hãy sử dụng mô hình thực thể - liên hệ để mô tả dữ liệu cho hệ thống quản lý lương của một phòng tài vụ, sau đó vẽ sơ đồ thực thể - liên hệ cho CSDL đó.

Chương 2

MÔ HÌNH CƠ SỞ DỮ LIỆU QUAN HỆ

Mục đích:

- Cung cấp cho học sinh nắm vững các khái niệm và một số phép toán cơ bản của mô hình CSDL quan hệ, từ đó biết vận dụng các phép toán đại số quan hệ trong ngôn ngữ vấn tin.

- Học sinh hiểu được các kiến thức mô hình thực thể - liên hệ và biết vận dụng để chuyển sơ đồ thực thể - liên hệ về các lược đồ CSDL quan hệ.

Nội dung tóm tắt:

Chương này đi sâu vào trình bày các khái niệm quan trọng cũng như một số phép toán cơ bản của mô hình CSDL quan hệ. Mặt khác, trong chương còn đề cập đến cách chuyển sơ đồ thực thể - liên hệ về các lược đồ CSDL quan hệ cùng với các kiến thức nhằm giúp đơn giản hoá và tinh chỉnh các lược đồ này.

NỘI DUNG

I. KHÁI NIỆM TOÁN HỌC CỦA QUAN HỆ

Khái niệm toán học làm nền tảng cho mô hình quan hệ là quan hệ hiểu theo nghĩa lý thuyết tập hợp, đó là tập con của tích Đề - các của các miền.

- Miền (domain) là một tập các giá trị, chẳng khác gì một kiểu dữ liệu (data type). Ví dụ tập các số nguyên là một miền, tập các xâu ký tự tạo thành tên người trong tiếng Anh có độ dài không quá 30 ký tự là một miền, tập hai số{0,1} cũng là một miền v.v.

- Tích Đề - các: Gọi D_1, D_2, ..., D_n là các miền. Tích Đề - các của n miền này ký hiệu là D_1 x D_2 x... x D_n là tập tất cả n- bộ (n tuples) (v_1, v_2,. .., v_n)

sao cho v_i thuộc D_i, với i = 1...n.

Ví dụ 2.1: n = 2, D_1 = {0,1}, D_2 = {a,b,c}.

Khi đó:

D_1 x D_2 = {(0, a), (0, b), (0, c), (1, a) (1, b), (1, c)}.

Quan hệ (relation) là một tập con của tích Đề - các của một hoặc nhiều miền. Như vậy, mỗi quan hệ có thể là vô hạn. Ở đây luôn giả thiết rằng, quan hệ là một tập hữu hạn.

Ví dụ 2.2: {(0, a), (0, c), (1, a)(1, b)} là một quan hệ, đó là tập con của tích Đề - các D_1 x D_2 được đề cập ở trên. Tập rỗng ∅ cũng là một quan hệ.

Mỗi phần tử của quan hệ gọi là một bộ (tuples). Quan hệ n – ngôi là tập con của tích Đề-các D_1 x D_2 x... x D_n của n - miền. Khi đó mỗi bộ của quan hệ có n thành phần (v_1, v_2, ..., v_n) được gọi là một n - bộ.

Có thể xem một quan hệ như một bảng (table), trong đó mỗi hàng (row) là một bộ và mỗi cột (column) tương ứng với một miền. Cột thường được đặt tên và gọi là thuộc tính. Tập các tên thuộc tính cho một quan hệ được gọi là một lược đồ quan hệ (Relation Scheme). Nếu chúng ta đặt tên cho một quan hệ là R và lược đồ quan hệ của nó có các thuộc tính A_1, A_2,..., A_n thì lược đồ quan hệ này có thể viết dưới dạng R(A_1, A_2,..., A_n).

Như vậy, khi ta nói cho một lược đồ quan hệ R(A_1, A_2,..., A_n) có nghĩa là ta đã cho một tập thuộc tính A_1, A_2,..., A_n và trên đó đã tồn tại một quan hệ R.

Ví dụ 2.3: Hình 2.1 cho thấy một quan hệ NHAN_VIEN có các thuộc tính HO_TEN, NAM_SINH, NOI_LAM_VIEC và LUONG là một quan hệ 4 ngôi.

NHAN_VIEN (HO_TEN NAM_SINH NOI_LAM_VIEC LUONG)

T1	Hoàng Văn An	1960	Viện KHVN	425
T2	Hoàng Thu B	1970	Trường ĐHTH	390
T3	Lê Văn Hải	1945	Viện KHVN	425

Hình 2.1. Quan hệ NHAN _VIEN

T1 = (Hoàng Văn An 1960, Viện KHVN, 425) là một bộ của quan hệ NHAN_VIEN. Có thể viết lược đồ quan hệ này là NHAN_VIEN (HO_TEN, NAM_SINH, NOI_LAM_VIEC, LUONG).

II. MỘT PHƯƠNG PHÁP KHÁC ĐỂ THÀNH LẬP QUAN HỆ

Khái niệm toán học hay khái niệm tập danh sách (set of list) của một quan hệ không phải là khái niệm duy nhất đối với CSDL quan hệ. Nếu chúng ta gán tên thuộc tính cho các cột thì thứ tự các cột không còn quan trọng nữa. Vì vậy, có thể xem các bộ như các ánh xạ từ tên các thuộc tính đến các giá trị trong miền của thuộc tính. Theo quan niệm này làm cho một số bảng biểu diễn cùng một quan hệ, trong khi theo định nghĩa toán học thì chúng lại biểu diễn cho những quan hệ khác nhau.

Ví dụ 2.4: Bảng:

MASV	HOTEN	NAMSINH
001	Nguyễn Hải Nam	1980
002	Bùi Như Tuyết	1978
003	Đỗ Trung Tá	1979

và bảng:

NAMSINH	MASV	HOTEN
1980	001	Nguyễn Hải Nam
1978	002	Bùi Như Tuyết
1979	003	Đỗ Trung Tá

Theo quan niệm tập ánh xạ, khi đó qua ánh xạ φ, bộ (001, Nguyễn Hải Nam, 1980) được định nghĩa là:

φ(MASV) = 001, φ(HOTEN) = Nguyễn Hải Nam, φ(NAMSINH) = 1980 nên bộ (001, Nguyễn Hải Nam, 1980) và bộ (1980, Nguyễn Hải Nam, 001) là như nhau. Do đó, hai bảng trên biểu diễn cho cùng một quan hệ. Tuy nhiên theo quan điểm toán học của quan hệ thì hai bộ trên không giống nhau và hai quan hệ trên cũng không được xem là như nhau. Bởi vì các hệ thống CSDL quan hệ hiện tại đều cho phép in ra các cột của một quan hệ theo thứ tự bất

kỳ nên định nghĩa của quan hệ theo quan niệm tập ánh xạ được coi là định nghĩa chuẩn, mặc dù có những trường hợp vẫn dùng định nghĩa tập danh sách cho các quan hệ. Tuy nhiên có một phương pháp đơn giản để chuyển đổi giữa hai cách trình bày này:

- Nếu cho trước một quan hệ theo nghĩa tập danh sách thì chỉ việc đặt các tên tuỳ ý cho các cột, qua đó có thể xem nó như một tập các ánh xạ.

- Ngược lại, nếu cho trước một quan hệ theo nghĩa tập ánh xạ thì giữ cố định các thuộc tính và coi nó như một tập các danh sách.

Quy ước: Nếu t là một bộ và X là một tập các thuộc tính thì ký hiệu t[X] thay cho các thành phần của t trong các thuộc tính của X.

Ví dụ 2.5: t là bộ (001, Nguyễn Hải Nam, 1980) thì t[{MASV, NAM-SINH}] = (001, 1980).

III. BIỂU DIỄN SƠ ĐỒ THỰC THỂ - LIÊN HỆ TRONG MÔ HÌNH QUAN HỆ

Tập các lược đồ quan hệ dùng để biểu diễn thông tin gọi là lược đồ CSDL quan hệ (Relational Database Scheme) và cùng với những giá trị hiện hành của các quan hệ tương ứng tạo ra CSDL quan hệ. Khi thiết kế có thể tự do tạo ra các quan hệ với tập thuộc tính nào đó như một lược đồ quan hệ. Tuy nhiên, có một khuôn mẫu điển hình bằng cách chuyển đổi các sơ đồ thực thể - liên hệ sang các lược đồ CSDL quan hệ. Dữ liệu của sơ đồ thực thể - liên hệ được biểu diễn bởi hai loại quan hệ:

* Một tập thực thể E có thể được biểu diễn bởi một quan hệ mà lược đồ quan hệ của nó chứa tất cả các thuộc tính của tập thực thể đó. Mỗi bộ của quan hệ biểu diễn một thực thể trong thể hiện hiện hành của E.

Ví dụ 2.6: Tập thực thể CUSTOMERS trong ví dụ 1.14, hình 1.3 được biểu diễn bởi quan hệ CUSTOMERS(CNO, CNAME, CADDR, BALANCE).

Nếu E là một tập thực thể có các thực thể được xác định qua mối liên hệ với một tập thực thể F khác thì lược đồ quan hệ của E cũng có những thuộc tính của F cần cho khoá của E.

Ví dụ 2.7: Trong 1.14, trên quan hệ cho tập MANAGERS chỉ có duy nhất một thuộc tính ENO là khoá cho MANAGERS. Giá trị của ENO cho một gian hàng trưởng có số hiệu là số hiệu của thực thể nhân viên giữ chức vụ này.

* Mối liên hệ R giữa các tập thực thể E_1, E_2,..., E_k được biểu diễn bởi một quan hệ có lược đồ quan hệ chứa các thuộc tính trong các khoá của mỗi tập E_1, E_2,..., E_k. Có thể đặt lại tên cho các thuộc tính (nếu cần) để tránh trùng tên. Một bộ t trong quan hệ này biểu diễn cho một danh sách các thực thể e_1, e_2,..., e_k trong đó e_i là một thực thể của tập thực thể E_i. Nghĩa là e_i là một thực thể duy nhất của tập thực thể E_i mà giá trị của thuộc tính khoá của E_i có trong thành phần của bộ t ở những thuộc tính này. Sự có mặt của bộ t trong quan hệ chỉ ra rằng danh sách các thực thể (e_1, e_2,..., e_k) là phần tử hiện hành của mối liên hệ R.

Ví dụ 2.8: Hãy chuyển sơ đồ thực thể - liên hệ của hình 1.3 sang lược đồ CSDL quan hệ. Ở đây thực hiện việc chuyển đổi theo kiểu thủ công (sau này sẽ có một vài bổ sung nhằm đơn giản hoá và tinh chỉnh các lược đồ này). Dưới đây là các lược đồ quan hệ cho các tập thực thể, mỗi lược đồ xuất phát từ một tập thực thể cùng tên với quan hệ.

(1) emps (**ENO**, ENAME, SALARY)

(2) managers (**ENO**)

(3) depts (**DNO**, *DNAME*)

(4) suppliers (**SNO**, SNAME, SADDR)

(5) items (**INO**, *INAME*)

(6) orders (**ONO**, ODATE)

(7) customers (**CNO**, CNAME, CADDR, BALANCE)

Trong các quan hệ trên, các thuộc tính của thực thể được lấy làm các thuộc tính của quan hệ. Trường hợp đặc biệt, quan hệ MANAGERS có thuộc tính duy nhất là khoá vay mượn từ EMPS.

Bây giờ hãy xem xét các mối liên hệ. Không một quan hệ nào được tạo ra cho mối liên hệ isa, bởi vì có chỉ một thuộc tính ENO được lặp lại (và đổi tên khi lặp lại) và lưu các thông tin giống như trong quan hệ MANAGERS, nghĩa là nó chỉ liệt kê số hiệu của tất cả các nhân viên làm gian hàng trưởng. Các mối liên hệ còn lại sinh ra các quan hệ sau:

(8) WORK_IN(ENO, DNO)

(9) MANAGES(ENO, DNO)

(10) CARRIES(INO, DNO)

(11) SUPPLIES(SNO, INO, PRICE)

(12) INCLUDES(ONO, INO, QUANTY)

(13) PLACED_BY(ONO, CNO)

Trong mỗi trường hợp, tập các thuộc tính của quan hệ là tập các khoá của các tập thực thể được nối kết bằng mối liên hệ cùng tên với quan hệ. Chẳng hạn SUPPLIES nối kết SUPPLIERS, ITEMS và PRICE có khoá tương ứng là SNO, INO và PRICE, đó là ba thuộc tính có trong lược đồ quan hệ SUPPLIES tương ứng với mối liên hệ SUPPLIES và vì tên của các thuộc tính khoá không trùng nhau nên không phải đổi tên cho chúng.

Hai quan hệ WORK_IN và MANAGES có cùng tập thuộc tính nhưng ý nghĩa của chúng khác nhau. Cụ thể bộ (e, d) trong WORK_IN có nghĩa e là nhân viên ở gian hàng d, trong khi đó, một bộ tương tự trong MANAGES lại có nghĩa e là trưởng gian hàng d.

Mười ba quan hệ này không phải là thiết kế lý tưởng cho lược đồ CSDL quan hệ ST.

IV. KHOÁ CỦA CÁC QUAN HỆ

Khoá (key) của một quan hệ $R(A_1, A_2,..., A_n)$ là tập con khác rỗng $K \subseteq \{ A_1, A_2,..., A_n \}$, thoả mãn các tính chất sau đây:

1- Với bất kỳ 2 bộ $t_1, t_2 \in R$ đều tồn tại một thuộc tính $A \in K$ sao cho $t_1[A] \neq t_2[A]$. Nói một cách khác, không tồn tại 2 bộ mà có giá trị bằng nhau trên mọi thuộc tính của K. Điều kiện này có thể viết $t_1[K] \neq t_2[K]$. Do vậy, mỗi giá trị của K là xác định duy nhất.

2- Không có tập con thực sự nào của K có tính chất (1)

Ví dụ 2.9:

HANG_HOA (MSMH, TEN_HANG, SO_LUONG)

MSMH	TEN_HANG	SO_LUONG
10101	Sắt phi 6	1000
10102	Sắt phi 8	2000
20001	Xi măng	1000

Quan hệ HANG_HOA

Trong ví dụ trên biểu diễn quan hệ HANG_HOA trong đó mã số mặt hàng (MSMH) là khoá. Mỗi giá trị MSMH đều xác định duy nhất một loại mặt hàng trong quan hệ HANG_HOA.

Một điều quan trọng cần phải nhớ là khoá phụ thuộc vào lược đồ quan hệ không phụ thuộc vào thể hiện của quan hệ.

Cũng nên nhận xét rằng, một quan hệ có thể có nhiều khoá. Chẳng hạn, xét quan hệ DEPTS(DNO, DNAME) chúng ta không muốn đặt cho hai phòng cùng tên và cùng mã nên chúng ta có thể khai báo DNAME là khoá và DNO là một khoá khác. Dĩ nhiên, trong thực tế yêu cầu này phụ thuộc vào quyết định của người thiết kế CSDL. Nếu ta gán khoá cho cả DNAME và DNO thì lược đồ CSDL vật lý phải thiết kế sao cho không thể chứa hai bộ cùng tên phòng hoặc cùng mã phòng. Việc khẳng định khoá được thực hiện bởi người thiết kế CSDL sau khi ta đã xem xét và cân nhắc kỹ lưỡng các dữ liệu và những ràng buộc mà dữ liệu phải tuân theo.

Khi một quan hệ có nhiều khoá, ta nên chọn một khoá xem như là một khoá duy nhất vì rất nhiều cấu trúc lưu trữ vật lý cần có một khoá duy nhất hoặc ít nhất các khoá khác không được hỗ trợ bởi cấu trúc này, khoá đó gọi là khoá chính (Primary key). Tập các khoá có trong quan hệ gọi là các khoá dự kiến (Candidate key).

Khi các quan hệ xuất phát từ một sơ đồ thực thể - liên hệ thì dễ dàng chỉ ra đâu là khoá cho các quan hệ. Nếu các khoá được chọn cho các tập thực thể là khoá nhỏ nhất (nghĩa là không có tập con nào của nó có thể làm khoá) thì ta có thể xác định khoá theo các quy tắc sau:

1- Nếu một quan hệ xuất phát từ một tập thực thể thì khoá của tập thực thể sẽ là khoá của quan hệ.

Ví dụ 2.10: CNO là khoá của tập thực thể CUSTOMERS, khi chuyển sang quan hệ thì CNO là khoá của quan hệ CUSTOMERS, hay SNO là khoá của tập thực thể SUPPLIERS, khi chuyển sang quan hệ thì SNO là khoá của quan hệ SUPPLIERS.

2- Nếu một quan hệ xuất phát từ mối liên hệ nhiều - nhiều thì khoá của quan hệ thường là tập tất cả các thuộc tính của quan hệ.

Ví dụ 2.11: Mối liên hệ SUPPLIES1 giữa các tập thực thể SUPPLIERS có khoá là sno và ITEMS có khoá là INO là mối liên hệ nhiều - nhiều, khi chuyển

sang lược đồ quan hệ sẽ được lược đồ quan hệ SUPPLIES1(SNO, INO) và khoá của nó là tập thuộc tính {SNO, INO}.

3- Nếu một quan hệ xuất phát từ mối liên hệ một - một giữa các tập E và F thì cả khoá cho E và khoá cho F đều làm khoá của quan hệ. Chú ý rằng, các tập thực thể và các quan hệ có thể có nhiều khoá dự kiến.

Ví dụ 2.12:

Quan hệ MANAGES xuất phát từ mối liên hệ MANAGES thuộc loại một - một giữa các tập thực thể MANAGERS và DEPTS trong đó ENO và DNO là khoá tương ứng của MANAGERS và DEPTS nên ENO hoặc DNO sẽ là khoá của quan hệ MANAGES

4- Nếu một quan hệ xuất phát từ mối liên hệ loại nhiều - một từ E_1, E_2,..., E_{K-1} đến E_K thì hợp các khoá của E_1, E_2,..., E_{K-1} thường sẽ làm khoá của quan hệ.

Ví dụ 2.13: Quan hệ SUPPLIES xuất phát từ mối liên hệ SUPPLIES thuộc loại nhiều - một từ các tập thực thể SUPPLIERS và ITEMS đến PRICE trong đó SNO và INO là khoá tương ứng của SUPPLIERS và ITEMS nên tập {SNO, INO} sẽ là khoá của quan hệ SUPPLIES.

Vậy khóa chính của các quan hệ trong ví dụ trên là các thuộc tính in đậm, khoá dự kiến là các thuộc tính in nghiêng được cho dưới đây:

(1) emps (**ENO**, ENAME, SALARY)

(2) managers (**ENO**)

(3) depts (**DNO**, *DNAME*)

(4) suppliers (**SNO**, SNAME, SADDR)

(5) items (**INO**, *INAME)*

(6) orders (**ONO**, ODATE)

(7) customers (**CNO**, CNAME, CADDR, BALANCE)

(8) Work_in (**ENO**, DNO)

(9) Manages (**ENO**, *DNO*)

(10) Carries (**INO**, DNO)

(11) Supplies (**SNO**, INO, PRICE)

(12) Includes (**ONO**, INO, QUANTITY)

(13) Placed_by (**ONO**, CNO)

V. QUAN HỆ CÓ KHOÁ CHUNG

Khi hai quan hệ có chung một khoá dự tuyển ta có thể kết hợp các thuộc tính của hai lược đồ quan hệ này và thay hai quan hệ này bằng một quan hệ có các thuộc tính kết hợp, nhờ đó ta tiết kiệm được bộ nhớ dùng để lưu trữ lặp lại các giá trị khoá trong quan hệ, đồng thời khị được vấn tin về các thuộc tính của hai quan hệ câu trả lời sẽ thực hiện nhanh hơn.

Ví dụ 2.14: Có hai quan hệ (trong ví dụ trên)

DEPTS(DNO, DNAME)

và MANAGES(DNO, ENO)

đều có khoá dự tuyển là DNO, ở quan hệ đầu nó là khoá chính, còn trong quan hệ sau thì không, do đó có thể thay thế DEPTS và MANAGES bằng một quan hệ DEPTS(DNO, DNAME, MGR).

Chú ý rằng, ta quyết định gọi tên quan hệ mới là DEPTS. Các thuộc tính DNO, DNAME vẫn là các thuộc tính cùng tên trong quan hệ DEPTS cũ, trong khi đó ENO trong quan hệ MANAGES được thay bằng MGR trong quan hệ mới. Không có gì sai khi đổi tên các thuộc tính miễn là nó vẫn còn theo đúng ý nghĩa của nó.

Các thể hiện cũ và mới:

DEPTS (DNO DNAME)

12	ti vi
31	tủ lạnh
5	quạt điện

MANAGES (DNO ENO)

12	1
31	17
5	20

DEPTS (DNO DNAME MGR)

12	ti vi	1
31	tủ lạnh	17
5	quạt điện	20

VI. CÁC BỘ KHIẾM KHUYẾT

Khi kết hợp hai hay nhiều quan hệ, có một vấn đề cần phải khắc phục nếu không sẽ cản trở việc kết hợp các quan hệ mặc dù có những lợi ích khi thực hiện. Trong ví dụ trên đã thừa nhận tập số hiệu các gian hàng là như nhau trong hai quan hệ DEPTS và MANAGES. Trong thực tế có thể không phải như vậy. Chẳng hạn siêu thị ST có một gian hàng bán rượu có số hiệu là 16 và tên là rượu nhưng hiện chưa có trưởng gian hàng. Do đó có thể thêm bộ (16, rượu) vào quan hệ DEPTS cũ, nhưng dường như không có cách nào để thêm bộ này vào quan hệ DEPTS mới, vì những bộ như thế đòi hỏi phải có một giá trị nào đó cho thuộc tính MGR.

Những bộ dữ liệu cần phải chia sẻ một giá trị với một bộ trong một quan hệ khác nhưng không tìm được một giá trị nào được gọi là những bộ khiếm khuyết (bộ treo). Có hai giải pháp cho các bộ khiếm khuyết:

- Bổ sung vào lược đồ CSDL những ràng buộc "tồn tại" (tham chiếu toàn vẹn), đó là những điều kiện dưới dạng "nếu một giá trị v xuất hiện trong thuộc tính A của một bộ trong quan hệ R thì v cũng phải xuất hiện trong thuộc tính B của một bộ trong quan hệ S". Chẳng hạn, phải đảm bảo rằng mỗi số hiệu gian hàng xuất hiện trong thuộc tính DNO của quan hệ DEPTS cũ cũng phải xuất hiện trong thuộc tính DNO của quan hệ MANAGES cũ và ngược lại. Khi đó dĩ nhiên là có hạn chế trong các thao tác chèn hoặc xoá dữ liệu.

- Nếu giá trị nào thiếu thì ta lưu giá trị null. Giá trị này có thể là giá trị cho tất cả các thuộc tính ngoại trừ khoá chính và nó nghĩa là giá trị thiếu (không xác định).

- Nếu thừa nhận vấn đề bộ khiếm khuyết được giải quyết bằng các giải pháp trên thì có thể kết hợp các quan hệ khi chúng có chung khoá dự tuyển.

Ví dụ 2.15: Hãy kết hợp các quan hệ trong lược đồ CSDL quan hệ ST.

- Kết hợp (1) và (8) vì có chung khoá dự tuyển ENO được:

EMPS(**ENO**, ENAME, SALARY, DNO)

- Kết hợp (2), (3) và (9) vì có chung khoá dự tuyển được:

DEPTS(**DNO**, DNAME, ENO)

- Kết hợp (5) và (10) vì có chung khoá dự tuyển INO được:

ITEMS(**INO**, INAME, DNO)

- Kết hợp (6) và (13) vì có chung khoá dự tuyển ONO được:

ORDERS(**ONO**, ODATE, CNO)

SUPPLIERS(**SNO**, SNAME, SADDR)

CUSTOMERS(**CNO**, CNAME, CADDR, BALANCE)

SUPPLIES(**SNO, INO**, PRICE)

INCLUDES(**ONO, INO**, QUANTY)

Chú ý: - Khoá chính là các thuộc tính in đậm

- Có thể kết hợp (1) và (2) nhưng trường hợp này có quá nhiều giá trị null.

VII. MỘT SỐ PHÉP TOÁN TRONG MÔ HÌNH QUAN HỆ

Trong mô hình quan hệ có hai hệ ký hiệu khác nhau được sử dụng để biểu diễn các phép toán trên các quan hệ:

- Hệ ký hiệu đại số (Algebraic Notation) được gọi là đại số quan hệ (Relational Algebra), trong đó các câu vấn tin được diễn tả bằng cách áp dụng các phép toán đặc biệt trên các quan hệ.

- Hệ ký hiệu logic (Logical Notation) được gọi là phép tính quan hệ (Relation Caculus), trong đó các câu vấn tin được diễn tả bằng công thức logic sao cho các bộ trong câu trả lời phải thỏa mãn công thức đó.

Trong phần này chỉ đề cập đến đại số quan hệ. Nó gồm một số phép toán như: hợp, hiệu, giao các quan hệ và một số phép toán khác ít quen thuộc hơn, còn hệ ký hiệu logic được dùng trong các hệ thống CSDL tri thức.

1. Các phép toán đại số quan hệ

Gọi R là quan hệ trên tập thuộc tính $\{A_1,..., A_n\}$. Ở đây luôn giả thiết rằng, quan hệ R là tập hữu hạn các bộ.

Toán hạng của đại số quan hệ là một quan hệ và kết quả của các phép toán đại số quan hệ cũng là các quan hệ.

Có năm phép toán cơ bản trong đại số quan hệ, đó là phép hợp, phép trừ, phép tích Descartes, phép chiếu và phép chọn và một số phép toán bổ sung được suy ra từ năm phép toán cơ bản. Các phép toán bổ sung được sử dụng như một dạng viết tắt cho một số tổ hợp của tập các phép toán cơ bản.

Một số phép toán như hợp, hiệu, tích Descartes và giao không phụ thuộc

vào tên của các thuộc tính mà phụ thuộc vào thứ tự của các thuộc tính, nghĩa là chúng là các phép toán trên kiểu danh sách các bộ chứ không phải kiểu tập ánh xạ. Nhưng vẫn có thể áp dụng chúng cho các quan hệ theo quan điểm kiểu ánh xạ (trong hầu hết các DBMS hỗ trợ mô hình quan hệ) bằng cách đặt một thứ tự cố định cho các thuộc tính trước khi thực hiện các phép toán này, rồi xác định lại tên thuộc tính cho quan hệ thu được.

Riêng với các phép hợp, giao và trừ, hai quan hệ tham gia phải khả hợp. Hai quan hệ R và S gọi là khả hợp nếu chúng cùng ngôi.

1.1. Phép hợp

Hợp của hai quan hệ R và S khả hợp là một quan hệ, ký hiệu là $R \cup S$ và là tập tất cả các bộ t sao cho $t \notin R$ hoặc $t \notin S$.

Biểu diễn hình thức phép hợp có dạng:

$$R \cup S = \{t \mid t \in R \text{ hoặc } t \in S\}$$

Ví dụ 2.16:

(a)

R	A	B	C
	a_1	b_1	c_1
	a_1	b_2	c_1
	a_2	b_2	c_2

S	A	B	C
	a_2	b_1	c_2
	a_2	b_2	c_2

$R \cup S$	A	B	C
	a_1	b_1	c_1
	a_1	b_2	c_1
	a_2	b_2	c_2
	a_2	b_1	c_2

(b)

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

S	D	E	F
	b	g	a
	d	a	f

$R \cup S$	A	B	C
	a	b	c
	d	a	f
	c	b	d
	b	g	a

1.2. Phép trừ

Hiệu của hai quan hệ R và S khả hợp là một quan hệ ký hiệu là R - S và là tập tất cả các bộ t sao cho t thuộc R nhưng không thuộc S.

Biểu diễn hình thức phép trừ có dạng:

$$R - S = \{t \mid t \in R \text{ và } t \notin S\}.$$

Ví dụ 2.17:

* Với R, S ở ví dụ 2.16 (a) trên:

R - S (A C D)

a_1	b_1	c_1
a_1	b_2	c_1

* Với R, S ở ví dụ 2.16 (b) trên:

R - S (A C D)

a	b	c
a	b	d

1.3. Tích Descartes

R là quan hệ n - ngôi và S là quan hệ m - ngôi. Tích Descartes của hai quan hệ R và S ký hiệu là R x S là tập tất cả các (n + m) - bộ, với n thành phần đầu là một bộ thuộc R và m thành phần sau là của một bộ thuộc S.

Biểu diễn hình thức có dạng:

R x S = {t | t có dạng (a_1, a_2,..., a_n, b_1, b_2,..., b_m), trong đó (a_1,..., a_n) ∈ R và (b_1,..., b_m) ∈ S}.

Ví dụ 2.18:

* Với R, S ở ví dụ 2.15 (a) trên:

R (A B C)

a_1	b_1	c_1
a_1	b_2	c_1

S (D E F)

d_1	e_1	f_1
d_2	e_2	f_1
d_1	e_2	f_2

R x S (A B C D E F)

a_1	b_1	c_1	d_1	e_1	f_1
a_1	b_1	c_1	d_2	e_2	f_1
a_1	b_1	c_1	d_1	e_2	f_2
a_1	b_2	c_1	d_1	e_1	f_1
a_1	b_2	c_1	d_2	e_2	f_1
a_1	b_2	c_1	d_1	e_2	f_2

R (A B C)

a	b	c
d	a	f
c	b	d

S (A B C)

a	b	c
d	a	f

R x S (A B C D E F)

a	b	c	a	b	c
a	b	c	d	a	f
d	a	f	a	b	c

d	a	f	d	a	f
c	b	d	a	b	c
c	b	d	d	a	f

1.4. Phép chiếu (Projection)

Cho quan hệ R (A_1, A_2,..., A_n), $X \subseteq \{A_1, A_2,..., A_n\}$. Phép chiếu quan hệ R trên tập thuộc tính X là một quan hệ trên tập thuộc tính X, ký hiệu là $\prod_X(R)$ và được biểu diễn hình thức như sau:

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

Trong đó t[X] là giá trị của bộ t trên tập thuộc tính X.

Ví dụ 2.19:

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

R

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_2	b_2	c_2	d_2
a_2	b_2	c_3	d_3

$\prod_X(R)$

A	B
a_1	b_1
a_2	b_2

$\prod_Y(R)$

A	C
a_1	c_1
a_2	c_2
a_2	c_3

1.5. Phép chọn (Selection)

Phép chọn là phép tính để xây dựng một tập con các bộ của quan hệ đã cho, thoả mãn biểu thức F xác định. Biểu thức F được diễn tả bằng một tổ hợp Boolean của các toán hạng, mỗi toán hạng là một phép so sánh đơn giản giữa hai biến là hai thuộc tính hoặc giữa một biến là một thuộc tính và một hằng, cho giá trị "đúng" hoặc "sai" đối với mỗi bộ đã cho khi kiểm tra riêng bộ ấy.

Các phép so sánh trong biểu thức F là <, =, >, > =, < = và ≠; các phép logic là ∧ (và), ∨ (hoặc) và ¬ (không).

Cho quan hệ R(A_1, A_2,..., A_n). Phép chọn quan hệ R với điều kiện F là một quan hệ trên tập thuộc tính (A_1, A_2,..., A_n) ký hiệu là $\delta_F(R)$.

Hình thức hoá phép chọn được định nghĩa như sau:

$\delta_F(R) = \{t \in R \mid F(t) = \text{đúng}\}$.

F(t) được hiểu là giá trị của biểu thức F đối với bộ t.

Ví dụ 2.20: Cho quan hệ sau.

R (A B C D)

a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_2	b_2	c_2	d_2
a_2	b_2	c_2	d_3

Các phép chọn:

$\delta_{A= a1}(R)$ = (A B C D)

a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2

$\delta_{(A=a1)\ V(D=d2)}$ (R) = (A B C D)

a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_2	b_2	c_2	d_2

1.6. Phép giao

Giao của hai quan hệ R và S khả hợp là một quan hệ, ký hiệu là $R \cap S$ và là tập tất cả các bộ t sao cho t thuộc cả R và S.

Biểu diễn hình thức phép giao có dạng sau:

$$R \cap S = \{t \mid t \in R \text{ và } t \in S\}.$$

Ví dụ 2.21:

R (A B C)

a_1	b_1	c_1
a_1	b_2	c_1

S (A B C)

d_1	c_1	f_1
a_1	b_1	c_1
d_1	e_2	f_2

$R \cap S$ (A B C)

a_1	b_1	c_1

Chú ý: Phép giao của 2 quan hệ R và S có thể biểu diễn qua phép trừ:

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

1.7. Phép kết nối (Join)

Gọi θ là một trong các phép so sánh $\{=, >, > =, <, < =, \neq\}$.

Phép kết nối θ của quan hệ R đối với thuộc tính A và quan hệ S đối với thuộc tính B là những bộ t của tích Descartes R x S sao cho t[A] θ t[B]

$$R \rhd\lhd S = \{(t \mid t \in R \times S \text{ và } t[A]\ \theta\ t[B]\}.$$

$$A\ \theta\ B$$

Đặc biệt khi phép sánh θ là phép "=" thì phép kết nối đó gọi là kết nối bằng. Trường hợp kết nối bằng đối với hai thuộc tính cùng tên là A và kết quả thu được loại bỏ đi một trong hai cột R.A hoặc S.A, thì phép kết nối đó gọi là *"kết nối tự nhiên"* và sử dụng ký hiệu "*" thay cho ⊳⊲.

Ví dụ 2.22:

R (A B C)

a_1	1	1
a_2	2	1
a_1	2	2

S (C D E)

1	d_1	e_1
2	d_2	e_2
3	d_3	e_3

$R \rhd\lhd S$ = $B \geqslant C$

A	B	C	F	D	E
a_1	1	1	1	d_1	e_1
a_2	2	1	1	d_1	e_1
a_2	2	1	2	d_2	e_2
a_1	2	2	1	d_1	e_1
a_1	2	2	2	d_2	e_2

Kết quả nối tự nhiên:

R(ABC) * S(CDE) =

A	B	C	D	E
a_1	1	1	d_1	e_1
a_2	2	1	d_1	e_1
a_1	2	2	d_2	e_2

2. Đại số quan hệ - một ngôn ngữ vấn tin

Chúng ta có thể dùng các phép toán của đại số quan hệ để đặt nhiều câu hỏi tự nhiên về các quan hệ.

Ví dụ 2.23: Có ba quan hệ:

S (SNO, SNAME, SEX, STATUS, CITY): Các nhà cung cấp.

P (PNO, PNAME, COLOR, WEIGHT, PRICE): Các mặt hàng.

SP (SNO, PNO, QTY, SDATE): Các mặt hàng đã cung cấp.

Trong đó:

SNO: Số hiệu nhà cung cấp.

SNAME: Tên nhà cung cấp.

SEX: Giới tính.

STATUS: Tình trạng.

CITY: Địa chỉ nhà cung cấp.

PNO: Số hiệu mặt hàng.

PNAME: Tên mặt hàng.

COLOR: Màu sắc mặt hàng.

WEIGHT: Trọng lượng.

PRICE: Giá mặt hàng.

QTY: Số lượng mặt hàng.

SDATE: Ngày nhập hàng.

- Tìm số hiệu của những nhà cung cấp đã cung cấp mặt hàng có số hiệu mặt hàng là P2:

$$\Pi_{SNo}\,(\sigma_{PNo\,=\,'P2'}(SP))$$

- Tìm số hiệu của những nhà cung cấp đã cung cấp ít nhất là một mặt hàng màu đỏ:

$$\Pi_{SNo}\,(\sigma_{COLOR\,=\,'RED'}(P*SP))$$

hoặc

$$\Pi_{SNo}\,((\sigma_{COLOR\,=\,'RED'}(P)\,*SP)$$

Nói chung các phép toán của đại số quan hệ là khá đơn giản, rất mạnh và là một đại số có tính đầy đủ, không cần thủ tục. Tuy nhiên, đây là ngôn ngữ khá gần ngôn ngữ lập trình thủ tục, dễ dàng mở rộng và chủ yếu là làm cơ sở cho việc thiết lập các ngôn ngữ con dữ liệu bậc cao hơn.

Câu hỏi và bài tập chương 2

1. Nêu định nghĩa quan hệ, cho ví dụ.

2. Nêu các phép toán đại số quan hệ, cho ví dụ.

3. Chuyển các sơ đồ thực thể - liên hệ ở các bài tập từ 9 đến 13 (chương 1) sang lược đồ CSDL quan hệ.

4. Cho hai quan hệ R và S như sau:

R	A	B	C	D
	1	0	0	0
	1	1	0	0
	1	1	1	0
	1	1	1	1

S	A	B	C	D
	1	1	1	1
	2	2	1	1
	1	1	1	0

a. Tính $R \cup S$, $R \cap S$, $R - S$ và $S - R$.

b. Giả sử X = {A, B}, Y= {A, C, D}

Tính $\Pi_X(R)$, $\Pi_X(S)$, $\Pi_X(R \cup S)$, $\Pi_Y (R \cap S)$

c. Tính $\delta_{(A=1)\vee (D=0)}(R)$, $\delta_{(A=1)\wedge(D=0)}(R \cup S)$

5. Cho 3 quan hệ:

R	A	B	C	D
	a_1	b_1	c_1	d_1
	a_2	b_1	c_1	d_1
	a_2	b_2	c_2	d_2
	a_2	b_2	c_3	d_2

S	A	B	C	D
	a_1	b_1	c_3	d_1
	a_2	b_1	c_1	d_1
	a_1	b_1	c_1	d_1

U	A	E	F	G
	a_1	e_2	f_1	g_1
	a_2	e_1	f_1	g_1
	a_1	e_2	f_2	g_2
	a_4	e_2	f_2	g_2

Tìm:

a. $R \cup S$, $R \cap S$, $R - S$.

b. $R \times U$, R^*U, $\Pi_{A,B}(R)$, $\delta_{(A=a1) \vee (E= e2)}(U)$.

c. $P_{A,B}\ \delta_{(A=a1) \vee (E= e2)}(R^*U))$.

d. $\delta_{(A=a1) \vee (E= e2)}(P_{A,B}\ (R^*U))$.

6. Cho hai quan hệ R và S:

R	A	B	C	D	E
	0	0	0	0	1
	0	0	1	1	0
	1	1	1	1	1
	0	0	0	1	1

S	D	E
	1	1
	1	0

Tính:

R x S, R*S.

8. Cho 3 quan hệ:

DS (Sbd, Hoten, Ngaysinh, Gioitinh, Quequan)

SBD_PH (Sbd, Sophach)

DTM (Sophach, Diem)

Trong đó:

- Sbd: Số báo danh
- Hoten: Họ và tên sinh viên
- Ngaysinh: Ngày sinh
- Gioitinh: Giới tính
- Quequan: Quê quán
- Sophach: Số phách

- Diem: Điểm thi của môn học

Hãy dùng các phép toán của đại số quan hệ để trả lời các câu hỏi sau:

a. Cho biết họ và tên, ngày sinh, giới tính, quê quán, điểm thi của mỗi sinh viên.

b. Cho biết họ và tên, ngày sinh, giới tính, điểm thi của những sinh viên có điểm thi > 5.

c. Cho biết họ và tên, ngày sinh, điểm thi của những sinh viên có giới tính là "Nu" và quê ở "Hung Yen".

d. Cho biết họ và tên, ngày sinh, điểm thi của những sinh viên có quê ở "Hai Duong" hoặc "Hai Phong" và có điểm thi < 3 hoặc điểm thi > 8.

9. Cho 3 quan hệ:

MatHang(Mamh, Tenmh, Mau, DVT)

KhHang(Makh, Tenkh, Diachi, DT, Gioitinh)

MuaBan(Mamh, Makh, Muaban, NgayMB, Soluong, Dongia)

Trong đó:

- Mamh: Mã mặt hàng
- Tenmh: Tên mặt hàng
- Mau: Màu mặt hàng
- DVT: Đơn vị tính
- Makh: Mã khách hàng
- Tenkh: Tên khách hàng
- Diachi: Địa chỉ khách hàng
- DT: Điện thoại
- Gioitinh: Giới tính
- MuaBan: Mua bán trong đó mua thì ghi là .F., bán thì ghi là .T.
- NgayMB: Ngày mua hoặc bán
- Soluong: Số lượng
- Dongia: Đơn giá

Hãy dùng các phép toán của đại số quan hệ để trả lời các câu hỏi sau:

a. Cho biết makh, tenkh, tenmh của những khách hàng đã bán mặt hàng có mamh = "MH001" hoặc mamh = "MH002".

b. Cho biết makh của những khách hàng đã bán mặt hàng màu đỏ với số lượng >100 trong quý I năm 2003.

c. Cho biết tenkh, diachi, DT của những khách hàng đã bán mặt hàng màu "Vang" hoặc "Xanh" với số lượng >100.

d. Cho biết tenkh, tenmh của những khách hàng có giới tính là "Nam" đã bán mặt hàng màu "Den" và mua mặt hàng màu "Xanh" với 200> soluong>100.

e. Cho biết makh chưa tham gia mua bán lần nào.

Chương 3

NGÔN NGỮ CON DỮ LIỆU SQL

Mục tiêu:

- Học sinh hiểu được các câu lệnh ngôn ngữ con dữ liệu SQL, biết cách sử dụng các câu lệnh ngôn ngữ SQL trong vấn tin.

- Học sinh phải thực hiện các câu lệnh ngôn ngữ SQL trên môi trường Visual Foxpro Microsoft Access.

Nội dung tóm tắt:

Trong chương này sẽ trình bày SQL như một ngôn ngữ đầy đủ, vừa là ngôn ngữ định nghĩa dữ liệu (DDL), vừa là ngôn ngữ thao tác dữ liệu (DML). Các ví dụ trong chương là các ví dụ đã chạy thực tế bằng các câu lệnh SQL chuẩn ISO trên môi trường Visual Foxpro.

NỘI DUNG

Chương này trình bày ngôn ngữ dữ liệu SQL *(Structured Query Language)*. Đây là ngôn ngữ con dữ liệu quan hệ được xác nhận là rất mạnh, phổ dụng và lại dễ sử dụng.

SQL được phát triển từ ngôn ngữ SEQUEL-2, thử nghiệm và cài đặt tại trung tâm nghiên cứu của hãng IBM ở San Jose, California cho hệ thống QTCSDL lớn điển hình là System - R. Trong System - R, SQL vừa đóng vai trò là một ngôn ngữ định nghĩa dữ liệu - DDL vừa là ngôn ngữ thao tác dữ liệu - DML. SQL là một ngôn ngữ phi thủ tục, chuẩn mực và điển hình. Do vậy, hiện nay rất nhiều sản phẩm phần mềm thương mại đều được cài đặt SQL như Oracle, Visual Foxpro, Visual Basic, Access.... Trong tài liệu này sẽ trình bày các khả năng của ngôn ngữ, đồng thời cung cấp cho bạn đọc thêm kinh

nghiệm và cách nhìn các hệ QTCSDL tạm gọi là "kinh điển".

Phép toán cơ bản trong SQL là phép ánh xạ, được miêu tả như một khối select - from - where. Các mệnh đề của ngôn ngữ SQL sẽ được trình bày chi tiết bằng các ví dụ.

Các thuật ngữ trong CSDL quan hệ như quan hệ, thuộc tính, bộ,... được thay thế bằng các thuật ngữ như bảng (table), cột (column), bản ghi (record) hoặc hàng (row) để phù hợp với ý nghĩa của các hệ mềm này.

I. TẠO BẢNG

Mệnh đề tạo bảng có dạng tổng quát như sau:

CREATE TABLE tên_bảng *(tên_cột loại_dữ _liệu [not null],...)*

Trong đó:

- *Tên_bảng, tên_cột:* là xâu ký tự bất kỳ bắt đầu bằng một chữ cái sau đó là chữ cái hoặc chữ số, không chứa ký hiệu trống, không trùng với từ khoá. Trong một bảng tên cột là duy nhất. Thứ tự của cột trong bảng là không quan trọng.

- *Loại_dữ_liệu:* Trong mệnh đề create table dùng một số loại dữ liệu như sau:

+ *Integer:* Số nguyên từ -2 147 483 648 đến 2 147 483 647.

+ *Smallinteger:* Số nguyên từ -32 768 đến 32 767.

+ *Number (n, p):* Số thập phân với độ dài tối đa là n trong đó có p chữ số phần thập phân.

+ *Float:* Số dấu phẩy động.

+ *Char(n):* Xâu ký tự có độ dài tối đa là n, $n \leqslant 255$.

+ *Date:* Dữ liệu dạng ngày tháng.

+ *Logical:* Dữ liệu kiểu logic, nhiều khi dùng từ khoá là Log,

Ví dụ 3.1:

* Cho CSDL ở chương 3 gồm ba bảng S, P và SP. Thiết lập ba bảng đó như sau:

- Tạo bảng S:

```
CREATE TABLE S
(SNO CHAR(5) NOT NULL,
```

SNAME CHAR(8) NOT NULL,
SEX LOG,
STATUS NUMBER (5,0),
CITY CHAR(30) NOT NULL)

Chú ý rằng trong đó NULL là giá trị ngầm định.

- Tạo bảng P:

CREATE TABLE P
(PNO CHAR(5) NOT NULL,
PNAME CHAR(10) NOT NULL,
WEIGHT NUMBER(10,2) NOT NULL,
COLOR CHAR(10) NOT NULL,
CITY CHAR(30) NOT NULL)

- Tạo bảng SP:

CREATE TABLE SP
(SNO CHAR(5) NOT NULL,
PNO CHAR(5) NOT NULL,
QTY NUMBER(5,0),
SDATE DATE NOT NULL,
PRICE NUMBER(8,2)).

* Xoá bảng:

Mệnh đề xoá bảng có dạng tổng quát như sau:

DROP TABLE tên_bảng

Bảng có tên được chỉ ra trong mệnh đề được xoá khỏi CSDL.

* Vào dữ liệu:

Có ba cách biểu diễn mệnh đề vào dữ liệu:

INSERT INTO S(SNo, Sname, Status, City)
VALUES (1, 'Thanh Hương', 30, 'TP Hồ Chí Minh')

Nếu vị trí của các cột trong bảng là cố định mệnh đề có thể viết:

INSERT INTO S

VALUES (1, 'Thanh Hương', 30, 'TP Hồ Chí Minh')

Người vào dữ liệu có thể quên vị trí của các cột, khi đó có thể biểu diễn như sau:

INSERT INTO S(SNo, Status, City, Sname)

VALUES (1, 30, 'TP Hồ Chí Minh', 'Thanh Hương',

Chú ý: Đây là mệnh đề thêm một bản ghi vào một bảng nhưng thường sử dụng để tổ chức vào dữ liệu.

II. KHỐI SELECT

Một trong những cấu trúc tiêu biểu nhất trong ngôn ngữ SQL là khối Select được miêu tả về cú pháp như một khối Select - from - where.

Một cách tổng quát, khối Select bao gồm 3 mệnh đề chính:

Select: Xác định nội dung của các cột cần đưa ra kết quả.

From: Xác định các bảng cần lấy thông tin ra.

Where: Xác định các bản ghi thoả mãn điều kiện chọn lọc để đưa ra kết quả.

Ngoài ra, để mở rộng khả năng của ngôn ngữ, khối Select - from - where còn được bổ sung thêm các mệnh đề *group by, having, order by*, các hàm mẫu và số phần mềm còn thêm cả mệnh đề *compute, for browse*. Trong các phần sau sẽ trình bày chi tiết từng mệnh đề.

Dạng tổng quát của khối lệnh Select được biểu diễn như sau:

SELECT [* | DISTINCT] *danh_sách_cột | biểu thức*

FROM *danh_sách_tên_bảng | tên_các_view*

[WHERE *biểu_thức_điều_kiện*]

[GROUP BY *danh_sách_tên_cột*]

[HAVING *biểu_thức_điều_kiện*]

[ORDER BY {*tên_cột | số_thứ_tự_cột | biểu_thức*}

[ASC/ DESC]]

Trong đó mệnh đề where được biểu diễn dạng:

WHERE [NOT] *biểu_thức phép_sánh biểu_thức*

WHERE [NOT] *tên_cột* [NOT] LIKE *xâu_ký_tự*

WHERE [NOT] *biểu_thức* [NOT] BETWEEN *biểu_thức and biểu_thức*

WHERE [NOT] *biểu_thức* [NOT] IN ({*danh_sách* | *câu_hỏi_con*})

WHERE [NOT] EXISTS *(câu_hỏi_con)*

WHERE [NOT] *biểu_thức phép_sánh* {ANY | ALL *(câu_hỏi_con)*}

WHERE [NOT] *tên_cột phép_kết_nối tên_cột*

WHERE [NOT] *biểu_thức_logic*

WHERE [NOT] *biểu_thức_logic* {AND | OR}[NOT] *biểu_thức_logic*

1. Tìm kiếm theo câu hỏi đơn giản

1.1. Tìm kiếm không điều kiện

Trước hết làm quen với các câu hỏi chỉ liên quan tới một bảng. Trong mệnh đề Select có danh sách chiếu. Danh sách này xác định tên các cột cần có trong bảng kết quả. Nếu xuất hiện giá trị '*' có nghĩa là chọn toàn bộ các cột của bảng.

Ví dụ 3.2: Cho biết tất cả các thông tin về các nhà cung cấp:

```
SELECT *
FROM S
```

Sau khi thực hiện một mệnh đề SQL, để bảng kết quả đúng là một quan hệ (có nghĩa là không có bộ trùng nhau). Trong mệnh đề Select cần thêm từ khoá Distinct.

Ví dụ 3.3: Cho biết số hiệu những mặt hàng đã được cung cấp:

```
SELECT DISTINCT PNO
FROM SP
```

1.2. Tìm kiếm với điều kiện đơn giản

Ví dụ 3.4: Tìm mã số những nhà cung cấp đã cung cấp mặt hàng có số hiệu là P2.

```
SELECT SNO
FROM SP
WHERE PNO = 'P2'
```

Trong SQL các phép sánh được sử dụng bao gồm >, <, > =, < =, = và <>. Các phép tính trên dùng cho mọi loại dữ liệu.

1.3. Tìm kiếm có xử lý xâu ký tự

Xử lý xâu ký tự gần đúng còn gọi là phép tính "thông minh". Trong trường hợp người sử dụng không nhớ rõ tên người hoặc địa danh...

Ví dụ 3.5: Cho biết tất cả các thông tin về các nhà cung cấp có họ là 'Phạm'. Khi đó có thể viết:

```
SELECT *
FROM S
WHERE SNAME LIKE 'Phạm %'
```

Trong SQL sử dụng ký hiệu '%' là thay thế cho một xâu con, dấu gạch dưới ' _' để thay thế cho một ký tự.

A%B: xâu ký tự bất kỳ bắt đầu bằng chữ A và kết thúc bằng chữ B.

%A: xâu ký tự bất kỳ có ký tự kết thúc bằng chữ A.

A_B: xâu gồm 3 ký tự có ký tự thứ 2 là bất kỳ.

A_: xâu gồm 2 ký tự có ký tự đầu là A.

1.4. Xử lý ngày tháng

Ngoài các phép tính thông thường, SQL còn có thể xử lý dữ liệu dạng ngày tháng.

Ví dụ 3.6: Tìm số hiệu mặt hàng, số lượng của những mặt hàng bán trước ngày 24 tháng 4 năm 1994 là 10 ngày.

```
SELECT PNO, QTY
FROM SP
WHERE {04/24/94} - SDATE = 10
```

1.5. Tìm kiếm nhờ sử dụng IN và BETWEEN

Ví dụ 3.7: Tìm những mặt hàng đã cung cấp có giá trị từ 1000 đến 2000.

```
SELECT PNO
FROM SP
WHERE PRICE BETWEEN 1000 AND 2000
```

Ví dụ 3.8: Tìm số hiệu những nhà cung cấp đã cung cấp ít nhất một trong các mặt hàng có số hiệu P1, P2, P3.

```
SELECT DISTINCT SNO
```

FROM SP

WHERE PNO IN ('P1', 'P2', 'P3')

1.6. Các hàm thư viện

Trong SQL có các hàm mẫu gồm:

Count(tên_cột): Đếm số hàng trong bảng dữ liệu nguồn (không đếm những hàng có giá trị NULL trong cột chỉ định). Đặc biệt count(*): Đếm số hàng thoả mãn yêu cầu tìm kiếm mà không cần quan tâm tới bất kỳ một cột nào.

Max(tên_cột | biểu_thức_số): Tìm giá trị lớn nhất của cột kiểu số hoặc biểu thức.

Min(tên_cột | biểu_thức_số): Tìm giá trị nhỏ nhất của cột kiểu số hoặc biểu thức.

Sum(tên_cột | biểu_thức): Tính tổng của cột kiểu số chỉ định hoặc biểu thức.

Avg(tên_cột | biểu_thức_số): Tính giá trị trung bình của cột kiểu số hoặc biểu thức.

Ví dụ 3.9: Cho biết số lần mặt hàng có số hiệu P2 đã được cung cấp:

SELECT COUNT (*)

FROM SP

WHERE PNO = 'P2'

Ví dụ 3.10: Tìm hiệu số số lượng mặt hàng có số hiệu mặt hàng là P1 cung cấp một lần nhiều nhất và một lần ít nhất của nhà cung cấp có số hiệu là S1:

SELECT MAX (QTY) – MIN (QTY)

FROM SP

WHERE SNO = 'S1' AND PNO = 'P1'

Ví dụ 3.11: Cho biết số lượng các mặt hàng khác nhau đã được cung cấp:

SELECT COUNT (DISTINCT PNO)

FROM SP

Chú ý: Biểu_thức_điều_kiện trong mệnh đề Where là một biểu thức Logic và biểu thức này tác động lên từng bản ghi của toàn bảng chỉ ra trong mệnh đề From.

1.7. Tìm kiếm nhờ mệnh đề Group by

Ví dụ 3.12: Tìm mã số những mặt hàng, số lần cung cấp mà các nhà cung cấp đã cung cấp cho khách hàng:

```
SELECT PNO, COUNT(PNO)
FROM SP
GROUP BY PNO
```

Trong mệnh đề này, bảng dữ liệu SP được lấy ra, sau đó phân thành nhóm theo số hiệu mặt hàng (PNO). Sau đó gộp các bản ghi của nhóm thành một bản ghi để đưa ra kết quả.

Chú ý:

- Nếu Group by theo nhiều cột thì giữa các cột phân cách nhau bởi dấu phẩy.
- Tên các cột có mặt trong mệnh đề Group by phải có mặt trong mệnh đề Select.

1.8. Tìm kiếm có sử dụng mệnh đề Having

Mệnh đề Having được sử dụng cùng mệnh đề Group by. Sau Having là biểu thức điều kiện. Biểu thức điều kiện này không tác động vào từng bản ghi của toàn bảng được chỉ ra trong mệnh đề From mà chỉ tác động lần lượt từng nhóm các bản ghi đã chỉ ra tại mệnh đề Group by.

Ví dụ 3.13: Tìm số hiệu những nhà cung cấp đã cung cấp hơn hai mặt hàng:

```
SELECT SNO
FROM SP
GROUP BY SNO
HAVING COUNT (DISTINCT PNO) > 2
```

1.9. Tìm kiếm có sắp xếp

Ví dụ 3.14: Tìm tên các mặt hàng màu đỏ, kết quả đưa ra sắp xếp theo thứ tự giảm dần của mã số mặt hàng:

```
SELECT PNAME, PNO
FROM P
WHERE COLOR = 'Đỏ'
ORDER BY PNO ASC
```

Chú ý:

- Order by dùng để sắp xếp kết quả dữ liệu đầu ra.

- Sau mệnh đề Order by là tên cột rồi đến chiều sắp xếp tăng hoặc giảm (ASC hoặc DESC). Nếu sắp xếp theo nhiều cột thì thứ tự sắp xếp ưu tiên từ trái qua phải, nếu không chỉ ra chiều sắp xếp thì hệ thống ngầm định là ASC.

- Biểu thức sắp xếp sau mệnh đề Order by nếu có mặt trong mệnh đề Select thì có thể dùng thứ tự xuất hiện của nó trong mệnh đề trên thay cho biểu thức sắp xếp.

Ví dụ 3.15: Đưa ra số hiệu mặt hàng, tên mặt hàng có màu đỏ. Danh sách đưa ra sắp xếp theo chiều tăng dần của số hiệu mặt hàng:

```
SELECT PNAME, PNO
FROM P
WHERE COLOR = 'Đỏ'
ORDER BY 2
```

2. Tìm kiếm với câu hỏi phức tạp

Trong phần này trình bày việc tìm kiếm với nhiều bảng qua việc sử dụng ánh xạ lồng nhau hoặc qua phép kết nối.

2.1. Tìm kiếm bằng kết nối

Trong phép nối, các cột tham gia kết nối phải có miền trị là sánh được với nhau. Tên cột của các bảng khác nhau có thể viết tường minh qua tên bảng (Tên_bảng.Tên_cột).

Ví dụ 3.16: Với mỗi mặt hàng đã được cung cấp, cho biết số hiệu của mặt hàng, tên và địa chỉ của hãng đã cung cấp mặt hàng đó:

```
SELECT DISTINCT PNO, SNAME, CITY
FROM SP, S
WHERE SP.SNO = S. SNO
```

Chú ý:

- Trong các phép tìm kiếm có hơn một bảng, nếu tên các cột là không duy nhất thì bắt buộc phải viết tên cột dạng tường minh. Trường hợp không có mệnh đề Where khi đó phép tích Đề - các sẽ được thực hiện.

- Điều kiện trong mệnh đề Where thường là điều kiện kết nối dữ liệu hoặc là điều kiện tìm kiếm dữ liệu.

2.2. Tìm kiếm bằng ánh xạ lồng

Ví dụ 3.17: Tìm tên những nhà cung cấp đã cung cấp mặt hàng có số hiệu P2:

```
SELECT SNAME
FROM S
WHERE SNO IN (SELECT SNO
                    FROM SP
                    WHERE PNO = 'P2')
```

Ví dụ 3.18: Tìm tên các nhà cung cấp không cung ứng mặt hàng có số hiệu P1:

```
SELECT SNAME
FROM S
WHERE 'P1' NOT IN
                    (SELECT PNO
                    FROM SP
                    WHERE SNO = S. SNO)
```

Chú ý: Về đặc tính sử dụng câu hỏi con:

- Phép lồng nhau có thể được lồng nhiều mức.

- Câu hỏi con phải đặt trong hai dấu ngoặc đơn trong mệnh đề Where hoặc Having và đi sau toán tử so sánh dữ liệu.

- Nội dung chỉ định trong mệnh đề Select của câu hỏi con chỉ là một tên (Một cột hoặc một biểu thức).

- Kết quả của câu hỏi con phải có kiểu tương thích với Query mẹ. Trong Query con không được sử dụng các phép toán so sánh Like, Between. Kết quả chọn lọc dữ liệu trong Query con chỉ dùng cho việc so sánh của Query mẹ mà không được đưa ra bảng kết quả, vì vậy trong Query con không được sử dụng mệnh đề Order by.

- Người ta thường dùng Query con khi giá trị để so sánh cho việc tìm kiếm trong Query mẹ là chưa tường minh.

Ví dụ 3.19: Cho biết số hiệu nhà cung cấp, số hiệu mặt hàng của các mặt hàng đã được cung cấp với số lượng lớn nhất:

```
SELECT SNO, PNO
FROM SP
WHERE QTY = (SELECT MAX(QTY) FROM SP)
```

Ví dụ 3.20: Cho biết số hiệu nhà cung cấp, tên nhà cung cấp chưa cung cấp mặt hàng nào:

```
SELECT SNO, SNAME
FROM S
WHERE SNO NOT IN
                (SELECT SNO
                 FROM SP)
```

2.3. Tìm kiếm với từ khoá [NOT] EXISTS

Ví dụ 3.21: Đưa ra tất cả các thông tin về các nhà cung cấp đã cung cấp ít nhất một mặt hàng nào đó:

```
SELECT *
FROM S
WHERE EXISTS
        (SELECT SNO
         FROM SP, S
         WHERE SP.SNO = S. SNO)
```

Ví dụ 3.22: Đưa ra tất cả các thông tin về các nhà cung cấp chưa cung cấp bất cứ một mặt hàng nào:

```
SELECT *
FROM S
WHERE NOT EXISTS
          (SELECT SNO
           FROM SP, S
           WHERE SP.SNO = S. SNO)
```

2.4. Tìm kiếm có sử dụng lượng từ ANY và ALL

Ví dụ 3.23: Đưa ra tên mặt hàng, màu, trọng lượng của những mặt hàng đã được cung cấp bởi nhà cung cấp có số hiệu S1:

```
SELECT PNAME, COLOR, WEIGHT
FROM P
WHERE PNO = ANY(SELECT PNO
                FROM SP
                WHERE SNO = 'S1')
```

Ví dụ 3.24: Tìm số hiệu nhà cung cấp đã cung cấp một mặt hàng nào đó với số lượng lớn hơn hoặc bằng số lượng mỗi lần cung cấp một mặt hàng của các nhà cung cấp:

```
SELECT SNO
FROM SP
WHERE QTY >= ALL
             (SELECT QTY
             FROM SP)
```

Mệnh đề trên hoàn toàn tương đương với:

```
SELECT SNO
FROM SP
WHERE QTY = (SELECT MAX (QTY)
FROM SP)
```

Chú ý: Trong nhiều trường hợp có thể thay thế giữa các mệnh đề 'exists' với 'in' hoặc 'not in', 'equal any' tương đương với 'not equal all' v.v.

Ví dụ 3.25: Ví dụ 3.22 có thể tương đương với:

```
SELECT *
FROM S
WHERE SNO NOT IN
          (SELECT SNO
          FROM SP)
```

Hoặc:

```
SELECT *
FROM S
WHERE SNO <> ALL
```

(SELECT SNO
FROM SP)

2.5. Tìm kiếm có chứa phép tính tập hợp

Ví dụ 3.26: Tìm số hiệu những nhà cung cấp hiện thời chưa cung cấp một mặt hàng nào cả:

```
SELECT SNO
FROM S
MINUS
(SELECT SNO
FROM SP)
```

Ví dụ 3.27: Tìm số hiệu các mặt hàng đã được cung cấp bởi các nhà cung cấp có số hiệu là S1 hoặc S2:

```
(SELECT PNO
FROM SP
WHERE SNO = 'S1')
UNION
(SELECT PNO
FROM SP
WHERE SNO = 'S2')
```

III. CÁC MỆNH ĐỀ CẬP NHẬT DỮ LIỆU

1. Thêm một bộ

Mệnh đề thêm một bộ có dạng tổng quát:

INSERT INTO *tên_bảng (danh_sách_tên_cột)*
VALUE *(các_giá_trị)*
[câu hỏi con]

Ví dụ 3.28: Có thể bổ sung một tập các bản ghi là kết quả xử lý của một câu hỏi nào đó, chẳng hạn:

```
INSERT INTO P
SELECT *
```

FROM W

WHERE COLOR = 'Đỏ'

Nếu như bảng W và bảng P có cùng lược đồ.

2. Xoá bản ghi

Mệnh đề xoá bản ghi có thể được thực hiện cho một hoặc nhiều bản ghi thoả mãn một điều kiện nào đó. Dạng tổng quát là:

DELETE

[FROM *{tên_bảng / tên_view }*]

[WHERE *biểu_thức_điều_kiện*]

Ví dụ 3.29: Loại bỏ nhà cung cấp có số hiệu S1 khỏi bảng S:

DELETE FROM S WHERE SNO = 'S1'

Ví dụ 3.30: Loại bỏ các mặt hàng đã được cung cấp sau ngày 20 tháng 5 năm 1994:

DELETE FROM SP WHERE SDATE > {05/20/94}

Ví dụ 3.31: Loại bỏ những nhà cung cấp chưa cung cấp mặt hàng nào:

DELETE FROM S WHERE SNO NOT IN

(SELECT SNO FROM SP)

3. Sửa đổi dữ liệu

Mệnh đề sửa đổi các giá trị của các bản ghi trong bảng của CSDL theo một điều kiện nào đó có dạng tổng quát là:

UPDATE [*tên_bảng*]

SET [*tên_cột = biểu thức,...*]

[FROM *tên_bảng*]

[WHERE *biểu_thức_điều_kiện*]

Ví dụ 3.32: Đổi màu các mặt hàng có số hiệu P2 thành màu vàng:

UPDATE P SET COLOR = 'Vàng' WHERE PNO = 'P2'

4. Tạo tệp chỉ số

Trong SQL không có cơ chế tự động tạo tệp chỉ số cho các cột của bảng. Việc tạo chỉ số là do người sử dụng tự chọn. Mệnh đề tổng quát có dạng:

CREATE [UNIQUE] INDEX *tên_chỉ_số ON tên_bảng (tên_cột* [ASC | DESC])

Chú ý: Ngầm định của hệ thống là ASC.

Bỏ chỉ số thì sử dụng mệnh đề:

DROP INDEX *tên_chỉ_số*

Ví dụ 3.33: Tạo tệp chỉ số I3 theo cột City của bảng S:

CREATE INDEX I3 ON S(CITY)

Trong SQL có thể tổ chức đa chỉ số, tức là tổ chức một tệp chỉ số cho nhiều cột, mỗi cột có thể có chiều tăng giảm khác nhau, thứ tự được tính từ trái qua phải.

Ví dụ 3.34: Tạo tệp chỉ số 14 cho bảng SP theo cột SNO tăng dần và cột PNO giảm dần:

CREATE INDEX 14 ON SP (SNO ASC, PNO DESC)

5. Thêm cột mới

Thêm cột DONGIA (đơn giá) cho bảng với kiểu số liệu dạng số thập phân:

ALTER TABLE SP ADD DONGIA DECIMAL(8.2)

6. Xoá cột

Xoá cột DONGIA (đơn giá) của bảng SP:

DROP COLUMN DONGIA FROM SP

IV. TẠO VIEW CỦA NGƯỜI SỬ DỤNG

Dạng tổng quát:

CREATE VIEW *tên_view (danh_sách_tên_cột)*

AS *mệnh_đề_select*

Ví dụ 3.35: Tạo view PP gồm các cột PNO, PNAME từ bảng P của các mặt hàng màu đỏ.

```
CREATE VIEW PP (PNO, PNAME) AS
SELECT PNO, PNAME
FROM P WHERE COLOR = 'Đỏ'
```

Tên các cột của view và tên các cột của bảng chỉ ra trong mệnh đề Select

có thể khác nhau, tên cột của view là do người sử dụng tự đặt tương ứng với từng cột của mệnh đề Select.

- Tạo view TINHTOAN gồm các cột PNO, SNO, SLMOI từ bảng SP với số lượng mỗi mặt hàng đều được tăng thêm là 20:

```
CREATE VIEW TINHTOAN (SNO, PNO, SLMOI)
AS (SELECT  SNO, PNO, QTY + 20
FROM SP)
```

V. CÁC MỆNH ĐỀ VỀ AN TOÀN DỮ LIỆU

Trong ngôn ngữ SQL cho phép người sử dụng kiểm tra dữ liệu của mình khi cập nhật và tuyên bố quyền truy nhập dữ liệu cần thiết để đảm bảo cho tính nhất quán dữ liệu. Đặc biệt trong trường hợp nhiều người cùng truy nhập tới một bảng, một bản ghi của CSDL, hệ thống cần được đảm bảo an toàn.

1. Trao quyền truy nhập

GRANT tên_quyền_truy_nhập ON đối_tượng

TO tên_người sử dụng [WITH GRANT OPTION]

Các quyền truy nhập trong SQL bao gồm: *read (đọc), select (chọn), write (ghi), insert (bổ sung), update (sửa đổi), delete (xoá) và run (thực hiện).*

- *Đối_tượng:* là tên bảng, tên view hoặc tên chương trình nào đó.

- *Tên_người_sử_dụng:* tên một người, một nhóm hoặc một danh sách hoặc sử dụng từ khoá public, world cho mọi người sử dụng.

- Từ khoá *with grant option* đảm bảo để người sử dụng có thể tiếp tục trao quyền sử dụng cho người khác nữa.

Ví dụ 3.36: Trao quyền đọc bảng S cho cô Hồng:

```
GRANT READ ON S TO Hong WITH GRANT OPTION
```

Trong mệnh đề này, cô Hồng có thể trao quyền đọc bảng S cho người khác.

2. Hủy bỏ quyền truy nhập

Rút quyền đọc bảng S của cô Hồng:

```
REVOKE READ ON S FROM Hong
```

Câu hỏi và bài tập chương 3

1. Nêu các câu lệnh định nghĩa dữ liệu trong ngôn ngữ SQL.

2. Nêu các câu lệnh thao tác dữ liệu trong ngôn ngữ SQL.

3. Nêu các câu lệnh an toàn dữ liệu trong ngôn ngữ SQL.

4. Cho CSDL có 3 bảng:

DS(SBD, HOTEN, NGAYSINH, GIOITINH, QUEQUAN)

SBD_PH(SBD,SOPHACH)

DTM(SOPHACH, DIEM)

Trong đó:

- SBD: Số báo danh
- HOTEN: Họ và tên sinh viên
- NGAYSINH: Ngày sinh
- GIOITINH: Giới tính
- QUEQUAN: Quê quán
- SOPHACH: Số phách
- DIEM: Điểm thi của môn học

Hãy dùng các câu lệnh của SQL thực hiện các công việc sau:

a. Nêu các câu lệnh tạo 3 bảng trên và chọn kiểu dữ liệu thích hợp cho các cột.

b. Sử dụng câu lệnh vào dữ liệu, nhập vào 3 bảng trên một số bản ghi.

c. Xoá sinh viên có số báo danh 123.

d. Cho biết họ và tên, ngày sinh, giới tính, quê quán, điểm thi của mỗi sinh viên.

e. Cho biết họ và tên, ngày sinh, giới tính, quê quán của các sinh viên có họ Phạm.

f. Cho biết họ và tên, ngày sinh, giới tính, quê quán của các sinh viên có tên là Thương.

g. Cho biết họ và tên, ngày sinh, giới tính, điểm thi của những sinh viên có điểm thi 5.

h. Cho biết họ và tên, ngày sinh, điểm thi của những sinh viên có giới tính là "Nu" và quê ở "Hai Duong". Danh sách đưa ra sắp xếp theo chiều giảm dần của điểm.

k. Cho biết họ và tên, ngày sinh, điểm thi của những sinh viên có quê ở "Hai Phong" hoặc "Hung Yen" và có điểm thi < 3 hoặc điểm thi > 8. Danh sách đưa ra sắp xếp theo chiều giảm dần của điểm. Danh sách đưa ra sắp xếp theo chiều tăng dần của điểm.

l. Cho biết họ và tên, ngày sinh, giới tính, quê quán của các sinh viên không đi thi.

m. Đưa ra số lượng sinh viên, điểm trung bình của các sinh viên mỗi tỉnh.

n. Đưa ra tổng số lượng sinh viên nữ, điểm trung bình của các sinh viên nữ mỗi tỉnh. Danh sách đưa ra sắp xếp theo giảm dần của điểm trung bình.

o. Đưa ra tổng số, điểm trung bình của các sinh viên nữ mỗi tỉnh có điểm trung bình > 7. Danh sách đưa ra sắp xếp theo chiều giảm dần của số lượng sinh viên, nếu số lượng bằng nhau thì sắp xếp theo chiều tăng dần của điểm trung bình.

p. Tăng điểm thi thêm 2 điểm cho các học sinh quê ở Sơn La.

5. Cho CSDL có 3 bảng:

MATHANG(Mamh, Tenmh, Mau, DVT)

KHHANG(Makh, Tenkh, Diachi, DT, Gioitinh)

MUABAN(Mamh, Makh, Muaban, NgayMB, Soluong, Dongia)

Trong đó:

- Mamh: Mã mặt hàng
- Tenmh: Tên mặt hàng
- Mau: Màu mặt hàng
- DVT: Đơn vị tính
- Makh: Mã khách hàng
- Tenkh: Tên khách hàng
- Diachi: Địa chỉ khách hàng
- DT: Điện thoại
- Gioitinh: Giới tính
- MuaBan: Mua bán, trong đó mua thì ghi là .F., bán thì ghi là .T.
- NgayMB: Ngày mua hoặc bán
- Soluong: Số lượng
- Dongia: Đơn giá

Hãy dùng các câu lệnh của SQL thực hiện các công việc sau:

a. Nêu các câu lệnh tạo 3 bảng trên và chọn kiểu dữ liệu thích hợp cho các cột.

b. Sử dụng câu lệnh vào dữ liệu, nhập vào 3 bảng trên mỗi bảng một số bản ghi.

c. Cho biết makh của các khách hàng đã bán hàng.

d. Cho biết makh, tenkh, tenmh của những khách hàng đã bán mặt hàng có mamh = "MH001" hoặc mamh = "MH002".

e. Cho biết thông tin của các mặt hàng đã mua có đơn giá nhỏ hơn đơn giá trung bình mỗi lần mua một mặt hàng.

f. Cho biết makh của những khách hàng đã bán mặt hàng màu đỏ với số lượng >100 trong quý I năm 2003.

g. Cho biết tenkh, diachi, DT của những khách hàng đã bán mặt hàng màu "Vang" hoặc "Xanh" với số lượng >100.

h. Cho biết tenkh, tenmh của những khách hàng có giới tính là "Nam" đã bán mặt hàng màu "Den" và mua mặt hàng màu "Xanh" với 200 > soluong >100.

k. Cho biết các thông tin của các khách chưa tham gia mua bán lần nào.

l. Cho biết các thông tin của các khách chưa mua hàng lần nào.

m. Cho biết các thông tin của các mặt hàng chưa được mua bán lần nào.

n. Cho biết các thông tin của các khách hàng, số lần mua và số tiền đã mua hàng có số lần mua lớn nhất và nhỏ nhất trong tháng 1 năm 1999.

o. Cho biết các thông tin của các khách đã mua hàng có số lần mua lớn hơn 3. Danh sách đưa ra sắp xếp theo chiều giảm dần của số lần mua.

p. Cho biết mamh, tenmh, số lượng tồn kho của mỗi mặt hàng.

q. Xoá đi mặt hàng có mamh là 001.

s. Thêm cột thanhtien và tính số tiền mỗi lần mua bán một mặt hàng.

Chương 4

LÝ THUYẾT THIẾT KẾ CƠ SỞ DỮ LIỆU QUAN HỆ

Mục tiêu:

- Học sinh hiểu được các khái niệm, kiến thức về lý thuyết phụ thuộc hàm, nắm vững khái niệm về khoá, các dạng chuẩn của lược đồ CSDL quan hệ.

- Học sinh biết vận dụng kiến thức tìm khoá và chuẩn hoá các lược đồ CSDL quan hệ.

Nội dung tóm tắt:

Trọng tâm của chương này là các kiến thức về lý thuyết phụ thuộc hàm, phép tách lược đồ quan hệ, áp dụng các kiến thức vào việc tìm khoá và chuẩn hoá các lược đồ CSDL quan hệ.

NỘI DUNG

Khi thiết kế một CSDL quan hệ thường đòi hỏi phải chọn các lược đồ quan hệ. Việc chọn tập các lược đồ này có thể tốt hơn hoặc xấu hơn tập các lược đồ khác dựa trên một số tiêu chuẩn nào đó. Do vậy, cần thiết phải nghiên cứu các tính chất cơ bản cũng như các thuật toán để có thể nhận được những tập lược đồ phù hợp. Trọng tâm của việc thiết kế các lược đồ CSDL là các phụ thuộc dữ liệu, tức là các mối ràng buộc có thể giữa các giá trị hiện hữu của các lược đồ. Chẳng hạn thuộc tính này xác định duy nhất thuộc tính kia như số hiệu nhà cung cấp SNO xác định tên nhà cung cấp SNAME ở lược đồ quan hệ SUPPLIERS (trong CSDL ST) và nói rằng có một "phụ thuộc hàm" của SNAME vào SNO, v.v.

Vậy làm thế nào để thiết kế một CSDL cho tốt? Có thể khảo sát ví dụ lược đồ quan hệ cung cấp như sau:

S(NAME, ADD, PRO, PRICE)

Gồm tên nhà cung cấp, địa chỉ, mặt hàng và giá. Xem xét các vấn đề nảy sinh như sau:

* *Dư thừa dữ liệu (Redundancy):*

Dễ dàng thấy rằng, mỗi khi xuất hiện tên nhà cung cấp thì địa chỉ của ông ta lại lặp lại trong quan hệ.

* *Không nhất quán (Inconsistency)* (Dị thường xuất hiện khi sửa dữ liệu):

Là hệ quả của việc dư thừa dữ liệu. Ví dụ, khi sửa đổi địa chỉ của một nhà cung cấp ở một bộ nào đó còn các bộ khác vẫn giữ nguyên. Khi đó xảy ra một nhà cung cấp lại không có địa chỉ duy nhất.

* *Dị thường khi thêm bộ (Insertion anomalies):*

Một nhà cung cấp chưa cung cấp một mặt hàng (PRO) nào cả, khi đó không thể đưa địa chỉ, tên nhà cung cấp là một bản ghi (bộ) vào quan hệ vì rằng sẽ phải đưa giá trị nào vào vị trí của thuộc tính PRO và PRICE.

* *Dị thường khi xoá bộ (Deletion anomalies):*

Là vấn đề ngược lại của vấn đề dị thường khi thêm bộ, không thể xoá tất cả các mặt hàng được cung ứng bởi một nhà cung cấp, vì sẽ làm mất dấu vết tìm ra địa chỉ của nhà cung cấp này.

Do vậy, quan hệ S nêu trên có thể được phân chia thành những quan hệ khác nhau nhằm tránh tất cả những điều đã nêu để đạt được một lược đồ CSDL (tập các lược đồ quan hệ) sao cho tốt hơn.

I. PHỤ THUỘC HÀM

* Quy ước về ký hiệu:

- Các chữ hoa ở đầu bộ các chữ cái in hoa như A, B, C,... biểu thị một thuộc tính đơn.

- Các chữ hoa ở cuối bộ các chữ cái in hoa như U, V,..., X, Y, Z biểu thị một tập thuộc tính, có thể là tập chỉ có một thuộc tính.

- R được dùng để biểu thị một lược đồ quan hệ. Cũng có thể đặt tên các quan hệ bằng lược đồ của chúng. Chẳng hạn, một quan hệ có các thuộc tính A, B, C có thể được viết là ABC hoặc R(ABC).

- Sử dụng r cho một quan hệ, là thể hiện của lược đồ quan hệ R.

- Ký hiệu nối kết chuỗi được dùng biểu thị cho phép hợp. Do đó A_1, A_2... A_n được dùng để biểu diễn tập các thuộc tính $\{A_1, A_2, .., A_n\}$ và XY viết tắt của $X \cup Y$. Trường hợp XA hay AX cũng được viết thay cho $X \cup \{A\}$ hay $\{A\} \cup X$, với X là tập các thuộc tính và A là một thuộc tính đơn.

1. Khái niệm phụ thuộc hàm

Khái niệm về phụ thuộc hàm (trong một quan hệ) *(Functional dependendies)* là một quan niệm có tầm quan trọng hết sức lớn đối với việc thiết kế lược đồ CSDL. Ở đây sẽ trình bày các khái niệm một cách hình thức.

Định nghĩa 4.1

Cho R là một lược đồ quan hệ với $U = \{A_1, A_2, ..., A_n\}$ là tập thuộc tính. X và Y là tập con của U.

Nói rằng $X \rightarrow Y$ (đọc là X xác định hàm Y hoặc Y phụ thuộc hàm vào X) nếu với bất kỳ quan hệ r nào đó là giá trị hiện hành của R và bất kỳ hai bộ t_1, $t_2 \in r$ mà $t_1[X] = t_2[X]$ thì $t_1[Y] = t_2[Y]$.

Phụ thuộc hàm ký hiệu là FD, cần chú ý rằng chỉ xét các phụ thuộc hàm thoả mãn cho mọi quan hệ trên lược đồ quan hệ tương ứng của nó. Không thể xem xét một phụ thuộc hàm thoả mãn một quan hệ r đặc biệt (ví dụ quan hệ rỗng) của lược đồ R rồi sau đó quy nạp rằng phụ thuộc hàm đó là thoả mãn trên R.

Ví dụ 4.1: Trong lược đồ CSDL ST (Hình 1.3)

* Một số phụ thuộc hàm cơ bản, nhất là những phụ thuộc hàm khẳng định rằng một khoá xác định được tất cả các thuộc tính của lược đồ quan hệ. Chẳng hạn trong SUPPLIERS có:

SNO → SNAME, SNO → SADDR.

Hay trong SUPPLIES có:

SNO INO → PRICE

Còn trong CUSTOMERS có:

CNO → CNAME CADDR BALANCE

* Một số phụ thuộc hàm tầm thường như:

SNAME → SNAME SADDR → SADDR SNAME SADDR → SADDR

* Một số phụ thuộc hàm khác:

SNO INO → SADDR SNAME INAME

2. Hệ tiên đề cho phụ thuộc hàm

Gọi F là tập tất cả các phụ thuộc hàm đối với lược đồ quan hệ R và $X \rightarrow Y$ là một phụ thuộc hàm với $X, Y \subseteq U$. Nói rằng $X \rightarrow Y$ được suy diễn logic từ F nếu mỗi quan hệ r thoả mãn các phụ thuộc hàm của F thì cũng thoả $X \rightarrow Y$. Chẳng hạn $F = \{A \rightarrow B, B \rightarrow C\}$ thì $A \rightarrow C$ được suy ra từ F.

* Gọi F^+ là bao đóng (closure) của F, tức là tất cả các phụ thuộc hàm được suy diễn logic từ F. Nếu $F = F^+$ thì F là họ đầy đủ (full family) của các phụ thuộc hàm.

* Để có thể xác định khoá của một lược đồ quan hệ và hiểu được các phép suy diễn logic cho các phụ thuộc hàm cần tính được F^+ từ F, hoặc ít nhất phải khẳng định được $X \rightarrow Y$ có thuộc F^+ hay không nếu biết phụ thuộc hàm $X \rightarrow Y$ và tập phụ thuộc hàm F. Do đó, đòi hỏi phải có những quy tắc suy diễn cho biết làm sao có thể suy ra một hay nhiều phụ thuộc hàm từ các phụ thuộc hàm khác. Tập các quy tắc này được được Armstrong đưa ra năm 1974 và được gọi là hệ tiên đề Armstrong. Cho R là lược đồ quan hệ với $U = \{A_1,..., A_n\}$ là tập các thuộc tính của nó và $X, Y, Z, W \subseteq U$.

2.1. Hệ tiên đề Armstrong

Bao gồm:

- A1 (phản xạ): Nếu $Y \subseteq X$ thì $X \rightarrow Y$

Quy tắc này đưa ra những phụ thuộc hàm tầm thường là những phụ thuộc hàm mà vế trái chứa vế phải. Sử dụng quy tắc này chỉ phụ thuộc vào U, không phải vào F.

- A2 (tăng trưởng): Nếu $Z \subseteq U$ và $X \rightarrow Y$ thì $XZ \rightarrow YZ$.

Trong đó ký hiệu XZ thay cho ký hiệu $X \cup Z$ và $X \rightarrow Y$ có thể thuộc F hoặc được suy diễn logic từ F.

- A3 (bắc cầu): Nếu $X \rightarrow Y$ và $Y \rightarrow Z$ thì $X \rightarrow Z$.

Ví dụ 4.2:

Cho lược đồ quan hệ ABCD với các phụ thuộc hàm $A \rightarrow C$ và $B \rightarrow D$

Chứng minh rằng $AB \rightarrow ABCD$.

Thật vậy, ta có:

$A \rightarrow C$ (giả thiết)

$\Rightarrow AB \rightarrow ABC$ (luật tăng trưởng)

và B → D (giả thiết)

⇒ ABC → ABCD (luật tăng trưởng)

Vậy AB → ABCD (luật bắc cầu)

2.2. Tính đúng đắn của hệ tiên đề Armstrong

Bổ đề 4.1

Hệ tiên đề Armstrong là đúng. Có nghĩa là F là tập các phụ thuộc hàm đúng trên quan hệ R. Nếu X→ Y là một phụ thuộc hàm được suy dẫn từ F nhờ hệ tiên đề Armstrong thì X → Y là đúng trên quan hệ R.

Chứng minh: Lần lượt kiểm tra tính đúng đắn của ba bổ đề A_1, A_2, A_3.

- A_1: Tiên đề A_1 rõ ràng là đúng vì không thể có hai bộ bằng nhau trên X mà lại không bằng nhau trên tập con của nó.

- A_2: Giả sử rằng quan hệ r thoả X → Y và tồn tại hai bộ t, u ∈ r sao cho t[XZ] = u[XZ} mà t[YZ} ≠ u[YZ}. Vì rằng t[Z] = u[Z] nên để t[YZ} ≠ u[YZ thì t[Y] ≠ u[Y]. Nhưng vì t[X] = u{X] nên t{Y] ≠ u{Y] là trái với giả thiết X → Y. Vậy t[YZ} = u{YZ}.

- A_3: Cho X → Y và Y → Z đúng trên quan hệ r:

Giả sử tồn tại hai bộ t và u ∈ r sao cho t{X} = u{X} và t[Z} ≠ u[Z].

Từ X → Z suy ra vì t{X} = u[X] nên t[Y] = u[Y]

Nhưng lại có t[Y] = u[Y] và t[Z] ≠ u[Z] là trái với giả thiết là Y→ Z. Vậy t[Z] = u[Z].

Suy ra X → Z đúng trên quan hệ r.

Từ hệ tiên đề Armstrong suy ra một số luật sau đây:

2.3. Các quy tắc suy diễn bổ sung

Bổ đề 4.2

a. Nếu X → Y và X → Z thì X → YZ

b. Luật tựa bắc cầu: Nếu X → Y và WY → Z thì XW → Z

c. Luật tách: Nếu X → Y và Z ⊆ Y thì X → Z

Chứng minh:

- Từ X → Y dùng luật tăng trưởng, thêm X có X → XY.

Từ X → Z dùng luật tăng trưởng thêm Y có XY → YZ và cuối cùng

dùng luật bắc cầu suy ra vì $X \rightarrow XY$ và $XY \rightarrow XZ$ nên $X \rightarrow YZ$.

- Từ $X \rightarrow Y$, dùng luật tăng trưởng, thêm W có $WX \rightarrow WY$. Dùng luật bắc cầu cho $WX \rightarrow WY$ và $WY \rightarrow Z$ suy ra $WX \rightarrow Z$.

- Vì $Z \subseteq Y$ nên $Y \rightarrow Z$ (theo luật phản xạ).

Dùng luật bắc cầu cho $X \rightarrow Y$ và $Y \rightarrow Z$ có $X \rightarrow Z$.

Một hệ quả quan trọng của luật tách và luật hợp là nếu $X \rightarrow Y$ suy ra $X \rightarrow Ai$ với mọi $Ai \in Y$.

2.4. Bao đóng của tập thuộc tính

Để dễ dàng cho việc chứng minh tính đầy đủ của hệ tiên đề Armstrong, ở đây đưa thêm khái niệm bao đóng (closure) của tập các thuộc tính đối với tập các phụ thuộc hàm. Gọi F là tập các phụ thuộc hàm trên tập thuộc tính U, $X \subseteq U$. X^+ là bao đóng của X (đối với F) được định nghĩa như sau:

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

Nói cụ thể: X^+ là tập tất cả các thuộc tính A mà phụ thuộc hàm $X \rightarrow A$ có thể được suy diễn logic từ F nhờ hệ tiên đề Armstrong.

Bổ đề 4.3

$X \rightarrow Y$ suy dẫn từ hệ tiên đề Armstrong khi và chỉ khi $Y \subseteq X^+$.

Chứng minh:

- Đảo: Giả sử $Y = A_1 \ldots A_n$ với $A_1, \ldots, A_n$ là các thuộc tính và $Y \subseteq X^+$ suy ra $A_i \in X^+$.

Từ định nghĩa X^+ có $X \rightarrow A_i$. Áp dụng tiên đề Armstrong cho mỗi $A_i \in Y$, luật hợp suy ra $X \rightarrow Y$.

Thuận: Giả sử có $X \rightarrow Y$, áp dụng hệ tiên đề Armstrong cho mỗi i có $X \rightarrow A_i$, $A_i \in Y$ nhờ luật tách, suy ra $A_i \in X^+$. Từ đó suy ra $Y \subseteq X^+$.

Suy ra: Để chứng minh $X \rightarrow Y$, ngoài cách dùng định nghĩa phụ thuộc hàm, áp dụng hệ tiên đề Armstrong và các quy tắc bổ sung còn có thể áp dụng bổ đề trên bằng cách:

- Tính X^+.

- Khẳng định $Y \subseteq X^+$.

Ví dụ 4.3: Cho lược đồ quan hệ R và tập các phụ thuộc hàm:

F = {AB -> E, AG -> I, BE -> I, E -> G, GI -> H}

Chứng minh rằng: AB -> GHI.

Giải:

- Tính AB^+

Đặt $X_0 = AB$

Ta có X_1 = ABE (vì AB -> E)

Ta có X_2 = ABEIG (vì AB -> E, BE -> I, E -> G)

Ta có X_3 = ABEIGH (vì AB -> E, AG -> I, BE -> I, E -> G, GI -> H)

Ta có X_4 = ABEIGH (vì AB -> E, AG -> I, BE -> I, E -> G, GI -> H)

Vì $X_3 = X_4$ = ABEIGH $\supseteq$ GHI nên AB -> GHI.

Định lý 4.1

Hệ tiên đề Armstrong là đúng và đầy đủ.

Chứng minh:

Tính đúng đắn của hệ tiên đề đã được chứng minh qua bổ đề 4.1. Ở đây chỉ cần chứng minh tính đầy đủ, tức là nếu $X \rightarrow Y$ không thoả mãn trên r thì $X \rightarrow Y$ không thể suy dẫn logic từ F.

Gọi F là tập các phụ thuộc hàm trên tập thuộc tính U. Giả sử rằng $X \rightarrow Y$ là không thể suy dẫn được từ hệ tiên đề Armstrong. Xét quan hệ r gồm hai bộ phận như sau:

11...1	11...1
11...1	00...0
Các thuộc tính thuộc X	Các thuộc tính còn lại

Trước hết cần chỉ ra rằng, tất cả các phụ thuộc hàm thuộc F đều thoả mãn r. Thật vậy, giả sử $(V \rightarrow W) \in F$ nhưng không thoả trên r. Do đó $V \subseteq X^+$, hoặc hai bộ của r sẽ không bằng nhau ít nhất trên một thuộc tính của V. Như vậy W không thể là tập con của X^+ hoặc $V \rightarrow W$ thoả mãn trên r.

Gọi $A \in W$ nhưng A không thuộc X^+. Vì $XV \subseteq X^+$, $X \rightarrow V$ suy ra từ bổ đề 4.3 $(V \rightarrow W) \in F$ do vậy, nhờ luật bắc cầu suy ra $X \rightarrow A$. Nhưng do A không thuộc X^+ như giả thiết, do vậy dẫn đến mâu thuẫn. Từ đó kết luận rằng mỗi $(V \rightarrow W) \in F$ đều thoả mãn trên r.

Bây giờ cần chứng minh rằng, $X \rightarrow Y$ không thoả mãn trên r. Giả sử rằng

$X \rightarrow Y$ là thoả mãn trên r. Như trên có $X \subseteq X^+$ và suy ra $Y \subseteq X^+$, nếu không hai bộ thuộc r là bằng nhau trên X nhưng không bằng nhau trên Y. Theo bổ đề 4.3 thì $X \rightarrow Y$ có thể suy ra từ hệ tiên đề, điều đó là hoàn toàn mâu thuẫn. Do vậy, $X \rightarrow Y$ không thể đúng trên r. Đến đây có thể kết luận: Nếu $X \rightarrow Y$ không suy dẫn được từ tiên đề Armstrong thì $X \rightarrow Y$ không thể suy dẫn được từ F. Vậy hệ tiên đề là đầy đủ.

2.5. Tính toán bao đóng

Việc tính bao đóng F^+ của tập các phụ thuộc hàm F trong trường hợp tổng quát là rất khó khăn và tốn kém thời gian, bởi vì tập các phụ thuộc hàm F^+ rất lớn cho dù F có thể là nhỏ. Chẳng hạn $F = \{A \rightarrow B_1, A \rightarrow B_2, ., A \rightarrow B_n\}$. F^+ khi đó còn được tính cả những phụ thuộc hàm $A \rightarrow Y$ với $Y \subseteq \{B_1, \ldots, B_n\}$. Như vậy sẽ có 2^n tập con của Y nên có 2^n phụ thuộc hàm. Nhưng tính X^+, bao đóng của tập thuộc tính X lại không khó. Theo bổ đề 4.3 có thể kiểm tra được $X \rightarrow Y \in F^+$ hay không bằng cách tính X^+ ứng với F. Việc tính bao đóng X^+ được thể hiện qua thuật toán sau đây:

Thuật toán 4.1: Tính bao đóng của tập các thuộc tính đối với một tập các phụ thuộc hàm.

- Vào: tập U hữu hạn các thuộc tính, tập các phụ thuộc hàm F trên U và $X \subseteq U$.
- Ra: X^+, bao đóng của X đối với F.
- Phương pháp: Tính liên tiếp tập các thuộc tính $X_0, X_1, \ldots$ theo quy tắc:
- $X_0 = X$
- $X_{i+1} = X_i \cup A$ sao cho $\exists (Y \rightarrow Z) \in F, A \in Z$ và $Y \subseteq X_i$.

Bởi vì $X = X_0 \subseteq \ldots \subseteq U$, U là hữu hạn nên sẽ tồn tại một chỉ số i mà $X_i = X_{i+1}$. Khi đó $X^+ = X_i$.

Ví dụ 4.4: Cho F là tập tám phụ thuộc hàm sau:

$AB \rightarrow C$ $\quad D \rightarrow EG$ $\quad ACD \rightarrow B$

$C \rightarrow A$ $\quad BE \rightarrow C$ $\quad CE \rightarrow AG$

$BC \rightarrow D$ $\quad CG \rightarrow BD$

Và X = BD. Tính X^+

Áp dụng thuật toán 4.1:

Đặt $X_0 = BD$.

Muốn tính X_1, hãy chọn các phụ thuộc hàm có vế trái là con của BD (vế trái là B, D hoặc BD) và kết nạp các vế phải của chúng vào X_0. Ở đây chỉ có $D \rightarrow EG$.

Khi đó có $X_1 = BDEG$

Tương tự tìm các phụ thuộc hàm có vế trái là con X_1 (có $D \rightarrow EG$, $BE \rightarrow C$). Khi đó có:

$X_2 = BCDEG$ v.v. Tiếp tục sẽ có:

$X_3 = X_4 = ABCDEG$ và cuối cùng:

$X^+ = (BD)^+ = ABCDEG$.

Sau đây cần phải chỉ ra rằng thuật toán 4.1 là đúng.

Định lý 4.2

Thuật toán 4.1 tính X^+ là đúng.

Chứng minh: Chứng minh bằng quy nạp.

- Bước cơ sở: Đúng vì $A \in X$, rõ ràng $X \rightarrow A$.

- Bước quy nạp: Giả sử bước j - 1 đúng. Cần chứng minh cho bước thứ j. Tức là, nếu A thêm vào X_j thì $A \in X^+$. X_{j-1} chỉ chứa các thuộc tính X^+, A sẽ là thuộc tính được đưa vào X_j. Thật vậy, $A \in Z$, $(Y \rightarrow Z) \in F$ và $Y \subseteq X_{j-1}$, $Y \subseteq X^+$ theo giả thiết quy nạp. $X \rightarrow Y$ và $Y \rightarrow Z$ có $X \rightarrow Z$, $Z \rightarrow A$ (theo luật phản xạ) và $X \rightarrow A$ (theo luật bắc cầu) và do đó $A \in X^+$.

Ngược lại, cần chứng minh rằng, nếu $A \in X^+$ thì A phải thuộc vào X_i nào đó. Có điều không quan trọng là thuật toán 4.1 có thể kết thúc sớm hơn trước khi tính toán bước thứ j cho X_j. Nếu thuật toán dừng ở bước $X_i = X_{j+1}$ với $i < j$ thì rõ ràng $X_i = X_j$.

Do vậy $X_i = X^+$, trong đó có cả thuộc tính A.

Nếu trong quá trình chứng minh cần sử dụng tới hệ tiên đề Armstrong: $X \rightarrow Y$ suy dẫn từ F thì mỗi thuộc tính $A \in Y$ được thêm vào tại mỗi X_j nào đó. Các bước quy nạp sẽ thực hiện thêm một số dòng, trong đó mỗi dòng là một phụ thuộc hàm thuộc F và sử dụng luật phản xạ hoặc giả thiết của bước quy nạp trước hoặc sử dụng luật tăng trưởng và luật bắc cầu. Cuối cùng sẽ là $X \rightarrow Y$.

2.6. Khoá của lược đồ quan hệ

Đã có khái niệm khoá cho tập thực thể, đó là tập các thuộc tính xác định duy nhất một thực thể. Cũng đã có khái niệm khoá cho các quan hệ và bây giờ có khái niệm tương tự cho các quan hệ có các phụ thuộc hàm.

Cho lược đồ quan hệ R với các thuộc tính $U = \{A_1,..., A_n\}$ và các phụ thuộc hàm F, $X \subseteq U$. Ta nói X là một khoá của R nếu:

a. $X \rightarrow U \in F^+$. Nghĩa là X xác định hàm tất cả các thuộc tính (các phụ thuộc hàm này thuộc F hoặc được suy diễn logic từ F).

b. Không có $\varnothing \neq Y \subset X$ mà $Y \rightarrow U \in F^+$.

Các thuộc tính thuộc một khoá nào đó gọi là thuộc tính khoá. Còn các thuộc tính không nằm trong một khoá nào cả gọi là thuộc tính không khoá (thuộc tính thứ cấp).

Chú ý rằng, tính cực tiểu ở điều kiện (a) không được đặt ra khi nói đến khoá của tập thực thể. Ở đây thuật ngữ "khoá" bao hàm cả đặc tính cực tiểu. Vì vậy, khoá của một tập thực thể sẽ chỉ là khoá cho một lược đồ quan hệ biểu diễn cho tập thực thể đó nếu khoá này cực tiểu. Nếu không một hoặc nhiều tập con của khoá đó sẽ được dùng làm khoá cho lược đồ quan hệ. Một lược đồ quan hệ có nhiều khoá, thường chỉ định một khoá làm khoá chính (primary key). Thuật ngữ "khoá dự kiến" (dự tuyển) (candidate key) dùng để biểu thị tập các thuộc tính nhỏ nhất xác định hàm tất cả các thuộc tính khác (khoá). Còn thuật ngữ khoá dành cho một khoá dự kiến được chọn làm khoá chính.

Trong phần trước ta đã định nghĩa khoá là tập thuộc tính K bé nhất thoả mãn tính chất: Với mọi bộ t_1, t_2 khác nhau thuộc r của lược đồ quan hệ R ta luôn có $t_1[K] \neq t_2[K]$. Theo định nghĩa khoá trên ta đã chỉ ra tính cực tiểu của K hơn nữa ta cũng có với mọi bộ khác nhau t_1, t_2 của r ta luôn có $t_1[K] \neq t_2[K]$.

Thật vậy, giả sử $t_1[K] = t_2[K]$.

Vì nếu K là khoá của R nên $K \rightarrow U$ suy ra $t_1[U] = t_2[U]$ tức là $t_1 = t_2$ (mâu thuẫn).

Như vậy, định nghĩa khoá ở trên hoàn toàn phù hợp với định nghĩa khoá ở phần trước.

Từ định nghĩa trên suy ra:

* Muốn chứng minh X là khoá của R, phải thực hiện các bước sau:

- Chứng minh $X \rightarrow U \in F^+ \Leftrightarrow X^+ = U$.
- Chứng minh $\varnothing \neq \forall Y \subset X$ ta có $Y^+ \neq U$

$\Leftrightarrow \forall A \in X, \varnothing \neq X - A$ và $\{X - A\}^+ \neq U$

* Muốn tìm khoá của một lược đồ quan hệ ta dựa vào thuật toán tìm khoá.

2.7. Thuật toán tìm khoá

Thuật toán 4.2

- Vào: Lược đồ quan hệ R với tập thuộc tính U và tập phụ thuộc hàm F.
- Ra: Tập K là khoá của R.

Phương pháp:

- Đặt K = U
- Lặp lại quá trình loại khỏi K thuộc tính A mà $\{K - A\}^+ = U$.

Mô tả thuật toán (tựa Pascal):

```
Begin
  K:= U;
  FOR each A IN K DO
  IF {K - A}+ = U THEN K: = K - A
  END.
```

Thuật toán trên tìm được một khoá của R. Muốn tìm các khoá khác (nếu có) ta có thể thay đổi thứ tự loại bỏ các thuộc tính trong K.

Ví dụ 4.5: Cho lược đồ quan hệ ABCD với các phụ thuộc hàm $A \rightarrow C$ và $B \rightarrow D$.

Hãy tìm khoá của lược đồ quan hệ trên.

- K = U = ABCD
- Lần lượt loại bỏ các thuộc tính trong K:
  + Loại bỏ D: Ta có $ABC^+ = U$ (vì $A \rightarrow C$ và $B \rightarrow D$) nên K = ABC
  + Loại bỏ C: Ta có $AB^+ = U$ (vì $A \rightarrow C$ và $B \rightarrow D$) nên K = AB
  + Loại bỏ B: Ta có $A^+ = AC \neq U$ (vì $A \rightarrow C$) nên K = AB
  + Loại bỏ A: Ta có $B^+ = BD \neq U$ (vì $B \rightarrow D$) nên K = AB

Vậy K = AB.

Từ thuật toán tìm khoá ta có các nhận xét sau:

- Các thuộc tính không xuất hiện trong cả vế trái và vế phải của tập phụ thuộc hàm F phải có trong khoá.

- Các thuộc tính chỉ xuất hiện bên vế trái của các phụ thuộc hàm trong F cũng phải thuộc khoá.

- Trong quá trình tìm khoá có thể bỏ đi tất cả các thuộc tính đơn phía bên phải của các phụ thuộc hàm trong F. Tuy nhiên cần phải kiểm tra lại vì không phải lúc nào các thuộc tính đó cũng bỏ được

Các nhận xét này giúp có thể phát hiện ra khoá một cách nhanh chóng. Chẳng hạn cho lược đồ quan hệ R với tập thuộc tính {A, B, C, D} và tập phụ thuộc hàm F = {A → C, B → D} có thể phát hiện ra ngay khoá của R là AB.

2.8. Phủ của tập các phụ thuộc hàm

Gọi F và G là các tập của các phụ thuộc hàm. Nói rằng F và G là tương đương nếu $F^+ = G^+$. Nếu F và G là tương đương đôi khi còn nói F phủ G (và G phủ F). Dễ dàng kiểm tra liệu F và G có tương đương với nhau không.

Phương pháp kiểm tra: Lấy mọi phụ thuộc hàm Y → Z thuộc F, kiểm tra xem liệu Y → Z có thuộc G^+ không? Dùng thuật toán 4.1 để tính Y_G^+ và kiểm tra liệu $Z \subseteq Y_G^+$ không? Nếu tồn tại một phụ thuộc hàm Y → Z thuộc F mà không thuộc G^+ thì chắc chắn $F^+ \neq G^+$.

Nếu mỗi phụ thuộc hàm V → W thuộc F cũng thuộc G^+ thì mỗi phụ thuộc hàm V → W thuộc F^+ cũng thuộc G^+.

Để kiểm tra mỗi phụ thuộc hàm G là thuộc F^+ quá trình làm hoàn toàn tương tự. Do đó F và G là tương đương khi và chỉ khi mỗi phụ thuộc hàm thuộc F là thuộc G^+ và mỗi phụ thuộc hàm thuộc G là thuộc F^+.

Bổ đề 4.4

Mỗi tập các phụ thuộc hàm F đều được phủ bằng tập các phụ thuộc hàm G mà vế phải các phụ thuộc hàm đó bao gồm không quá một thuộc tính.

Chứng minh:

Gọi G là tập các phụ thuộc hàm X → A sao cho với X → Y thuộc F thì A ∈ Y. Từ X → Y suy ra X → A (theo luật tách).

Do vậy $G \subseteq F^+$.

Ngược lại, có $F \subseteq G^+$ vì nếu $Y = A_1,..., A_n$ thì X → Y được suy ra từ

$X \rightarrow A_1,..., X \rightarrow A_n$ nhờ luật hợp.

Để có thể phục vụ quá trình thiết kế lược đồ cơ sở dữ liệu, sau đây sẽ đưa ra một số khái niệm.

Gọi tập các phụ thuộc hàm F là tối thiểu nếu:

a. Mỗi vế phải của một phụ thuộc hàm phụ thuộc F chỉ có một thuộc tính.

b. Không tồn tại một phụ thuộc hàm $X \rightarrow A$ thuộc F mà:

$$F^+ = (F - \{X \rightarrow A\})^+$$

c. Không tồn tại một phụ thuộc hàm $X \rightarrow A$ thuộc hàm F và một tập con Z của X mà:

$$F^+ = (F - \{ X \rightarrow A\} \cup \{Z \rightarrow A\})^+$$

Về trực quan, điều kiện (b) bảo đảm cho tập F không có một phụ thuộc hàm nào là dư thừa và để kiểm tra $X \rightarrow A$ có dư thừa hay không bằng cách tính X^+ ứng với $F - \{X \rightarrow A\}$.

Điều kiện (c) bảo đảm không có một thuộc tính nào tham gia phía trái của phụ thuộc hàm là dư thừa và phép kiểm tra các phụ thuộc hàm dư thừa ở vế trái như sau: Thuộc tính B trong X đối với phụ thuộc hàm $X \rightarrow A$ là dư thừa nếu và chỉ nếu A thuộc $(X - \{B\})F^+$.

Vế phải của phụ thuộc hàm ở điều kiện (a) chỉ có một thuộc tính bảo đảm chắc chắn không có một thuộc tính nào ở vế phải là dư thừa.

Nếu G là một tập phụ hàm tối thiểu theo nghĩa trên và G tương đương F thì ta nói rằng G là phủ tối thiểu của F.

Định lý 4.3

Mỗi tập phụ thuộc hàm F đều tương đương với một tập F' tối thiểu.

Chứng minh:

Theo bổ đề 4.4, giả sử rằng không vế phải nào của các phụ thuộc hàm của F có nhiều hơn một thuộc tính. Để kiểm tra điều kiện (b) xét các phụ thuộc hàm $(X \rightarrow Y) \in F$. Nếu $(F - \{X \rightarrow Y\})^+ = F^+$ thì loại bỏ $X \rightarrow Y$ khỏi F. Để kiểm tra điều kiện (c) xét các phụ thuộc hàm $(X \rightarrow A) \in F$. Nếu $B \in X$ và $(X - B) \rightarrow A$ thì loại bỏ B khỏi X. Cuối cùng ta thu được F' tối thiểu.

Chú ý rằng, các phụ thuộc hàm F được sắp xếp theo một thứ tự khác nhau thì sẽ cho kết quả khác nhau.

Ví dụ 4.6: Cho tập F gồm:

$A \rightarrow B$ $A \rightarrow C$ $B \rightarrow C$

$B \rightarrow A$ $C \rightarrow A$

Có thể loại bỏ khỏi F: $B \rightarrow A$ và $A \rightarrow C$ hoặc loại bỏ $B \rightarrow C$ nhưng không thể đồng thời loại bỏ cả ba phụ thuộc hàm.

Như vậy điều kiện b) được thoả mãn. Cần kiểm tra điều kiện c) cho tập các phụ thuộc còn lại của F. Ở đây cũng cần lưu ý tới thứ tự các thuộc tính xuất hiện bên vế trái của các phụ thuộc hàm.

Loại bỏ các thuộc tính vế trái của các phụ thuộc hàm sao cho tập các thuộc tính vẫn là tương đương. Quá trình tiếp tục cho đến khi không thể loại bỏ được thuộc tính nào nữa. Như vậy, tập các phụ thuộc còn lại của F sẽ tạo nên F' và thoả mãn ba điều kiện của một tập tối thiểu.

Ví dụ 4.7: Cho tập F:

$AB \rightarrow C$ $D \rightarrow E$ $CG \rightarrow D$

$C \rightarrow A$ $D \rightarrow G$ $CE \rightarrow A$

$BC \rightarrow D$ $BE \rightarrow C$ $CE \rightarrow G$

$ACD \rightarrow B$ $CG \rightarrow B$

Áp dụng thuật toán 4.1 cho tập F theo thứ tự từ phải qua trái và từ dưới lên trên, rõ ràng $CE \rightarrow A$ là dư thừa vì có thể suy ra từ $C \rightarrow A$; $CG \rightarrow B$ là dư thừa vì $CD \rightarrow D$, $C \rightarrow A$ và $ACD \rightarrow B$ suy ra $CG \rightarrow B$.

Từ đó dễ dàng tính được $(CG)^+$ và không còn phụ thuộc hàm nào là dư thừa.

$ACD \rightarrow B$ có thể thay bởi $CD \rightarrow B$ vì $C \rightarrow A$.

Từ đó có tập tối thiểu là:

$AB \rightarrow C$ $D \rightarrow G$

$C \rightarrow A$ $BE \rightarrow C$

$BC \rightarrow D$ $CG \rightarrow D$

$CD \rightarrow B$ $CE \rightarrow G$

$D \rightarrow E$

Nếu loại bỏ CE → A, CG → D và ACD → B sẽ có một tập tối thiểu khác:

$AB \rightarrow C$ $\quad D \rightarrow G$

$C \rightarrow A$ $\quad BE \rightarrow C$

$BC \rightarrow D$ $\quad CG \rightarrow B$

$D \rightarrow E$ $\quad CE \rightarrow G$

Lưu ý rằng, hai tập tối thiểu nêu trên có số lượng các phụ thuộc hàm là khác nhau.

II. PHÉP TÁCH CÁC LƯỢC ĐỒ QUAN HỆ

Phép tách một lược đồ quan hệ $R = \{A_1, A_2,..., A_n\}$ là việc thay thế nó bằng một tập lược đồ $P = \{R_1,..., R_k\}$, trong đó $R_i \subseteq R$, $i = 1,..., k$ và $R = R_1 \cup R_2 \cup ... \cup R_k$.

Ở đây không đòi hỏi các lược đồ R_i phải là phân biệt. Mục tiêu của phép tách chủ yếu là loại bỏ các dị thường dữ liệu.

Ví dụ 4.8: Cho lược đồ quan hệ người cung cấp:

S(SNAME, ADD, PRO, PRICE)

Và giả sử có các phụ thuộc hàm:

SNAME → ADD

SNAME, PRO → PRICE

Lược đồ S có thể được thay thế bằng hai lược đồ khác là:

S_1(SNAME, ADD) và S_2(SNAME, PRO, PRICE)

Như vậy rõ ràng ở S_1 mỗi nhà cung cấp chỉ cần một lần lưu địa chỉ tương ứng của họ chứ không phải lặp đi lặp lại như ở lược đồ S.

Vấn đề đặt ra là liệu một quan hệ r bất kỳ thoả mãn trên S, khi tách ra thành S_1 và S_2 có còn phù hợp không? Hai hình chiếu $r[S_1]$ và $r[S_2]$ có còn phù hợp không? Hai hình chiếu $r[S_1]$ và $r[S_2]$ có còn giữ được cùng thông tin với r không? Nói cách khác, nếu kết nối hai hình chiếu $r[S_1]$ và $r[S_2]$ thành một quan hệ mới, ví dụ $s = r[S_1] * r[S_2]$, liệu $r \neq s$ hay $r = s$? Để nghiên cứu các điều kiện cho kết quả phép kết nối tự nhiên nêu trên là duy nhất và bằng quan hệ ban đầu, cần thiết có thêm một số khái niệm sau:

1. Kết nối không mất mát thông tin

Nếu R là một lược đồ quan hệ được tách thành các lược đồ con R_1, R_2,..., R_k và F là tập các phụ thuộc hàm, ta nói rằng phép tách là tách - kết nối không mất mát thông tin (lossless join decompoition) đối với F nếu với mỗi quan hệ r của R thoả mãn F ta có:

$$r = * \prod_{Ri}(r) = \prod_{R1}(r))* \prod_{R2}(r))*...* \prod_{Rk}(r)$$

Tức là mỗi quan hệ r là kết nối tự nhiên của các hình chiếu của nó trên các R_i, i = 1,..., k. Đặc tính tách - kết nối không mất mát thông tin là cần thiết nếu quan hệ bị tách cần phải được khôi phục lại từ việc tách của chính nó.

Sau đây xem xét một số tính chất của tách - kết nối không mất mát thông tin, nhưng trước tiên ta đưa ra một số ký hiệu.

Nếu P = (R_1,..., R_k) là một phép tách thì m_p là ánh xạ được định nghĩa là $m_\rho(r) = * \prod_{Ri}(r)$, i =1,..., k, có nghĩa là m_ρ (r) là kết nối của các phép chiếu của r trên các lược đồ con trong P. Điều kiện để kết nối không mất mát thông tin đối với tập phụ thuộc hàm F được biểu diễn như sau:

Với mọi r của R thoả mãn F, $r = m_\rho(r)$

Bổ đề 4.5

Gọi R là một lược đồ quan hệ, r = (R_1, R_2,..., R_k) là một phép tách của R, r là quan hệ của R và ri = $\prod_{Ri}(r)$ thì:

a. $r \subseteq m_\rho(r)$

b. Nếu $s = m_\rho(r)$ *thì* $\prod_{Ri}(r) = ri$

c. $m_\rho(m_\rho(r)) = m_\rho(r)$

Chứng minh:

Gọi t là một bộ thuộc r. Vậy với mỗi i, $t_i = t[R_i]$ r_i. Theo định nghĩa của phép kết nối tự nhiên, $t \in m_\rho(r)$ vì $t[R_i] = t_i$ cho $\forall$ i trên các thuộc tính của R_i.

Theo (a) có $r \subseteq s$, suy ra $\prod_{Ri(r)} \subseteq \prod_{Ri(s)}$. Điều đó có nghĩa là $ri \subseteq \prod_{Ri(s)}$. Giả sử với một i mà $ti \in \prod_{Ri(s)}$. Khi đó có ts sao cho $t[R_i] = t_i$. Cũng vì $t \in$ s sao cho $t[R_i] = u_j$ cho nên có $u_j \in r_i$ sao cho $t[R_i] = u_j$. Trong trường hợp này t[Ri] $\in r_i$. Nhưng vì $t[R_i] = t_i$, do vậy $t_i = r_i$ và do đó $\prod_{Ri(s)} \subseteq ri$. Từ đó có $r_i = \prod_{Ri(s)}$.

Nếu $s = m_\rho(r)$ thì theo b) có $\prod_{Ri(s)} = r_i$. Do vậy:

$$m_\rho(s) = *_{i=1}^{k} r_i = m_\rho(r).$$

Cần chú ý trong trường hợp tổng quát, với mỗi i, r_i là một quan hệ hiện hành của R_i.

$$\text{và } s = *_{i=1}^{k} r_i = m_\rho(r).$$

thì khi đó $\prod_{Ri(s)}$ không nhất thiết phải bằng r_i. Mối quan hệ ri có thể thiếu hoặc thừa một số bộ nào đó (đều gọi là mất thông tin). Chẳng hạn, nếu $R_1 = AB$, $R_2 = BC$ và các quan hệ tương ứng là $r_1 = \{a_1b_1\}$, $r_1 = \{b_1c_1, b_2c_2\}$, khi đó: $s = \{a_1\ b_1\ c_1\}$.

Và xảy ra $\prod_{BC}(s) = \{b_1c_1\}\ r_2$ (trường hợp này bị mất bộ $\{b_2c_2\} \in r_2$)

2. Kiểm tra phép tách kết nối không mất mát thông tin

Liệu một phép tách có kết nối không mất mát thông tin hay không đối với tập các phụ thuộc hàm được kiểm tra qua thuật toán sau đây:

Thuật toán 4.3: Kiểm tra phép tách kết nối không mất mát thông tin.

- Vào: Lược đồ quan hệ $R = \{A_1,..., A_n\}$, tập các phụ thuộc hàm F và phép tách $P = (R_1,..., R_k)$.

- Ra: Kết luận phép tách P có kết nối mất mát thông tin hay không?

Phương pháp:

- Thiết lập một bảng với n cột và k hàng; cột thứ j ứng với thuộc tính A_j, hàng thứ i ứng với lược đồ R_i.

- Tại hàng i và cột j điền ký hiệu a_j nếu $A_j \in R_i$, nếu không điền ký hiệu b_{ij}.

Bây giờ xem xét đến các phụ thuộc hàm trong F áp dụng cho bảng vừa thiết lập được. Mỗi lần xét một phụ thuộc hàm $(X \rightarrow Y) \in F$, tìm các hàng có giá trị bằng nhau ở tất cả các cột trong X (chẳng hạn có hai hàng t_1, t_2 mà

$t_1[X] = t_2[X]$) thì làm bằng các giá trị ở tất cả các cột của hai hàng đó trong Y (chẳng hạn $t_1[Y] = t_2[Y]$). Chú ý khi làm bằng giá trị trên mỗi cột trong Y, nếu một trong hai giá trị trên hai hàng đó là a_j thì ưu tiên làm bằng chúng bằng ký hiệu a_j. Nếu không thì làm bằng chúng bằng một trong hai ký hiệu b_{ij}. Tiếp tục áp dụng các phụ thuộc hàm cho bảng (kể cả việc lặp lại phụ thuộc hàm đã được áp dụng) cho tới khi không còn áp dụng được nữa.

Xem xét bảng kết quả: Nếu xuất hiện một hàng gồm các ký hiệu a_1, a_2,..., a_n (hàng chứa toàn a) thì phép tách kết nối không mất mát thông tin. Ngược lại thì phép tách kết nối mất mát thông tin (lossy join).

Ví dụ 4.9: Xem xét ví dụ ở quan hệ cung cấp ở trên:

S (SNAME, ADD, PRO, PRICE)

Được tách làm hai quan hệ:

S_1 (SNAME, ADD)

và S_2 (SNAME, PRO, PRICE)

Với tập phụ thuộc hàm:

SNAME → ADD, SNAME, PRO → PRICE.

Bảng ban đầu được thiết lập như sau:

SNAME	ADD	PRO	PRICE
a_1	a_2	b_{13}	b_{14}
a_1	b_{22}	a_3	a_4

Áp dụng phụ thuộc hàm SNAME → ADD cho hai hàng của bảng. Hai hàng làm bằng a_2.

Bảng kết quả là:

SNAME	ADD	PRO	PRICE
a_1	a_2	b_{13}	b_{14}
a_1	a_2	a_3	a_4

Bảng kết quả có dòng thứ hai có các giá trị toàn là a, do đó nó là phép tách kết nối không mất mát thông tin.

Định lý 4.4

Thuật toán 4.3 xác định chính xác một phép tách kết nối có hay không mất mát thông tin.

Từ định lý trên dễ dàng kiểm tra trường hợp tách thành hai lược đồ con qua định lý sau:

Định lý 4.5

Nếu $\rho = (R_1, R_2)$ là một phép tách của lược đồ quan hệ R và F là tập phụ thuộc hàm thì ρ là tách không mất mát thông tin đối với F khi và chỉ khi $R_1 \cap R_2 \rightarrow R_1 - R_2$ hoặc $R_1 \cap R_2 \rightarrow R_2 - R_1$.

Chú ý rằng, các phụ thuộc hàm nêu trên không nhất thiết phải thuộc tập phụ thuộc hàm F ban đầu, chỉ cần chúng thuộc F^+.

Chứng minh:

Bảng ban đầu được thiết lập như sau:

	$R_1 \cap R_2$	$R_1 - R_2$	$R_2 - R_1$
Hàng cho R_1	aa... a	aa... a	bb... b
Hàng cho R_2	aa... a	bb... b	aa... a

Ta thấy ngay: nếu $R_1 \cap R_2 \rightarrow R_1 - R_2$ hoặc $R_1 \cap R_2 \rightarrow R_2 - R_1$ thì:

$R_1 \cap R_2 \rightarrow R_1 - R_2$ nên dòng hai của bảng kết quả toàn a.

Hoặc $R_1 \cap R_2 \rightarrow R_2 - R_1$ nên dòng một của bảng kết quả toàn a.

Do đó, phép tách kết nối không mất mát thông tin.

Ngược lại, nếu phép tách trên có kết nối không mất mát thông tin thì ở bảng kết quả có dòng một hoặc dòng hai toàn a.

Nếu dòng một toàn a thì thấy ngay $R_1 \cap R_2 \rightarrow R_2 - R_1$. Còn nếu dòng hai toàn a thì thấy ngay $R_1 \cap R_2 \rightarrow R_1 - R_2$.

Ví dụ 4.10: R = ABC và $F = \{A \rightarrow B\}$, $\rho = (R_1, R_2)$, $R_1 = AB$, $R_2 = AC$.

Ta có:

$A = R_1 \cap R_2$

$B = R_1 - R_2$

Mà $A \rightarrow B$ tức là $R_1 \cap R_2 \rightarrow R_1 - R_2$ nên phép tách trên kết nối không mất mát thông tin.

3. Phép tách bảo toàn phụ thuộc hàm

Theo phần trên, một phép tách cần phải có đặc tính kết nối không mất mát thông tin vì nó cho phép khôi phục lại một quan hệ từ các hình chiếu của nó. Một đặc tính quan trọng khác của phép tách $\rho = (R_1, R_2,... R_n)$ của lược đồ quan hệ R là có thể suy ra được tập các phụ thuộc hàm F của R từ các hình chiếu của F trên các R_i.

3.1. Khái niệm phép tách bảo toàn phụ thuộc hàm

Cho lược đồ quan hệ R, F là tập các phụ thuộc hàm.

Và $\rho = (R_1, R_2,... R_k)$ là một phép tách.

Chiếu của tập phụ thuộc hàm F trên một tập R_i ký hiệu $\prod_{Ri}(F)$ gồm những phụ thuộc hàm $X \rightarrow Y$ của F^+ mà $X \rightarrow Y$ nằm gọn trong R_i ($XY \subseteq R_i$).

$<=> \prod_{Ri}(F) = \{X \rightarrow Y \in F^+ \mid XY \subseteq R_i\}$

Quy ước:

- Cho R là một lược đồ quan hệ.
- F là tập các phụ thuộc hàm của R.
- Ký hiệu W = <R, F> gọi là một sơ đồ quan hệ (SĐQH).

Ví dụ 4.11:

W = <R, F> -SĐQH

$R = \{A, B, C, D\}$

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

$\rho = (R_1, R_2, R_3)$

$R_1 = \{A, B\}, R_2 = \{B, C\}, R_3 = \{C, A\}$

Suy ra:

$\prod_{R1}(F) = \{A \rightarrow B, B \rightarrow A\}$

$\prod_{R2}(F) = \{B \rightarrow C, C \rightarrow B\}$

$\prod_{R3}(F) = \{C \rightarrow D, D \rightarrow C\}$

Vậy cho W = <R,F>

Ta nói phân rã $\rho = (R_1, R_2,... R_k)$ bảo toàn phụ thuộc hàm nếu:

$$F \sim \bigcup_{i}^{k} \prod R_i(F)$$

Ví dụ trên có phân rã bảo toàn phụ thuộc hàm. Vì

$$F \sim \bigcup_{i=1}^{k} \prod R_i(F) = \{A \to B, B \to A; B \to C, C \to B, D \to C, C \to D\} = G$$

Thật vậy: hiển nhiên $G \subset F^+$

Tpcm $F \subset G^+$

Thật vậy $D \to A \in F$ mà $D \to C \in G^+$

$C \to B \in G^+$

$B \to A \in G^+$

$\Rightarrow D \to A \in G^+$

Chứng minh tương tự cho các phụ thuộc hàm khác. Vậy phép tách trên có bảo toàn phụ thuộc hàm.

Chú ý: Một phép tách có thể có kết nối không mất mát thông tin ứng với một tập phụ thuộc hàm F nhưng không bảo toàn F. Cũng có phép tách có thể bảo toàn F nhưng không có tính chất kết nối không mất mát thông tin.

Ví dụ 4.12: Cho lược đồ quan hệ $R = \{C, S, Z\}$

$F = \{CS \to Z, Z \to C\}$

$\rho = (SZ, CZ)$ có kết nối không mất mát thông tin vì:

$(SZ \cap CZ) \to (SZ - CZ)$ nhưng không bảo toàn phụ thuộc hàm.

Ngược lại: Cho R = ABCD

$F = \{A \to B, C \to D\}$ và $\rho = (AB, CD)$ có kết nối mất mát thông tin nhưng bảo toàn phụ thuộc hàm.

3.2. Thuật toán kiểm tra phép tách bảo toàn phụ thuộc hàm

- Vào: W = <R, F> - SĐQH

  $\rho = (R_1, R_2, ..., R_k)$ là một phép tách.

- Ra: Khẳng định phép tách ρ có bảo toàn phụ thuộc không?

Phương pháp: ý tưởng r bảo toàn phụ thuộc hàm:

$$<=> F \sim \bigcup_{i=1}^{k} \prod R_i(F) = G$$

Hiển nhiên $G \subseteq F^+$

Vậy ρ có bảo toàn phụ thuộc hàm <=> $F \subseteq G^+$

<=> $\forall X \to A \in F$ thì $X \to A \in G^+$

<=> $X_G^+ \ni A$

Vậy thuật toán để xét ρ có bảo toàn phụ thuộc không?

b1: Tính $G = \bigcup_{i=1}^{k} \prod R_i(F)$

b2: $\forall X \to Y \in F$ xét xem $X_G^+ \supseteq Y$ không?

- Nếu $\forall X \to Y \forall F$ ta đều có $X_G^+ \supseteq Y$ thì kết luận: ρ có bảo toàn phụ thuộc hàm.

- Nếu $\exists X \to A \in F$ mà X_G^+ không chứa Y thì kết luận: ρ không bảo toàn phụ thuộc hàm.

Ví dụ 4.13:

$R = \{A, B, C, D, E\}$

$F = \{AB \to C, CD \to E\}$

$P = (R_1, R_2, R_3)$

$R_1 = \{A, B\}$; $R_2 = \{C, D\}$; $R_3 = \{D, E\}$;

B_1: Tính $G = \{ AB \to A, AB \to B, CD \to C, CD \to D\}$

B_2: $AB \to C \in G^+$?

Ta có: $\{AB\}_G^+ = \{AB\}$ không chứa C. Vậy phép tách trên không bảo toàn phụ phuộc hàm.

III. CHUẨN HOÁ LƯỢC ĐỒ QUAN HỆ

Nếu một lược đồ quan hệ thiết kế không tốt sẽ gây ra những dị thường dữ liệu như dư thừa dữ liệu và do việc cập nhật dữ liệu (qua phép tính chèn, loại bỏ và thay đổi dữ liệu). Để tránh dị thường dữ liệu, lược đồ quan hệ cần thiết

phải được biến đổi thành các dạng phù hợp. Quá trình đó được xem là quá trình chuẩn hoá lược đồ quan hệ. Lược đồ quan hệ được chuẩn hoá là lược đồ quan hệ trong đó miền của mỗi thuộc tính chỉ chứa những giá trị nguyên tố (atomic) tức là không phân nhỏ được nữa và do đó mỗi giá trị trong quan hệ cũng là nguyên tố (đơn trị).

Ngược lại, gọi là lược đồ quan hệ không chuẩn hoá. Quá trình chuẩn hoá một lược đồ quan hệ có thể thành một hoặc nhiều lược đồ quan hệ chuẩn hoá khác và có kết nối không mất mát thông tin.

Lược đồ quan hệ R là chuẩn hoá thì quan hệ r của nó cũng là chuẩn hoá.

Ví dụ 4.14:

GV	MH	PHONG
A	CSDL	414, H3
B, C	CSDL	408, 410

Quan hệ không chuẩn hoá

GV	MH	PHONG
A	CSDL	414
A	CSDL	H3
B	CSDL	408
C	CSDL	410

Quan hệ chuẩn hoá

Trước khi mô tả chi tiết các dạng chuẩn hóa cần đưa ra một số khái niệm sau:

Cho một lược đồ quan hệ R với tập thuộc tính $U = \{A_1,..., A_n\}$. Thuộc tính AU được gọi là thuộc tính khoá nếu A thuộc một khoá nào đó của R, ngược lại A được gọi là thuộc tính không khoá.

Ví dụ 4.15: Cho lược đồ quan hệ R trên tập thuộc tính U = {A, B, C, D} với các phụ thuộc AB → C, B → D và BC → A. Rõ ràng AB và BC là khoá của lược đồ R. Khi đó A, B và C là thuộc tính khoá, còn D là thuộc tính không khoá.

Định nghĩa 4.2

Cho lược đồ quan hệ R trên tập thuộc tính $U = \{A_1,..., A_k\}$. X và Y là hai tập thuộc tính khác nhau $X \subseteq Y$ Và $Y \subseteq U$.

Y phụ thuộc hàm dầy đủ (fully functional dependence) vào X nếu Y phụ thuộc hàm vào X nhưng không phụ thuộc hàm vào bất kỳ một tập hợp con thực sự nào của X.

1. Các dạng chuẩn

Trong lý thuyết ban đầu, Codd đưa ra có ba dạng chuẩn của quan hệ:

Dạng không chuẩn hoá

↓

Dạng chuẩn thứ nhất (First Normal Form, viết tắt là 1NF)

↓

Dạng chuẩn thứ hai (Second Normal Form, viết tắt là 2NF)

↓

Dạng chuẩn thứ ba (Third Normal Form, viết tắt là 3NF)

Sau này còn có nhiều dạng chuẩn khác như: BCNF, 4NF, 5NF,...

1.1. Dạng chuẩn thứ nhất (First Normal Form)

Định nghĩa 4.3

Một lược đồ quan hệ R với tập thuộc tính U được gọi là ở dạng chuẩn thứ nhất (1NF) nếu miền của mỗi thuộc tính trong U chỉ chứa những giá trị nguyên tố (atomic).

Định nghĩa này cho thấy: Bất kỳ một lược đồ quan hệ chuẩn hoá nào cũng ở dạng 1NF và quan hệ r của nó cũng ở dạng chuẩn 1.

Quy ước: Một lược đồ quan hệ nếu không nêu rõ nó không là 1NF thì coi như nó đã ở 1NF.

1.2. Dạng chuẩn thứ hai (Second Normal Form)

Định nghĩa 4.4

Một lược đồ quan hệ R với tập thuộc tính U được gọi là ở dạng chuẩn thứ hai nếu nó ở dạng chuẩn thứ nhất và mỗi thuộc tính không khoá của R đều phụ thuộc hàm đầy đủ vào khoá. Hay nói một cách khác, mỗi thuộc tính không khoá của nó không phụ thuộc hàm vào một tập con thực sự nào của khoá.

Ví dụ 4.16:

* Cho lược đồ quan hệ R(SAIP) với các phụ thuộc hàm SI → P và S → A.

R không là 2NF.

Thật vậy: Đặt Y = SI, Y = S. Ta có A là thuộc tính không khoá vì R chỉ

có 1 khoá là SI. Mà $Y \subset X$ và $Y \rightarrow A$. Như vậy A không phụ thuộc hàm đầy đủ vào khoá.

* Cho lược đồ quan hệ R(SAIP) - 1NF với phụ thuộc hàm $SI \rightarrow P$.

R là 2NF.

Thật vậy:

Ta có A, P là các thuộc tính không khoá.

R chỉ có 1 khoá là SI và không có tập con nào của SI xác định hàm A và P, tức là S hoặc I không xác định hàm A và P. Như vậy A, P phụ thuộc hàm đầy đủ vào khoá SI.

1.3. Dạng chuẩn thứ ba (Third Normal Form)

Định nghĩa 4.5

Một lược đồ quan hệ R với tập thuộc tính U được gọi là ở dạng chuẩn thứ ba (3NF) nếu nó ở dạng chuẩn thứ hai và không có một tập con X nào của U, mà X không phải là khoá lại xác định hàm các thuộc tính không khoá của R.

Ví dụ 4.17:

* Cho lược đồ quan hệ (R(CSZ) - 1NF với phụ thuộc hàm $CS \rightarrow Z$.

Trong lược đồ này chỉ có một khoá là CS nên nó là khoá chính, thuộc tính không khoá là Z, cho thấy R là 3NF.

* Cho lược đồ R(SIDM) - 1NF và các phụ thuộc $SI \rightarrow D$ và $SD \rightarrow M$.

Ở đây chỉ có một khoá chính là SI. Thuộc tính không khoá là D, M. Rõ ràng R ở 2NF nhưng không phải ở 3NF (vì $SD \rightarrow M$).

Chú ý: Nếu R không chứa thuộc tính không khoá thì R ở dạng 3NF.

1.4. Dạng chuẩn Boye - Codd (Boye - Codd Normal Form)

Định nghĩa 4.6

Một lược đồ quan hệ R với tập các phụ thuộc hàm F được gọi là ở dạng chuẩn Boye - Codd (BCNF) nếu $X \rightarrow A$ thoả mãn trên R và $A \in X$ thì X là một khoá của R.

Hay nói một cách khác, không tồn tại một phụ thuộc hàm dạng $X \rightarrow A$ với $X \subset U$ và $A \notin X$ mà $X^+ \neq U$.

Từ định nghĩa ta thấy ngay: nếu R - BCNF thì R - 3NF, vì trong 3NF chỉ cấm các thuộc tính A không khoá phụ thuộc vào tập $X \subset U$ mà $A \notin X$

và $X^+ \neq U$, còn trong BCNF cấm tất cả các thuộc tính $A \notin X$ phụ thuộc vào tập $X \subset U$ mà $X^+ \neq U$.

Ví dụ 4.18: Cho R(CSZ) với tập thụ thuộc hàm, $CS \rightarrow Z$, $Z \rightarrow C$ rõ ràng R không ở BCNF mà ở 3NF vì $Z \rightarrow C$ nhưng Z không phải là khoá của R và C là thuộc tính khoá.

Từ ví dụ này cho thấy một lược đồ quan hệ có thể là 3NF nhưng chưa chắc phải là BCNF. Nhưng một lược đồ quan hệ là BCNF thì nó là 3NF.

2. Chuẩn hoá qua phép tách có kết nối không mất mát thông tin

Vấn đề đặt ra là tách có kết nối không mất mát thông tin một lược đồ quan hệ thành các lược đồ con ở dạng chuẩn thứ 3 hoặc dạng chuẩn Boye-codd vẫn bảo toàn các phụ thuộc hàm.

2.1. Phép tách lược đồ quan hệ thành BCNF kết nối không mất mát thông tin

Trước khi đưa ra thuật toán tách, cần khảo sát một số tính chất sau:

Bổ đề 4.6

a) Cho R là một lược đồ quan hệ với tập phụ thuộc hàm F. Gọi $\rho = (R_1, R_2, ..., R_k)$ *là một phép tách kết nối không mất mát thông tin của R đối với F. Gọi* F_i *là hình chiếu của F trên* R_i, $i = 1, ..., k$, *tức là tập các* $(X \rightarrow Y) \in F^+$ *sao cho* $X \supseteq R_i$ *và* $Y \supseteq R_i$.

Gọi $\delta = (S_1, ..., S_m)$ *là phép tách kết nối không mất mát thông tin của* R_i.

Khi đó phép tách của R thành $(R_1, ..., R_{i-1}, S_1, ..., S_m, R_{i+1}, ..., R_k)$ *đối với F là không mất mát thông tin.*

b) Mọi lược đồ có hai thuộc tính hoặc tập phụ thuộc hàm $F = \varnothing$ *đều có dạng BCNF.*

c) Nếu R không có dạng BCNF thì có thể tìm được các thuộc tính A và B trong R sao cho $(R - AB) \rightarrow A$. *Phụ thuộc hàm* $(R - AB) \rightarrow A$ *có thể cũng đúng trong trường hợp này.*

Bổ đề 4.7

Cho tập phụ thuộc hàm F trên R. Nếu chiếu F trên $R_1 \subseteq R$ *để được* F_1, *rồi lại chiếu* F_1 *trên* $R_1 \subseteq R_2$ *để được* F_2 *thì:*

$$F_2 = \pi_{R1}(F)$$

Thuật toán 4.4: Tách một lược đồ quan hệ thành BCNF.

- Vào: Lược đồ quan hệ R và tập phụ thuộc hàm F.

- Ra: Phép tách của R có kết nối không mất mát thông tin sao cho mỗi lược đồ quan hệ trong phép tách đều ở BCNF đối với phép chiếu của F trên lược đồ đó.

Phương pháp:

Cấu trúc phép tách ρ trên R theo phương pháp lặp liên tiếp. Tại mỗi bước phép tách ρ bảo đảm không mất mát thông tin đối với F.

- Bước đầu: ρ chỉ bao gồm R

- Các bước tiếp: Nếu S là một lược đồ thuộc ρ, S chưa ở BCNF, chọn $X \rightarrow A$ là phụ thuộc hàm không thoả mãn trên BCNF, tức là X không chứa khóa của S và $A \notin X$. Thay thế S trong ρ bởi S_1 và S_2 với:

$$S_1 = XA \text{ và } F_1 = \pi_{S1}(F),$$

$$S_2 = S - A \text{ và } F_2 = \pi_{S2}(F)$$

Theo định lý 4.5, phép tách S thành S_1 và S_2 có kết nối không mất mát thông tin đối với tập phụ thuộc hàm trên S, vì $S_1 \cap S_2 = X$, $X \rightarrow S_1 - S_2 = A$.

Theo bổ đề 4.6 phần (a), ρ được thay S bằng S_1 và S_2 là không mất mát thông tin. Mỗi lược đồ S_1, S_2 đều có số thuộc tính ít hơn S.

Quá trình tiếp tục cho tới khi tất cả các lược đồ đều ở BCNF. Chú ý rằng, tại mọi thời điểm ρ luôn đảm bảo không mất mát thông tin, vì rằng ρ ban đầu là R, mà bước thay đổi ρ đều bảo toàn tính chất đó.

Ví dụ 4.19: cho lược đồ R(CTHRSG), trong đó C: Giáo trình, T: Thầy giáo, H: Giờ, R: Phòng học, S: Sinh viên, G: Lớp. Tập các phụ thuộc hàm F như sau:

$C \rightarrow T$: Mỗi giáo trình có một thầy dạy.

$HR \rightarrow X$: Chỉ một môn học (giáo trình) ở một phòng học tại một thời điểm.

$HT \rightarrow R$: Tại mỗi thời điểm mỗi thầy giáo chỉ có thể dạy ở một phòng học

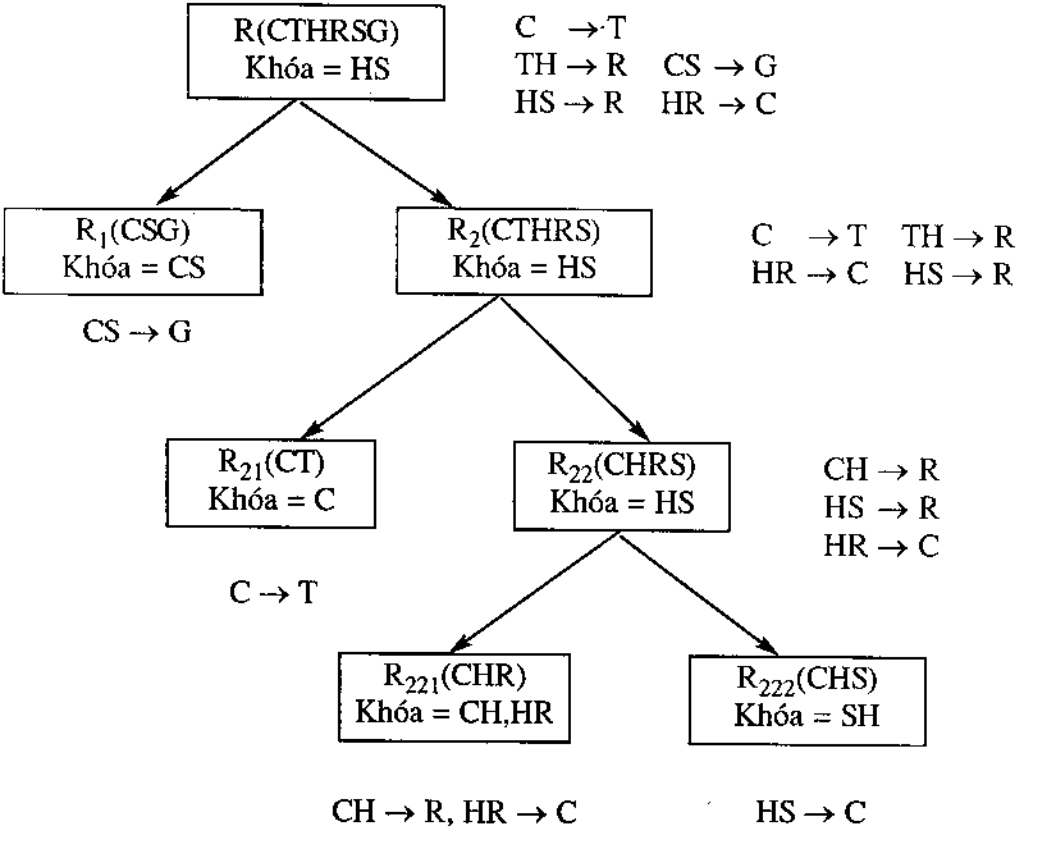
$CS \rightarrow G$: Mỗi sinh viên chỉ ở một lớp học trong mỗi giáo trình.

$HS \rightarrow R$: Mỗi sinh viên chỉ có thể ở một phòng học tại mỗi thời điểm.

Khoá của R là HS.

Tách lược đồ R thành các lược đồ BCNF.

Xét CS → G cho R vi phạm điều kiện BCNF vì CS không chứa khoá. Do vậy, dùng thuật toán 4.3 để tách R thành R_1(CSG) và R_2(CTHRS). Bước tiếp cần tính F^+ và chiếu xuống R_1 và R_2, sau đó kiểm tra các lược đồ đã ở BCNF chưa. Có thể biểu diễn quá trình tách qua sơ đồ sau:



Phép tách cuối cùng được R = (R_1, R_{21}, R_{221}, R_{222}).

2.2. Phép tách một lược đồ quan hệ thành 3NF có bảo toàn phụ thuộc hàm

Không phải lúc nào cũng tách được một lược đồ quan hệ thành dạng BCNF mà vẫn bảo toàn được các phụ thuộc hàm. Tuy nhiên luôn có thể tìm được phép tách bảo toàn phụ thuộc hàm thành dạng chuẩn thứ 3 như thuật toán sau:

Thuật toán 4.5: Tách một lược đồ quan hệ thành 3NF

- Vào: Lược đồ quan hệ R, tập các phụ thuộc hàm F (không làm mất tính tổng quát giả sử rằng nó là phủ tối thiểu).

- Ra: Phép tách của R bảo toàn các phụ thuộc hàm sao cho mỗi lược đồ con đều ở 3NF ứng với hình chiếu của F trên lược đồ đó.

* Phương pháp:

- Loại bỏ tất cả các thuộc tính của R nếu các thuộc tính đó không liên quan đến một phụ thuộc hàm nào của F (hoặc vế trái, hoặc vế phải). Về nguyên tắc, mọi thuộc tính như thế đều có thể tạo ta một lược đồ.

- Nếu có một phụ thuộc hàm nào của F mà liên quan tới tất cả các thuộc tính của R thì kết quả ra chính là R.

- Ngoài ra, phép tách ρ đưa ra các lược đồ gồm các thuộc tính XA cho phụ thuộc hàm $X \rightarrow A \in F$, nếu $X \rightarrow A_1$, $X \rightarrow A_2$,..., $X \rightarrow A_n \in F$ thì thay thế lược đồ $XA_1A_2... A_n$ cho các lược đồ $XA_i(1<= i <= n)$.

Ví dụ 4.20: Cho lược đồ quan hệ R(CTHRSG) với tập phụ thuộc hàm tối thiểu $C \rightarrow T$, $HR \rightarrow C$, $HT \rightarrow R$, $CS \rightarrow G$ và $HS \rightarrow R$ dùng cho thuật toán 4.4 cho tập lược đồ dạng chuẩn thứ 3:

$R_1(CT)$, $R_2(CHR)$, $R_3(HRT)$, $R_4(CGS)$ và $R_5(HRS)$.

Định lý 4.6

Thuật toán 4.4 tạo ra một phép tách bảo toàn các phụ thuộc hàm và các lược đồ đích ở dạng chuẩn 3.

2.3. Phép tách một lược đồ quan hệ thành 3NF có bảo toàn phụ thuộc hàm và kết nối không mất mát thông tin

Như đã biết, có thể tách một lược đồ quan hệ R bất kỳ thành một tập các lược đồ $\rho = (R_1, R_2,..., R_n)$ sao cho ρ có kết nối không mất mát thông tin và mỗi R_i có dạng BCNF (vì thế nó ở dạng 3NF). Ta cũng có thể tách R thành $\sigma = (S_1, S_2,. .., S_m)$ sao cho σ bảo toàn tập phụ thuộc hàm F và mỗi S_i có dạng 3NF. Liệu có thể tìm được một phép tách thành 3NF mà vẫn có cả hai đặc tính bảo toàn phụ thuộc hàm và kết nối không mất mát thông tin? Ta chỉ cần nối vào σ một lược đồ quan hệ X, trong đó X là khoá của R như định lý sau sẽ trình bày.

Định lý 4.7

Gọi σ là một phép tách của R thành 3NF được xây dựng bằng thuật toán 4.5 và X là khoá của R. Thế thì τ = σ ∪{X} là một phép tách của R mà tất cả các lược đồ quan hệ đều có dạng 3NF và phép tách này có đặc tính bảo toàn phụ thuộc hàm và kết nối không mất mát thông tin.

Chứng minh:

Dễ dàng chứng minh được với bất kỳ vi phạm 3NF nào trong X cũng khẳng định được một tập con thực sự của X xác định hàm X và như thế xác định được R, nên X không thể là khoá trong trường hợp này. Như vậy, X và các lược đồ trong σ phải có dạng 3NF. Rõ ràng τ bảo toàn phụ thuộc vì σ bảo toàn phụ thuộc

Để chứng minh τ có kết nối không mất mát thông tin, ta xây dựng các bảng và áp dụng thuật toán 4.2, và có thể chứng minh toàn bộ hàng của X sẽ trở thành a như sau:

Xét thứ tự $A_1, A_2,..., A_k$ là thứ tự khi thêm các thuộc tính của R - X vào X^+ trong thuật toán 4.1, chắc chắn rằng, cuối cùng thì tất cả các thuộc tính sẽ được thêm vào bởi vì X là khoá. Ta sẽ chứng minh bằng quy nạp trên i rằng cột tương ứng với A_i trong hàng của X được đặt bằng a trong phép kiểm tra của thuật toán 4.2.

Bước cơ sở i = 0 hiển nhiên đúng. Giả sử kết quả đúng với i - 1. Thế thì A_i được thêm vào X^+ do có một phụ thuộc hàm $Y \rightarrow A_i$ trong đó: $Y \subseteq X \cup \{A_1, A_2,..., A_{i-1}\}$. Thế thì YA_i thuộc σ và X giống nhau ở Y (tất cả đều là a) sau khi các cột cho $A_1, A_2,..., A_{i-1}$ ở hàng X được đặt là a. Vì vậy, các hàng này được biến đổi thành giống nhau ở thuộc tính A_i trong khi thực hiện thuật toán 4.2. Bởi vì hàng YA_i có ký hiệu a_i nên hàng X cũng thế.

Ví dụ 4.21:

Ta có thể lấy hợp của lược đồ CSDL của CTHRSG được tạo ra trong ví dụ 4.20 với khoá SH, thu được phép tách có kết nối không mất mát thông tin và bảo toàn phụ thuộc hàm. May mắn SH lại là tập con của HRS là một trong những lược đồ quan hệ đã có trong phép tách, vì thế có thể loại bỏ SH và chấp nhận lược đồ CSDL của ví dụ 4.20, nghĩa là (CT, CHR, THR, CSG, HRS).

Câu hỏi và bài tập chương 4

1. Nêu định nghĩa, tính chất phụ thuộc hàm F, bao đóng F^+ của tập phụ thuộc hàm F, bao đóng X^+ của tập thuộc tính X, cho ví dụ.

2. Nêu các tính chất của X^+, F^+.

3. Nêu định nghĩa khoá của lược đồ quan hệ, cho ví dụ.

4. Nêu thuật toán tìm X^+.

5. Nêu thuật toán tìm khoá, cho ví dụ.

6. Nêu các định nghĩa dạng chuẩn 1NF, 2NF, 3NF, cho ví dụ.

7. Trình bày phép tách SĐQH có kết nối không mất mát thông tin và thuật toán tách SĐQH có kết nối không mất tin về dạng BCNF, cho ví dụ.

8. Trình bày thuật toán tách SĐQH có kết nối không mất mát thông tin về dạng 3NF, cho ví dụ.

9. Trình bày định nghĩa phụ thuộc hàm đa trị, dạng chuẩn 4NF và thuật toán tách SĐQH có kết nối không mất mát thông tin về dạng 4NF, cho ví dụ.

10. Cho lược đồ quan hệ R và tập các phụ thuộc hàm:

a. $F = \{AB \rightarrow E, AG \rightarrow I, BE \rightarrow I, E \rightarrow G, GI \rightarrow H\}$ trên R

Chứng minh rằng: $AB \rightarrow GHI$.

b. $F = \{AB \rightarrow C, B \rightarrow D, CD \rightarrow E, CE \rightarrow GH, G \rightarrow A\}$ trên R

Chứng minh rằng: $AB \rightarrow E$ và $AB \rightarrow G$.

11. Cho sơ đồ quan hệ $W = <R, F>$ với $R = \{A, B, C, D\}$ và:

a. $F = \{A \rightarrow B, A \rightarrow C\}$. Phụ thuộc hàm nào trong dãy sau được suy dẫn từ F:

$A \rightarrow D$

$C \rightarrow D$

$BC \rightarrow A$

$A \rightarrow BC$

b. $F = \{A \rightarrow B, BC \rightarrow D\}$. Phụ thuộc hàm nào trong dãy sau được suy dẫn từ F:

$C \rightarrow D$

$A \rightarrow D$

$AD \rightarrow C$

$BC \rightarrow A$

$B \rightarrow CD$

12. Cho sơ đồ quan hệ W = <R, F>.

Các phép tách sau có kết nối mất mát thông tin không? Với:

a. R = {A, B, C} và:

F = {A → B}

P = (R_1, R_2), R_1 = {A, B}, R_2 = {A, C}.

b. R = {A, B, C} và:

F = {A → B, A → C }

P = (R_1, R_2), R_1 = {A, B}, R_2 = {A, C}.

13. Cho sơ đồ quan hệ W = <R, F>.

Các phép tách sau có kết nối mất mát thông tin không? Với:

a. R = {A, B, C, D, E, H, I, K, L} và

F = {AB → D, DE → H, IK → L, LB → C}

P = (R_1, R_2, R_3), R_1 = {A, B, C}, R_2 = {C, D, E, H},

R_3 = {E, H, I, K, L}.

b. R = (ABCDEF)

F = {AB → C, C → D, D → E, DE → F }

P = (R_1, R_2, R_3, R_4), R_1 = {A, B, C}, R_2 = {C, D}, R_3 = {D, E},

R_4 = {D, E, F}.

14. Hãy tìm khoá của sơ đồ quan hệ sau:

W = <R, F>

Trong đó:

R = (ABCDEF)

F = {AB → C, C → D, D → E, DE → F}

15. Hãy tìm các khoá của mỗi sơ đồ quan hệ sau:

a. W = < R, F>

Trong đó:

R = (ABCD)

F = {AB → C, D → B, D → B, C → ABD}

b. W = <R, F>

Trong đó:

R = (ABCDEG)

F = {AB → C, C → A, BC → D, ACD → B, D → EG, BE → C,

CG → BD, E → G}

16. Cho sơ đồ quan hệ sau:

a. W = <R, F>

Trong đó:

R = (ABCD)

F = {AB → C, D → B, C → ABD}

Và phép tách ρ = (ABC, BCD)

Hãy kiểm tra xem phép tách trên có kết nối mất mát thông tin không? Có bảo toàn phụ thuộc hàm không?

b. W = <R, F>

Trong đó:

R = (ABCDEG)

F = {AB → C, C → A, BC → D, ACD → B, D → EG, BE → C,

CG → BD, E → G}

Và phép tách ρ = (ABC, BCD, DEG)

Hãy kiểm tra xem phép tách trên có kết nối mất mát thông tin không? Có bảo toàn phụ thuộc hàm không?

17. Cho sơ đồ quan hệ W = <R, F>

Trong đó:

R = (ABCDE)

F = {AB → C, C → D, D → E}

Hãy tách thành các lược đồ quan hệ dạng BCNF có kết nối không mất mát thông tin; dạng 3NF có kết nối không mất mát thông tin và bảo toàn phụ thuộc hàm.

18. Cho sơ đồ quan hệ W = <R, F>. Trong đó:

R = (Masv, Mamon, Diem, Hoten, Ngaysinh, Gioi, Diachi, Quequan)

F = {Masv, Mamon → Diem, Masv → Hoten, Ngaysinh, Gioi, Diachi, Quequan}

R_1 = {Masv, Mamon, Diem}

R_2 = {Masv, Hoten, Ngaysinh, Gioi, Diachi, Quequan}

$\rho = (R_1, R_2)$

Phép tách trên có kết nối mất mát thông tin không? Có bảo toàn phụ thuộc hàm không?

19. Cho sơ đồ quan hệ QLCH = <R, F>.

Trong đó:

R = (Makh, Tenkh, Diachi, Gioitinh, Dienthoai, Mamh,Tenmh, Mau, Dongia, Dvtinh, Soluong, MuaBan, NgayMB).

X = {Tenkh, Diachi, Gioitinh, Dienthoai}

Y = {Tenmh, Mau, Dongia, Dvtinh}

Z = {Soluong, MuaBan, NgayMB}

MA = {Makh, Mamh}

F = { Makh → X, Mamh → Y, MA → Z}

a. Tìm khoá của lược đồ quan hệ R bằng định nghĩa khoá và bằng giải thuật tìm khoá.

b. Xét xem R có ở dạng chuẩn 2NF, 3NF, BCNF không? Tách R thành các lược đồ quan hệ dạng BCNF có kết nối không mất mát thông tin. 3NF bảo toàn phụ thuộc hàm và 3NF có kết nối không mất mát thông tin và bảo toàn phụ thuộc hàm.

c. Tìm khoá của mỗi lược đồ quan hệ vừa nhận được từ phép tách trên.

20. Cho sơ đồ quan hệ QLDiem = <R, F>.

R = (MaSV, TenSV, Diachi, Gioitinh, Lop, Mamon, Tenmon, SoHT, DiemthiL1, DiemthiL2, DiemthiL3).

X = {TenSV, Diachi, Gioitinh, Lop}

Y = {TenMon, SoHT}

Z = {DiemthiL1, DiemthiL2, DiemthiL3}

MA = {MaSV, MaMon}

F = {MaSV → X, MaMon → Y, MA → Z}

a. Tìm khoá của lược đồ quan hệ R bằng định nghĩa khoá và bằng giải thuật tìm khoá.

b. Tìm phủ tối thiểu F' của F.

c. Xét xem R có ở dạng chuẩn 2NF, 3NF, BCNF không?

d. Tách R thành các lược đồ quan hệ dạng BCNF có kết nối không mất mát thông tin. 3NF bảo toàn phụ thuộc hàm và 3NF có kết nối không mất mát thông tin và bảo toàn phụ thuộc hàm.

e. Tìm khoá của mỗi lược đồ quan hệ vừa nhận được từ phép tách trên.

TÀI LIỆU THAM KHẢO

1. ***Lý thuyết cơ sở dữ liệu,*** Đỗ Trung Tuấn, NXB Đại học Quốc gia, năm 2000.

2. ***Nguyên lý các hệ cơ sở dữ liệu và cơ sở tri thức*** *(Tập 1)* I.D. Ullman, bản dịch của Trần Đức Quang, NXB Thống kê, năm 2000.

3. ***Nhập môn cơ sở dữ liệu quan hệ,*** Lê Tiến Vương, NXB Thống kê, năm 2000.

4. ***Ngôn ngữ con cơ sở dữ liệu SQL,*** Ngô Trung Việt, NXB Giao thông vận tải, năm 1998.

5. ***Cơ sở dữ liệu - kiến thức và thực hành,*** Nguyễn Bá Tường, NXB Khoa học kỹ thuật, năm 2000.

6. ***Cơ sở dữ liệu - kiến thức và thực hành,*** Vũ Đức Thi, NXB Thống kê, năm 1997.

MỤC LỤC

NHÀ XUẤT BẢN HÀ NỘI
4 - TỐNG DUY TÂN, QUẬN HOÀN KIẾM, HÀ NỘI
ĐT: (04) 8252916; 8257063 - FAX (04) 8257063

GIÁO TRÌNH

CƠ SỞ DỮ LIỆU QUAN HỆ

NHÀ XUẤT BẢN HÀ NỘI - 2005

Chịu trách nhiệm xuất bản:
NGUYỄN KHẮC OÁNH

Biên tập:
TRƯƠNG ĐỨC HÙNG

Bìa:
TRẦN QUANG

Trình bày, kỹ thuật vi tính:
HOÀNG THÚY LƯƠNG

Sửa bản in:
LÊ XUÂN THỌ

In 1550 cuốn, khổ 17 x 24cm, tại Nhà in Hà Nội.
Giấy phép xuất bản số: 114GT/407 CXB ngày 29/3/2005
In xong và nộp lưu chiểu tháng 7 năm 2005.

BỘ GIÁO TRÌNH XUẤT BẢN NĂM 2005
KHỐI TRƯỜNG TRUNG HỌC KINH TẾ KỸ THUẬT TIN HỌC

1. THUẬT TOÁN VÀ KỸ THUẬT LẬP TRÌNH PASCAL
2. ĐÁNH MÁY VI TÍNH
3. SOẠN THẢO VÀ ĐÁNH MÁY VĂN BẢN
4. NGHIỆP VỤ THƯ KÝ
5. KẾ TOÁN MÁY
6. MARKETING
7. NGÔN NGỮ LẬP TRÌNH C
8. LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG VỚI (C++)
9. BẢNG TÍNH ĐIỆN TỬ (EXCEL)
10. VISUAL BASIC
11. CẤU TRÚC MÁY TÍNH
12. GIAO TIẾP
13. ACCESS
14. MẠNG MÁY TÍNH
15. THIẾT KẾ WEB
16. BẢO TRÌ PC
17. HỆ ĐIỀU HÀNH
18. CƠ SỞ DỮ LIỆU
19. PHÂN TÍCH THIẾT KẾ HỆ THỐNG
20. KỸ THUẬT SỐ
21. PHOTOSHOP
22. CAD/CAM

Giá: 13.000đ