



ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

GIÁO TRÌNH

THIẾT KẾ CƠ SỞ DỮ LIỆU

Biên soạn: Trịnh Minh Tuấn



NHÀ XUẤT BẢN
ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH

CHƯƠNG I

CÁC KHÁI NIỆM CỦA MỘT CƠ SỞ DỮ LIỆU

1.1. Dẫn nhập - Tại sao cần phải có một CSDL

Trong những năm gần đây, thuật ngữ "CƠ SỞ DỮ LIỆU" (Tiếng Anh là *DataBase*, viết tắt tiếng Việt là CSDL) đã trở nên khá quen thuộc không chỉ riêng với những người làm Tin học mà còn đối với cả những người làm trong nhiều lĩnh vực khác như Thống kê, Kinh tế, Quản lý Doanh nghiệp v.v... Các ứng dụng của Tin học vào công tác quản lý ngày càng nhiều hơn và càng đa dạng hơn. Có thể nói hầu hết các lĩnh vực kinh tế, xã hội, giáo dục, y tế v.v... đều đã ứng dụng các thành tựu mới của Tin học vào phục vụ công tác chuyên môn của mình. Chính vì lẽ đó mà ngày càng nhiều người quan tâm đến lĩnh vực thiết kế và xây dựng các CSDL.

Mục đích của chương I chỉ đơn giản là cung cấp các khái niệm cơ bản về CSDL để các học viên có một cái nhìn ban đầu về một cơ sở dữ liệu và một hệ quản trị CSDL. Trước hết chúng ta sẽ tìm hiểu lý do tại sao cần phải có một CSDL.

* Hệ thống các tập tin cổ điển (File System)

Cho đến nay vẫn còn một số đơn vị kinh tế, hành chính sự nghiệp v.v... sử dụng mô hình hệ thống các tập tin cổ điển: chúng được tổ chức riêng rẽ, phục vụ cho một mục đích của một đơn vị hay một đơn vị con trực thuộc cụ thể. Chẳng hạn, hãy xét ví dụ sau:

Ví dụ 1.1:

Tại một công ty người ta trang bị máy vi tính cho tất cả các phòng, ban nghiệp vụ. Bộ phận Văn phòng sử dụng máy tính để soạn

thảo các văn bản báo cáo bằng MicroSoft Word do thủ trưởng yêu cầu về tình hình hoạt động của đơn vị trong đó có chỉ tiêu về tổng số công nhân viên chức chia theo trình độ chuyên môn được đào tạo. Phòng Kế toán sử dụng máy tính để tính lương và in danh sách lương của từng bộ phận trong đơn vị dựa trên danh sách cán bộ viên chức cùng hệ số lương và các hệ số phụ cấp của họ do phòng Tổ chức cung cấp. Thông tin mà phòng Kế toán quản lý và khai thác là: Họ và Tên, Hệ số lương, Hệ số phụ cấp, Phụ cấp khác của các công nhân viên chức (CNVC) xếp theo từng phòng ban và sử dụng công cụ văn phòng là MicroSoft Excel. Phòng Tổ chức quản lý thông tin lịch của CNVC chi tiết hơn gồm Họ CNVC, Tên CNVC (để riêng thành một cột "Tên" để tiện sắp xếp theo vần Alphabet), Bí danh, Giới tính, Ngày sinh, Ngày tuyển dụng, Hoàn cảnh gia đình, Quá trình được đào tạo, Hệ số lương, Hệ số phụ cấp, Ngày xếp lương trên... nhưng thiếu thông tin về Phục cấp khác của CNVC. Phần mềm được sử dụng để quản lý là FoxPro for Windows.

Trong khi đó, tại Tổng công ty của họ, các phòng ban nghiệp vụ cũng được trang bị vi tính. Phòng Tổ chức cán bộ tại Tổng công ty sử dụng phần mềm MicroSoft Access để quản lý CNVC gồm các cán bộ chủ chốt từ trưởng phó phòng, quản đốc và phó quản đốc xí nghiệp trở lên của các công ty con trực thuộc. Thông tin quản lý tại đây cũng giống như thông tin quản lý tại phòng tổ chức của công ty con.

* Nhận xét *:

Ưu điểm:

- Việc xây dựng hệ thống các tập tin riêng tại từng đơn vị quản lý ít tốn thời gian bởi khối lượng thông tin cần quản lý và khai thác là nhỏ, không đòi hỏi đầu tư vật chất và chất xám nhiều, do đó triển khai ứng dụng nhanh.

- Thông tin được khai thác chỉ phục vụ cho mục đích hép nêん khả năng đáp ứng nhanh chóng, kịp thời.

Nhược điểm:

- Do thông tin được tổ chức ở mỗi phòng ban mỗi khác, cũng như phần mềm công cụ để triển khai mỗi nơi cũng rất khác nhau nên sự phối hợp tổ chức và khai thác ở các phòng ban là khó khăn. Thông tin ở phòng ban này không sử dụng được cho phòng ban khác, tại đơn vị con với đơn vị cấp trên. Cùng một thông tin được nhập vào máy tại nhiều nơi khác nhau gây ra lãng phí công sức nhập tin và không gian lưu trữ trên các vật mang tin. Sự trùng lắp thông tin có thể dẫn đến tình trạng không nhất quán dữ liệu. Chẳng hạn, nhân viên Nguyễn Văn Quang được ghi đầy đủ ở phòng Tổ chức, nhưng tại phòng Kế toán chỉ ghi tắt là Nguyễn v Quang.

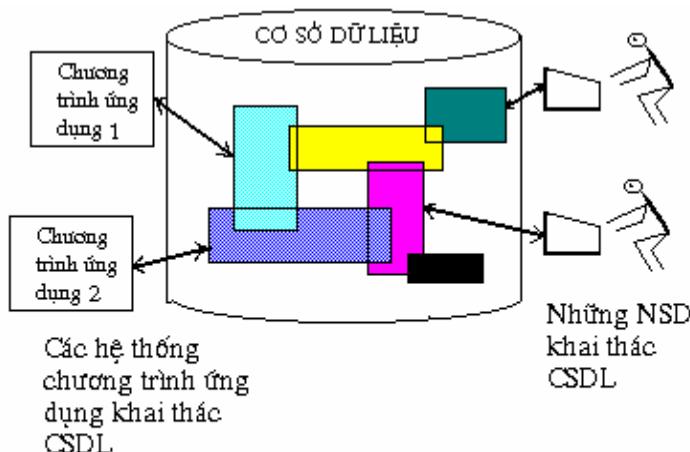
- Thông tin được tổ chức ở nhiều nơi nên việc cập nhật cũng dễ làm mất tính nhất quán dữ liệu. Một cán bộ chủ chốt của công ty có thay đổi về hoàn cảnh gia đình (mới cưới vợ / lấy chồng, sinh thêm con ...) có thể được cập nhật ngay tại đơn vị nhưng sau một thời gian mới được cập nhật tại Tổng công ty.

- Do hệ thống được tổ chức thành các hệ thống file riêng lẻ nên thiếu sự chia sẻ thông tin giữa các nơi. Việc kết nối các hệ thống này hay việc nâng cấp ứng dụng sẽ là rất khó khăn.

Qua phân tích trên chúng ta nhận thấy việc tổ chức dữ liệu theo hệ thống các tập tin có nhiều nhược điểm. Việc xây dựng một hệ thống tin đảm bảo được tính chất nhất quán dữ liệu, không trùng lắp thông tin mà vẫn đáp ứng được nhu cầu khai thác đồng thời của tất cả các phòng ban ở Công ty và Tổng Công ty là thực sự cần thiết.

1.2. Định nghĩa một CSDL.

Cơ sở dữ liệu là một hệ thống các thông tin có cấu trúc được lưu trữ trên các thiết bị lưu trữ thông tin thứ cấp (như băng từ, đĩa từ...) để có thể thỏa mãn yêu cầu khai thác thông tin đồng thời của nhiều người sử dụng hay nhiều chương trình ứng dụng với nhiều mục đích khác nhau.



Hình 1.2.1 Sơ đồ tổng quát về một cơ sở dữ liệu

Trong định nghĩa này cần nhấn mạnh những khía cạnh của định nghĩa được lưu ý qua các từ gạch chân. Trước hết, CSDL phải là một tập hợp các thông tin mang tính hệ thống chứ không phải là các thông tin rời rạc, không có mối quan hệ với nhau. Các thông tin này phải có cấu trúc và tập hợp các thông tin này phải có khả năng đáp ứng các nhu cầu khai thác của nhiều người sử dụng một cách đồng thời. Đó cũng chính là các đặc trưng của CSDL.

Rõ ràng, ưu điểm nổi bật của CSDL là:

- Giảm sự trùng lặp thông tin xuống mức thấp nhất và do đó bảo đảm được tính nhất quán và toàn vẹn dữ liệu.

- Đảm bảo dữ liệu có thể được truy xuất theo nhiều cách khác nhau.
- Khả năng chia sẻ thông tin cho nhiều người sử dụng và nhiều ứng dụng khác nhau.

Tuy nhiên, để đạt được các ưu điểm trên, CSDL đặt ra những vấn đề cần phải giải quyết. Đó là:

1- Tính chủ quyền của dữ liệu. Do tính chia sẻ của CSDL nên tính chủ quyền của dữ liệu có thể bị lu mờ và làm mờ nhạt tinh thần trách nhiệm, được thể hiện trên vấn đề an toàn dữ liệu, khả năng biểu diễn các mối liên hệ ngữ nghĩa của dữ liệu, và tính chính xác của dữ liệu. Điều này có nghĩa là người khai thác CSDL phải có nghĩa vụ cập nhật các thông tin mới nhất của CSDL.

2- Tính bảo mật và quyền khai thác thông tin của người sử dụng. Do có nhiều người được phép khai thác CSDL một cách đồng thời nên cần phải có một cơ chế bảo mật và phân quyền hạn khai thác CSDL. Các hệ điều hành nhiều người sử dụng hay hệ điều hành mạng cục bộ (Novell Netware, Windows For WorkGroup, WinNT, ...) đều có cung cấp cơ chế này.

3- Tranh chấp dữ liệu. Nhiều người được phép truy nhập vào cùng một tài nguyên dữ liệu (*Data Source*) của CSDL với những mục đích khác nhau: Xem, thêm, xóa hoặc sửa dữ liệu. Cần phải có một cơ chế ưu tiên truy nhập dữ liệu cũng như cơ chế giải quyết tình trạng khóa chết (DeadLock) trong quá trình khai thác cạnh tranh. Cơ chế ưu tiên có thể được thực hiện bằng việc cấp quyền (hay mức độ) ưu tiên cho từng người khai thác - người nào được cấp quyền hạn ưu tiên cao hơn thì được ưu tiên truy nhập dữ liệu trước; theo biến có hoặc loại truy nhập - quyền đọc được ưu tiên trước quyền ghi dữ liệu; dựa trên thời điểm truy nhập - ai có yêu cầu truy xuất trước thì có quyền truy nhập dữ liệu trước; hoặc theo cơ chế lập lịch truy xuất hay các cơ chế khóa [7]...

4- Đảm bảo dữ liệu khi có sự cố. Việc quản lý dữ liệu tập trung có thể làm tăng khả năng mất mát hoặc sai lệch thông tin khi có sự cố như mất điện đột xuất, một phần đĩa lưu trữ CSDL bị hư v.v... Một số hệ điều hành mạng có cung cấp dịch vụ sao lưu ảnh đĩa cứng (cơ chế sử dụng đĩa cứng dự phòng - RAID), tự động kiểm tra và khắc phục lỗi khi có sự cố, tuy nhiên, bên cạnh dịch vụ của hệ điều hành, để đảm bảo CSDL luôn luôn ổn định, một CSDL nhất thiết phải có một cơ chế khôi phục dữ liệu khi các sự cố bất ngờ xảy ra.

1.3. Các đối tượng sử dụng CSDL:

- Những người sử dụng CSDL không chuyên về lĩnh vực tin học và CSDL, do đó CSDL cần có các công cụ để cho những người sử dụng không chuyên có thể sử dụng để khai thác CSDL khi cần thiết.
- Các chuyên viên tin học biết khai thác CSDL. Những người này có thể xây dựng các ứng dụng khác nhau phục vụ cho các mục đích khác nhau trên CSDL.
- Những người quản trị CSDL, đó là những người hiểu biết về tin học, về các hệ quản trị CSDL và hệ thống máy tính. Họ là người tổ chức CSDL (khai báo cấu trúc CSDL, ghi nhận các yêu cầu bảo mật cho các dữ liệu cần bảo vệ ...) do đó họ phải nắm rõ các vấn đề kỹ thuật về CSDL để có thể phục hồi dữ liệu khi có sự cố. Họ là những người cấp quyền hạn khai thác CSDL, do vậy họ có thể giải quyết được các vấn đề tranh chấp dữ liệu, nếu có.

1.4. Hệ phần mềm quản trị CSDL.

Để giải quyết tốt tất cả các vấn đề đặt ra cho một CSDL như đã nêu trên: tính chủ quyền, cơ chế bảo mật hay phân quyền hạn khai thác CSDL, giải quyết tranh chấp trong quá trình truy nhập dữ

liệu, và phục hồi dữ liệu khi có sự cố ... thì cần phải có một hệ thống các phần mềm chuyên dụng. Hệ thống các phần mềm đó được gọi là hệ quản trị CSDL (tiếng Anh là DataBase Management System - DBMS). Đó là các công cụ hỗ trợ tích cực cho các nhà phân tích & thiết kế CSDL và những người khai thác CSDL. Cho đến nay có khá nhiều hệ quản trị CSDL mạnh mẽ được đưa ra thị trường như: Visual FoxPro, MicroSoft Access, SQL-Server, DB2, Sybase, Paradox, Informix, Oracle... với các chất lượng khác nhau.

Mỗi hệ quản trị CSDL đều được cài đặt dựa trên một mô hình dữ liệu cụ thể. Hầu hết các hệ quản trị CSDL hiện nay đều dựa trên mô hình quan hệ (Xem chương III). Dù dựa trên mô hình dữ liệu nào, một hệ quản trị CSDL cũng phải có:

- Ngôn ngữ giao tiếp giữa người sử dụng (NSD) và CSDL, bao gồm:
 - Ngôn ngữ mô tả dữ liệu (*Data Definition Language - DDL*) để cho phép khai báo cấu trúc của CSDL, khai báo các mối liên hệ của dữ liệu (*Data Relationship*) và các quy tắc (*Rules, Constraint*) quản lý áp đặt lên các dữ liệu đó.
 - Ngôn ngữ thao tác dữ liệu (*Data Manipulation Language - DML*) cho phép người sử dụng có thể thêm (*Insert*), xóa (*Delete*), sửa (*Update*) dữ liệu trong CSDL.
 - Ngôn ngữ truy vấn dữ liệu, hay ngôn ngữ hỏi đáp có cấu trúc (*Structured Query Language - SQL*) cho phép những người khai thác CSDL (chuyên nghiệp hoặc không chuyên) sử dụng để truy vấn các thông tin cần thiết trong CSDL.
 - Ngôn ngữ quản lý dữ liệu (*Data Control Language - DCL*) cho phép những người quản trị hệ thống thay đổi cấu trúc của các bảng dữ liệu, khai báo bảo mật thông tin và cấp quyền hạn khai thác CSDL cho người sử dụng.

- Từ điển dữ liệu (*Data Dictionary*) dùng để mô tả các ánh xạ liên kết, ghi nhận các thành phần cấu trúc của CSDL, các chương trình ứng dụng, mật mã, quyền hạn sử dụng v.v....

- Có biện pháp bảo mật tốt khi có yêu cầu bảo mật.

- Cơ chế giải quyết vấn đề tranh chấp dữ liệu. Mỗi hệ quản trị CSDL cũng có thể cài đặt một cơ chế riêng để giải quyết các vấn đề này. Một số biện pháp sau đây được sử dụng:

- Cấp quyền ưu tiên cho từng người sử dụng (người quản trị CSDL thực hiện).
- Đánh dấu yêu cầu truy xuất dữ liệu, phân chia thời gian, người nào có yêu cầu trước thì có quyền truy xuất dữ liệu trước.

- Hệ quản trị CSDL cũng phải có cơ chế sao lưu (Backup) và phục hồi (Restore) dữ liệu khi có sự cố xảy ra. Điều này có thể được thực hiện bằng cách:

- Định kỳ kiểm tra CSDL, sau một thời gian nhất định hệ quản trị CSDL sẽ tự động tạo ra một bản sao CSDL. Cách này hơi tốn kém, nhất là đối với các CSDL lớn.
- Tạo nhật ký (*LOG*) thao tác CSDL. Mỗi thao tác trên CSDL đều được hệ thống ghi lại, khi có sự cố xảy ra thì tự động lần ngược lại (*RollBack*) để phục hồi CSDL.

- Hệ quản trị CSDL phải cung cấp một giao diện (Interface) tốt, dễ sử dụng, dễ hiểu cho những người sử dụng không chuyên.

- Ngoài ra, một hệ quản trị CSDL phải đáp ứng được một yêu cầu rất quan trọng, đó là bảo đảm tính độc lập giữa dữ liệu và chương trình: Khi có sự thay đổi dữ liệu (như sửa đổi cấu lưu trữ các bảng dữ liệu, thêm các chỉ mục (Index) ...) thì các chương trình ứng dụng (Application) đang chạy trên CSDL đó vẫn không cần phải được viết lại, hay cũng không làm ảnh hưởng đến những NSD khác.

* Vài nét về quá trình phát triển các hệ quản trị CSDL:

Trải qua gần 40 năm nghiên cứu và cài đặt ứng dụng, các hệ quản trị CSDL không ngừng được phát triển. Các hệ quản trị CSDL đầu tiên ra đời vào đầu những năm 60 của thế kỷ 20 dựa trên mô hình dữ liệu phân cấp và mạng, trong số đó có hệ quản trị CSDL có tên là IMS của hãng IBM dựa trên mô hình dữ liệu phân cấp.

Năm 1976, hệ quản trị CSDL đầu tiên dựa trên mô hình dữ liệu quan hệ của hãng IBM mang tên System-R ra đời. Từ năm 1980 hãng IBM cho ra đời hệ quản trị CSDL trên các máy Main Frame mang tên DB2, tiếp theo là các hệ quản trị CSDL Dbase, Sybase, Oracle, Informix, SQL-Server ...

Từ những năm 1990 người ta bắt đầu cố gắng xây dựng các hệ quản trị CSDL hướng đối tượng (*Oriented Object DataBase Management System*) như Orion, Illustra, Itasca, ... Tuy nhiên hầu hết các hệ này đều vẫn là quan hệ - hướng đối tượng, nghĩa là, xét về bản chất, chúng vẫn dựa trên nền tảng của mô hình quan hệ. Hệ quản trị CSDL hướng đối tượng thuần nhất có thể là hệ ODMG ra đời vào năm 1996.

1.5. Các mức biểu diễn một CSDL.

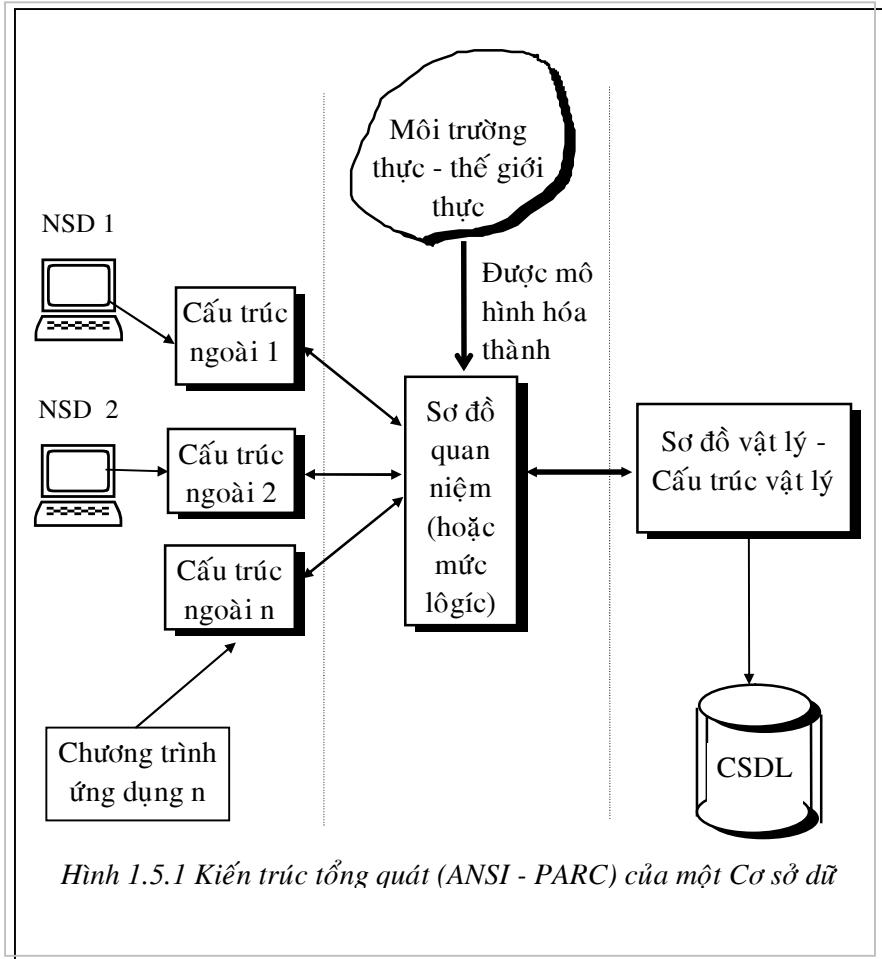
Theo kiến trúc ANSI-PARC, một CSDL có 3 mức biểu diễn: Mức trong (còn gọi là mức vật lý - *Physical*), mức quan niệm (*Conception* hay *Logical*) và mức ngoài – Xem hình 1.5.1.

1.5.1. Mức trong:

Đây là mức lưu trữ CSDL. Tại mức này, vấn đề cần giải quyết là, dữ liệu gì và được lưu trữ như thế nào? ở đâu (đĩa từ, băng từ,

track, sector ... nào)? Cần các chỉ mục gì? Việc truy xuất là tuần tự (*Sequential Access*) hay ngẫu nhiên (*Random Access*) đối với từng loại dữ liệu.

Những người hiểu và làm việc với CSDL tại mức này là người quản trị CSDL (*Administrator*), những người sử dụng (NSD) chuyên môn.



Hình 1.5.1 Kiến trúc tổng quát (ANSI - PARC) của một Cơ sở dữ

1.5.2. Mức quan niệm:

Tại mức này sẽ giải quyết cho câu hỏi CSDL cần phải lưu giữ bao nhiêu loại dữ liệu? Đó là những dữ liệu gì? Mỗi quan hệ giữa các loại dữ liệu này như thế nào?

Từ thế giới thực (*Real Universe*) các chuyên viên tin học qua quá trình khảo sát và phân tích, cùng với những người sẽ đảm nhận vai trò quản trị CSDL, sẽ xác định được những loại thông tin gì được cho là cần thiết phải đưa vào CSDL, đồng thời mô tả rõ mối liên hệ giữa các thông tin này. Có thể nói cách khác, CSDL mức quan niệm là một sự biểu diễn trừu tượng CSDL mức vật lý; hoặc ngược lại, CSDL vật lý là sự cài đặt cụ thể của CSDL mức quan niệm.

Ví dụ 1.2:

Người ta muốn xây dựng một hệ quản trị CSDL để quản lý các nhân viên của một công ty. Mỗi trướng (*thế giới thực*) của công ty ở đây gồm có các phòng ban (*Department*) - mỗi phòng ban có một tên gọi khác nhau, một địa chỉ trụ sở chính (*Location*), các số điện thoại (*Telephone*) để liên lạc, có một người làm trưởng phòng ban, hàng năm được cấp một khoản kinh phí để hoạt động (*Expense Budget*), và phải đạt một doanh thu (*Revenue Budget*). Để tránh viết tên phòng ban dài dãy đến viết sai, người ta thường đặt cho mỗi phòng ban một giá trị số (gọi là số hiệu phòng ban - *Department Number*) và sử dụng số hiệu này để xác định tên và các thông tin khác của nó.

Công ty có một số công việc có thể sắp xếp cho các nhân viên trong công ty. Để thuận lợi cho việc theo dõi công việc cũng như trong công tác tuyển chọn nhân viên mới, người ta lập thành một bảng các công việc (*JOBS*) gồm các thông tin: tên tắt công việc (*Job*), tên công việc (*Job Name*), mức lương tối thiểu (*Min Salary*) và tối đa (*Max Salary*) của công việc này và cho biết công việc này cần có

người lãnh đạo không. Một công việc có thể có nhiều người cùng làm.

Mỗi phòng ban có thể có từ 1 đến nhiều nhân viên (*Employee*). Mỗi nhân viên có một tên gọi, một công việc làm (Job), một khoản tiền lương hàng tháng (Salary), số hiệu phòng ban mà anh ta đang công tác. Nếu muốn, người ta có thể theo dõi thêm các thông tin khác như ngày sinh (*Birth Day*), giới tính (*Sex*) v.v... Để tránh viết tên nhân viên dài dãy dẫn đến sai sót, mỗi nhân viên có thể được gán cho một con số duy nhất, gọi là mã số nhân viên (*EmpNo*).

Nếu yêu cầu quản lý của công ty chỉ dừng ở việc theo dõi danh sách nhân viên trong từng phòng ban cùng các công việc của công ty thì cần 3 loại thông tin: Phòng ban (*DEPARTMENT*), Công việc (*JOBS*) và Nhân viên (*EMPLOYEE*) với các thông tin như trên là đủ. Có thể công ty có thêm yêu cầu quản lý cả quá trình tuyển dụng và nâng lương thì cần có thêm một (hoặc một số) loại thông tin về quá trình: Mã số nhân viên, lần thay đổi, thời gian bắt đầu và kết thúc sự thay đổi, mức lương, .v.v...

Từ môi trường thế giới thực, xuất phát từ nhu cầu quản lý, việc xác định các loại thông tin cần lưu trữ và các mối quan hệ giữa các thông tin đó như thế nào ... đó chính là công việc ở mức quan niệm.

1.5.3. Mức ngoài.

Đó là mức của người sử dụng và các chương trình ứng dụng. Làm việc tại mức này có các nhà chuyên môn, các kỹ sư tin học và những người sử dụng không chuyên.

Mỗi người sử dụng hay mỗi chương trình ứng dụng có thể được "nhìn" (View) CSDL theo một góc độ khác nhau. Có thể "nhìn" thấy toàn bộ hay chỉ một phần hoặc chỉ là các thông tin tổng hợp từ CSDL hiện có. Người sử dụng hay chương trình ứng dụng có thể hoàn toàn

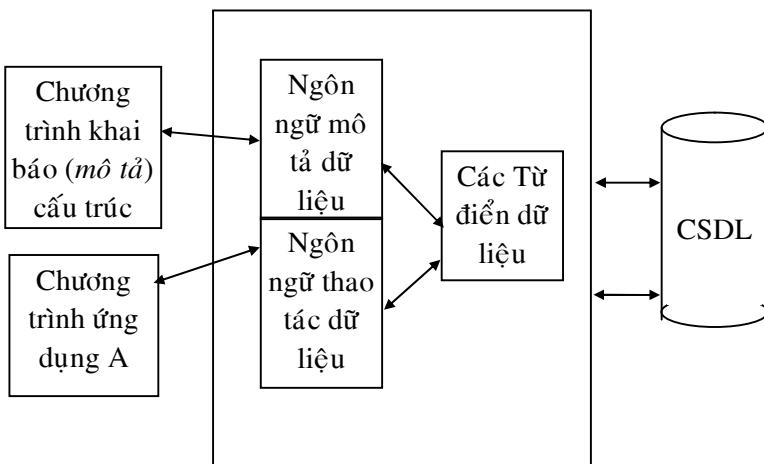
không được biết về cấu trúc tổ chức lưu trữ thông tin trong CSDL, thậm chí ngay cả tên gọi của các loại dữ liệu hay tên gọi của các thuộc tính. Họ chỉ có thể làm việc trên một phần CSDL theo cách "nhìn" do người quản trị hay chương trình ứng dụng quy định, gọi là khung nhìn (View).

Ví dụ 1.3:

Cũng ví dụ trên, Phòng Tổ chức nhân sự giờ đây còn quản lý thêm cả các thông tin chi tiết trong lý lịch của nhân viên trong công ty: quá trình đào tạo chuyên môn kỹ thuật - kinh tế - chính trị - quản lý Nhà nước, quá trình được khen thưởng, các lần bị kỷ luật, quá trình hoạt động Cách mạng bị địch bắt - bị tù đày, quá trình công tác, quá trình nâng lương, số lực lượng tiểu sử cha mẹ - anh chị em ruột - vợ chồng - con v.v... Rõ ràng rằng, Phòng Kế toán có thể chỉ được nhìn thấy CSDL là danh sách nhân viên đang làm các công việc cụ thể trong từng Phòng ban với các mức lương thỏa thuận, mà không được thấy lý lịch của các nhân viên. Lãnh đạo công ty có thể chỉ cần "nhìn" thấy số lượng nhân viên, tổng số lương phải trả và ai là người lãnh đạo của từng Phòng ban. Trong khi đó ngay cả những người trong Phòng Tổ chức nhân sự cũng có thể có người được xem lý lịch của tất cả cán bộ, công nhân viên của công ty, nhưng cũng có thể có người chỉ được xem lý lịch của những cán bộ, công nhân viên với mức lương từ xx đồng trở xuống...

Như vậy, cấu trúc CSDL vật lý (mức trong) và mức quan niệm thì chỉ có một; nhưng tại mức ngoài, mức của các chương trình ứng dụng và người sử dụng trực tiếp CSDL, thì có thể có rất nhiều cấu trúc ngoài tương ứng.

1.6. Sơ đồ tổng quát của một hệ quản trị CSDL



Hình 1.6.1. Sơ đồ tổng quát của một hệ quản trị CSDL

Hình 1.6.1 minh họa sơ đồ tổng quát của một hệ quản trị CSDL. Chúng ta thấy có 3 mức: mức chương trình khai báo cấu trúc và chương trình ứng dụng; mức mô tả CSDL, thao tác CSDL và các từ điển dữ liệu; và mức CSDL.

Mỗi hệ quản trị CSDL có một ngôn ngữ khai báo (hay mô tả: *Data Definition Language - DDL*) cấu trúc CSDL riêng. Những người thiết kế và quản trị CSDL thực hiện các công việc khai báo cấu trúc CSDL.

Các chương trình khai báo cấu trúc CSDL được viết bằng ngôn ngữ mà hệ quản trị CSDL cho phép. Hai công việc khai báo là khai báo cấu trúc lôgic (đó là việc khai báo các loại dữ liệu và các mối liên hệ giữa các loại dữ liệu đó, cùng các ràng buộc toàn vẹn dữ liệu - RBTV) và khai báo vật lý (dữ liệu được lưu trữ theo dạng nào?, có bao nhiêu chỉ mục?).

Các chương trình ứng dụng được viết bằng ngôn ngữ thao tác CSDL (*Data Manipulation Language - DML*) với mục đích:

- Truy xuất dữ liệu
- Cập nhật dữ liệu (thêm, xóa, sửa dữ liệu)
- Khai thác dữ liệu
- Ngôn ngữ thao tác CSDL còn được sử dụng cho những NSD thao tác trực tiếp với CSDL.

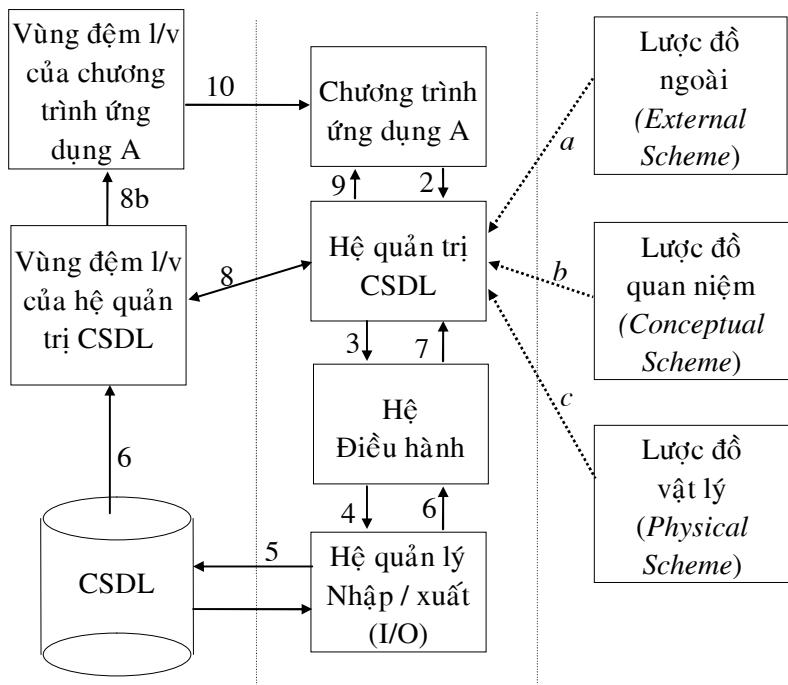
Từ điển dữ liệu (*Data Dictionary - DD*) là một CSDL của hệ quản trị CSDL sử dụng để lưu trữ cấu trúc CSDL, các thông tin bảo mật, bảo đảm an toàn dữ liệu và các cấu trúc ngoài. Những người đã làm quen với hệ quản trị CSDL của MicroSoft Access có thể thấy các từ điển dữ liệu này thông qua các bảng (*Table*) có tên bắt đầu bằng chữ MSys như MSysACEs, MSysColumn, MSysIMEXColumn, MSysIMEXSpecs, MSysIndexes, MSysMacros, MSysObjects, MSysQueries, MSysRelationships ... Từ điển dữ liệu còn được gọi là Siêu CSDL (*Meta-DataBase*).

(*) Quá trình hoạt động của một chương trình ứng dụng thông qua các tầng của CSDL:

Hình 1.6.2 cho chúng ta một cách nhìn về quá trình hoạt động của một chương trình ứng dụng thông qua các tầng của CSDL:

Các yêu cầu của chương trình ứng dụng được chuyển tới hệ quản trị CSDL (theo con đường số 2). Tại đây hệ quản trị CSDL sẽ tham khảo các từ điển dữ liệu (*Meta DataBase*) để tìm kiếm các ánh xạ cấu trúc ngoài với cấu trúc quan niệm và cấu trúc vật lý (các ngõ a, b và c). Tại đây hệ quản trị CSDL có thể sẽ tham khảo tới vùng đệm của nó để xác định xem câu trả lời đã có sẵn ở đó chưa, nếu có thì trả lại cho chương trình ứng dụng thông qua con đường số 9; ngược lại sẽ yêu cầu hệ điều hành truy xuất thông tin theo con đường số 3. Tới đây hệ điều hành sẽ gửi yêu cầu truy xuất thông tin trong CSDL

thông qua hệ thống xuất nhập của HĐH (các con đường số 4 và 5). Nếu việc truy xuất không thành công nó sẽ trả lại yêu cầu về cho hệ quản trị CSDL (có thể thông qua các mã lỗi) qua con đường số 6; nếu thành công thì dữ liệu sẽ được chuyển vào vùng đệm của hệ quản trị CSDL. Qua xử lý, hệ quản trị CSDL sẽ chuyển dữ liệu vào vùng đệm của chương trình ứng dụng để nó xử lý (qua con đường 8a) và cho ra kết quả trả lời của chương trình ứng dụng qua con đường số 10.



Hình 1.6.2. Quá trình hoạt động của một chương trình ứng dụng thông qua các tầng của CSDL.

Theo sơ đồ trên có thể nhận thấy các trực trặc có thể xảy ra tại các con đường (2a), (3), (4), (5), (6) và (8). Lỗi tại 2 con đường số (6) và (8) có thể là do tràn vùng làm việc.

1.7. Tính độc lập giữa dữ liệu và chương trình.

Lược đồ khái niệm là sự biểu diễn thế giới thực bằng một loại ngôn ngữ phù hợp của hệ quản trị CSDL. Qua hình 1.5.1 - Sơ đồ tổng quát của một CSDL theo kiến trúc ANSI - PARC, chúng ta có thể thấy, từ chương trình ứng dụng và người khai thác trực tiếp CSDL thông qua một khung nhìn tới CSDL (*View*) tồn tại hai mức độc lập dữ liệu. Thứ nhất, lược đồ vật lý có thể thay đổi do người quản trị CSDL mà hoàn toàn không làm thay đổi các lược đồ con. Người quản trị CSDL có thể tổ chức lại CSDL bằng cách thay đổi cách tổ chức, cấu trúc vật lý của dữ liệu trên các thiết bị nhớ thứ cấp để làm thay đổi hiệu quả tính toán của các chương trình ứng dụng, nhưng không đòi hỏi phải viết lại các chương trình ứng dụng. Điều này được gọi là tính độc lập vật lý của dữ liệu - hay tính độc lập của dữ liệu ở mức vật lý (*Physical Independence*). Tính độc lập dữ liệu mức vật lý được đảm bảo tới mức nào còn phụ thuộc vào chất lượng của hệ quản trị CSDL.

Thứ hai, giữa khung nhìn với lược đồ quan niệm cũng có thể tồn tại một loại độc lập về dữ liệu. Trong quá trình khai thác CSDL người ta có thể nhận thấy tính cần thiết phải sửa đổi lược đồ khái niệm như bổ sung thêm thông tin hoặc xóa bỏ các thông tin của các thực thể đang tồn tại trong CSDL. Việc thay đổi lược đồ khái niệm không làm ảnh hưởng tới các lược đồ con, do đó không cần phải viết lại các chương trình ứng dụng. Tính chất độc lập này được gọi là tính độc lập của dữ liệu ở mức lôgic (*Logical Independence*).

Tính độc lập giữa dữ liệu với chương trình ứng dụng là mục tiêu chủ yếu của các hệ quản trị CSDL. C.J. Date [3] đã định nghĩa tính độc lập dữ liệu là "*tính bất biến của các hệ ứng dụng đối với các thay đổi bên trong cấu trúc lưu trữ và chiến lược truy nhập CSDL*".

CHƯƠNG II

NHỮNG CÁCH TIẾP CẬN MỘT CSDL

Mô hình dữ liệu là sự trừu tượng hóa mô trường thực, nó là sự biểu diễn dữ liệu ở mức quan niệm. Mỗi loại mô hình dữ liệu đặc trưng cho một cách tiếp cận dữ liệu khác nhau của những nhà phân tích - thiết kế CSDL, mỗi loại đều có các ưu điểm và mặt hạn chế của nó nhưng vẫn có những mô hình dữ liệu nổi trội và được nhiều người quan tâm nghiên cứu. Cho đến nay đang tồn tại 5 loại mô hình dữ liệu, đó là: mô hình dữ liệu mạng, mô hình dữ liệu phân cấp, mô hình dữ liệu quan hệ, mô hình dữ liệu thực thể - kết hợp và mô hình dữ liệu hướng đối tượng. Chương này sẽ lần lượt giới thiệu các loại mô hình dữ liệu nêu trên.

2.1 Cách tiếp cận theo mô hình dữ liệu mạng

Mô hình dữ liệu mạng (*Network Data Model*) - còn được gọi tắt là mô hình mạng hoặc mô hình lưới (*Network Model*) là mô hình được biểu diễn bởi một đồ thị có hướng. Trong mô hình này người ta đưa vào các khái niệm: mẫu tin hay bản ghi (*Record*), loại mẫu tin (*Record Type*) và loại liên hệ (*Set Type*):

(a) *Loại mẫu tin (Record Type)* là mẫu đặc trưng cho 1 loại đối tượng riêng biệt. Chẳng hạn như trong việc quản lý nhân sự tại một đơn vị, đối tượng cần phản ánh của thế giới thực có thể là Phòng, Nhân viên, Công việc, lý lịch ... do đó có các loại mẫu tin đặc trưng cho từng đối tượng này. Trong đồ thị biểu diễn mô hình mạng mỗi loại mẫu tin được biểu diễn bởi một hình chữ nhật, một thể hiện (*Instance*) của một loại mẫu tin được gọi là bản ghi. Trong ví dụ trên loại mẫu tin Phòng có các mẫu tin là các phòng, ban trong đơn vị;

loại mẫu tin nhân viên có các mẫu tin là các nhân viên đang làm việc tại các phòng ban của cơ quan...

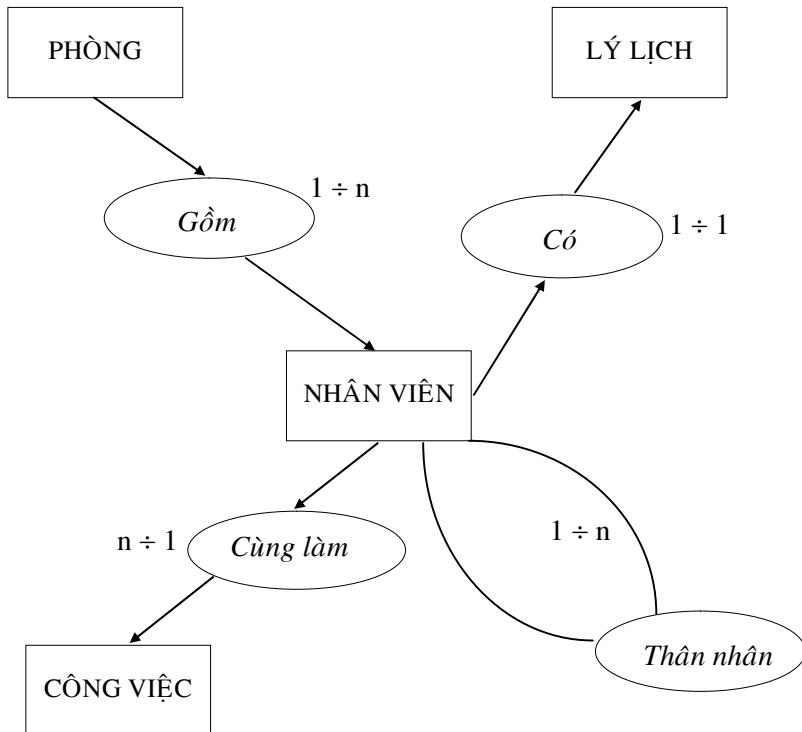
(b) *Loại liên hệ (Set Type)* là sự liên kết giữa một loại mẫu tin chủ với một loại mẫu tin thành viên. Trong đồ thị biểu diễn mô hình mạng mỗi loại liên hệ được biểu diễn bởi một hình bầu dục (*oval*) và sự liên kết giữa 2 loại mẫu tin được thể hiện bởi các cung có hướng (các mũi tên) đi từ loại mẫu tin chủ tới loại liên hệ và từ loại liên hệ tới loại mẫu tin thành viên.

Trong loại liên kết người ta còn chỉ ra số lượng các mẫu tin tham gia trong mối kết hợp. Có các loại liên hệ sau:

- $\div 1$ (*One-to-One*): Mỗi mẫu tin của loại mẫu tin chủ chỉ kết hợp với đúng 1 mẫu tin của loại mẫu tin thành viên. Ví dụ, mỗi nhân viên có duy nhất một lý lịch cá nhân.
- $\div n$ (*One-to-Many*): Mỗi mẫu tin của loại mẫu tin chủ chỉ kết hợp với 1 hay nhiều mẫu tin của loại mẫu tin thành viên. Ví dụ, mỗi phòng ban có từ 1 đến nhiều nhân viên. Mỗi 1 nhân viên chỉ thuộc một phòng ban nhất định.
- $n \div 1$ (*Many-to-One*): Nhiều mẫu tin của loại mẫu tin chủ chỉ kết hợp với đúng 1 mẫu tin của loại mẫu tin thành viên. Ví dụ, nhiều nhân viên cùng làm một công việc.
- **Đệ quy (*Recursive*)**: Một loại mẫu tin chủ cũng có thể đồng thời là loại mẫu tin thành viên với chính nó. Ta nói rằng loại liên hệ này là đệ quy.

Hình 2.1 biểu diễn một ví dụ về mô hình dữ liệu mạng đối với CSDL nhân sự của một đơn vị. Trong đồ thị này, chúng ta có 4 loại mẫu tin: phòng, nhân-viên, công-việc và lý-lịch; 4 loại liên hệ: phòng gồm 1 đến nhiều nhân-viên; nhân-viên có đúng 1 lý-lịch; nhiều nhân-viên cùng làm một công-việc; 1 nhân-viên có thể có 1 hay nhiều nhân-viên là thân nhân của mình.

Mô hình dữ liệu mạng tương đối đơn giản, dễ sử dụng nhưng nó không thích hợp trong việc biểu diễn các CSDL có quy mô lớn bởi trong một đồ thị có hướng khả năng diễn đạt ngữ nghĩa của dữ liệu, nhất là các dữ liệu và các mối liên hệ phức tạp của dữ liệu trong thực tế là rất hạn chế.



Hình 2.1 Mô hình dữ liệu mạng (Network Model)

2.2. Mô hình dữ liệu phân cấp.

Mô hình dữ liệu phân cấp (*Hierachical Data Model*) - được gọi tắt là mô hình phân cấp (*Hierachical Model*): Mô hình là một cây (*Tree*), trong đó mỗi nút của cây biểu diễn một thực thể, giữa nút con và nút cha được liên hệ với nhau theo một mối quan hệ xác định.

Mô hình dữ liệu phân cấp sử dụng các khái niệm sau:

(a) *Loại mẫu tin*: giống khái niệm mẫu tin trong mô hình dữ liệu mạng.

(b) *Loại mối liên hệ*: Kiểu liên hệ là phân cấp, theo cách:

- Mẫu tin thành viên chỉ đóng vai trò thành viên của một mối liên hệ duy nhất, tức là nó thuộc một chủ duy nhất. Như vậy, mối liên hệ từ mẫu tin chủ tới các mẫu tin thành viên là 1÷n, và từ mẫu tin (hay bản ghi - record) thành viên với mẫu tin chủ là 1÷1.
- Giữa 2 loại mẫu tin chỉ tồn tại 1 mối liên hệ duy nhất.

Ví dụ 2.2.1:

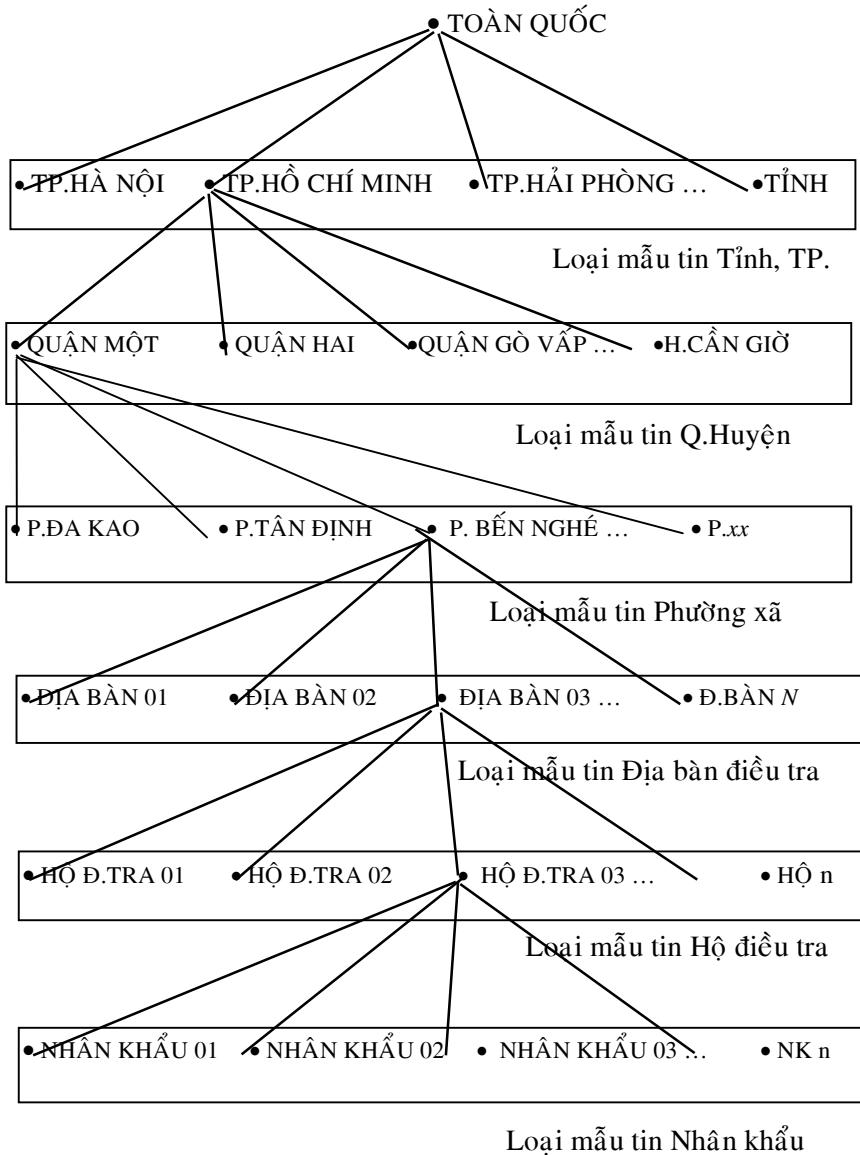
Trong cuộc Tổng điều tra số dân năm 1989, chương trình nhập phiếu điều tra được viết bằng ngôn ngữ CENS4 cho kết quả là các file dữ liệu nhập dạng văn bản được tổ chức như sau:

- Các dòng là các mẫu tin (hay bản ghi) có độ dài thay đổi.

- Có 6 loại mẫu tin:

- Mẫu tin đặc trưng cho tỉnh, thành phố gồm Mã số tỉnh thành, Tên tỉnh thành phố. '02' là Mã số Thành phố Hồ Chí Minh.
- Mẫu tin đặc trưng cho quận huyện gồm Mã số tỉnh thành+Mã số quận huyện, Tên quận huyện trong tỉnh thành phố đó. '0201' là Mã số quận Nhất của TP.Hồ Chí Minh.
- Mẫu tin đặc trưng cho phường xã gồm Mã số tỉnh thành+Mã số quận huyện+Mã số phường xã, Tên phường xã thuộc

quận huyện trong tỉnh thành phố đó. '020101' là Mã số
phường Bến Nghé, Quận Nhất, TP.Hồ Chí Minh.



Hình 2.2 Mô hình dữ liệu phân cấp (Hierachical Model)

- Mẫu tin đặc trưng cho địa bàn điều tra trong một phường xã. '02010101' là mã số địa bàn điều tra số 01 trong phường Bến Nghé.
- Mẫu tin đặc trưng cho hộ điều tra, gồm Mã số tỉnh + Mã số quận + Mã số phường + Mã số địa bàn + Số thứ tự hộ điều tra trong địa bàn, Tổng số nhân khẩu trong hộ, Trong đó: Nữ, Tổng số trẻ dưới 16 tuổi.
- Mẫu tin đặc trưng cho nhân khẩu của hộ, gồm các thông tin xác định hộ điều tra, Số thứ tự nhân khẩu trong hộ, Quan hệ với chủ hộ (1÷9), Giới tính (1,2,3), Tháng sinh, Năm sinh, Trình độ văn hóa, ...

Ở đây rõ ràng có một sự phân cấp trong file CSDL. Một tỉnh thành phố (thì) có nhiều quận huyện, một quận huyện chỉ thuộc một tỉnh thành duy nhất. Một quận huyện (thì) có nhiều phường xã và một phường xã chỉ thuộc một quận huyện duy nhất. Mỗi phường xã được chia thành nhiều địa bàn điều tra, mỗi địa bàn chỉ thuộc một phường xã duy nhất v.v...

Hình 2.2 mô tả cây phân cấp của mô hình dữ liệu đối với CSDL Tổng điều tra số dân Toàn quốc 0 giờ ngày 01 tháng 01 năm 1989.

2.3. Mô hình dữ liệu quan hệ.

Mô hình dữ liệu quan hệ (*Relational Data Model*) - còn được gọi tắt là mô hình quan hệ (*Relational Model*) do E.F.Codd [2] đề xuất năm 1970. Nền tảng cơ bản của nó là khái niệm lý thuyết tập hợp trên các quan hệ, tức là tập của các bộ giá trị (*Value Tuples*). Trong mô hình dữ liệu này những khái niệm sẽ được sử dụng bao

gồm thuộc tính (*Attribute*), quan hệ (*Relation*), lược đồ quan hệ (*Relation Schema*), bộ (*Tuple*), khóa (*Key*).

Mô hình dữ liệu quan hệ là mô hình được nghiên cứu nhiều nhất, và cho thấy rằng nó có cơ sở lý thuyết vững chắc nhất. Mô hình dữ liệu này cùng với mô hình dữ liệu thực thể kết hợp đang được sử dụng rộng rãi trong việc phân tích và thiết kế CSDL hiện nay. Chúng ta sẽ nghiên cứu chi tiết mô hình dữ liệu này ở các chương sau.

2.4. Mô hình dữ liệu thực thể - kết hợp.

Mô hình dữ liệu thực thể - kết hợp (*Entity - Relationship Model*) do P.P.Chen đề xuất vào năm 1976. Các khái niệm chủ yếu được sử dụng trong lý thuyết của mô hình này là:

Loại thực thể (*Entity Type*): Là một loại đối tượng cần quản lý trong CSDL, chẳng hạn, KHOA, LỚP-HỌC, MÔN-HỌC, GIÁNG-VIÊN, HỌC-VIÊN, tức là, cũng tương tự như khái niệm về loại mẫu tin trong mô hình mạng và mô hình phân cấp.

- Thực thể (*Entity*): Là một thể hiện hoặc một đối tượng của một loại thực thể. Khái niệm này tương tự như khái niệm mẫu tin trong mô hình dữ liệu mạng và mô hình dữ liệu phân cấp.

- Thuộc tính của loại thực thể (*Entity Attribute*): Là các đặc tính riêng biệt cơ bản của loại thực thể, tương tự khái niệm thuộc tính (*Attribute*) trong mô hình dữ liệu quan hệ sẽ trình bày trong Chương III. Ví dụ, loại thực thể KHOA có các thuộc tính Mã-Khoa, Tên-Khoa. Loại thực thể LỚP-HỌC có một số thuộc tính Mã-Lớp, Tên-Lớp, Niên-Khóa, Số-Học-Viên. Loại thực thể MÔN-HỌC có một số thuộc tính Mã-Môn, Tên-Môn, Số-Đv-Học-Trình. Loại thực thể HỌC-VIÊN có một số thuộc tính Mã-Học-Viên, Tên-Học-Viên,

Ngày-Sinh, Quê-Quán. Loại thực thể GIẢNG-VIÊN có một số thuộc tính Mã-Giảng-Viên, Tên-Giảng-Viên, Cấp-Học-Vị, Chuyên-Ngành. v.v...

- Khóa của loại thực thể (*Entity Key*): Đó là các thuộc tính nhận diện của loại thực thể. Căn cứ vào các giá trị của các thuộc tính nhận diện này người ta có thể xác định một thực thể duy nhất của một loại thực thể. Ví dụ, khóa của loại thực thể LỚP-HỌC có thể là Mã-Lớp; khóa của loại thực thể HỌC-VIÊN có thể là Mã-Học-Viên; khóa của loại thực thể MÔN-HỌC có thể là Mã-Môn-Học ... Khái niệm này cũng tương tự như khái niệm khóa (*Key*) trong mô hình dữ liệu quan hệ sẽ trình bày trong Chương III.

- Loại mối kết hợp (*Entity Relationship*): Tương tự như loại mối liên hệ trong mô hình dữ liệu mạng. Trong đồ thị biểu diễn của mô hình này người ta cũng sử dụng hình elip để thể hiện một mối kết hợp giữa các thực thể. Giữa 2 loại thực thể có thể tồn tại nhiều hơn một mối kết hợp.

- Số ngôi của mối kết hợp (*Relationship Degree*): Là tổng số loại thực thể tham gia vào mối kết hợp. Ví dụ, giữa loại thực thể SINH-VIÊN và KHOA tồn tại mối kết hợp "trực thuộc" - đó là mối kết hợp 2 ngôi. KẾT-QUẢ-THI (hoặc KIỂM-TRA) của sinh viên là mối kết hợp giữa 3 thực thể SINH-VIÊN, MÔN-THI và ĐIỂM-THI - đó là mối kết hợp 3 ngôi.

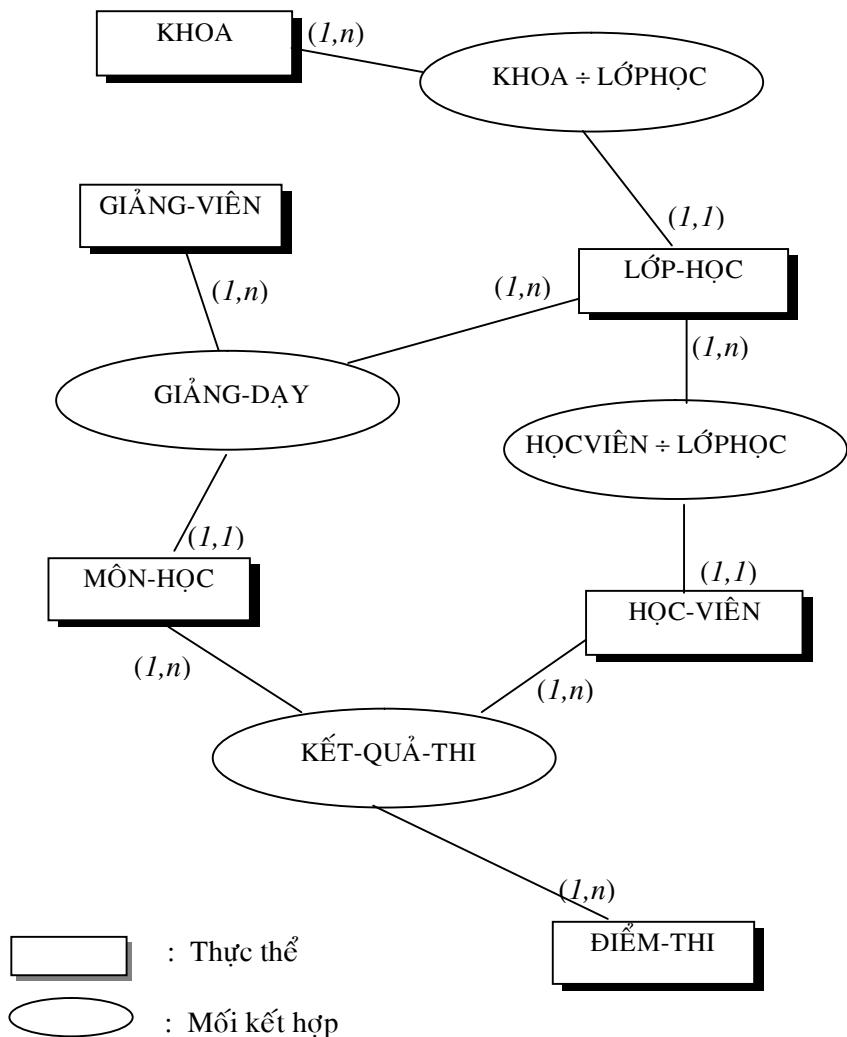
- Thuộc tính của mối kết hợp (*Relationship Attribute*): Mối kết hợp có thể có các thuộc tính của riêng nó. Thông thường mối kết hợp có các thuộc tính là khóa của các loại thực thể tham gia vào mối kết hợp, ngoài ra còn có thêm những thuộc tính bổ sung khác. Ví dụ, trong mối kết hợp 3 ngôi kể trên, thuộc tính của mối kết hợp này có thể bao gồm Mã-Học-Viên, Mã-Môn-Học, Điểm-Thi; và có thể có

thêm các thuộc tính bổ sung khác như Lần-Thi-Thứ, Ngày-Thi, Ghi-Chú v.v...

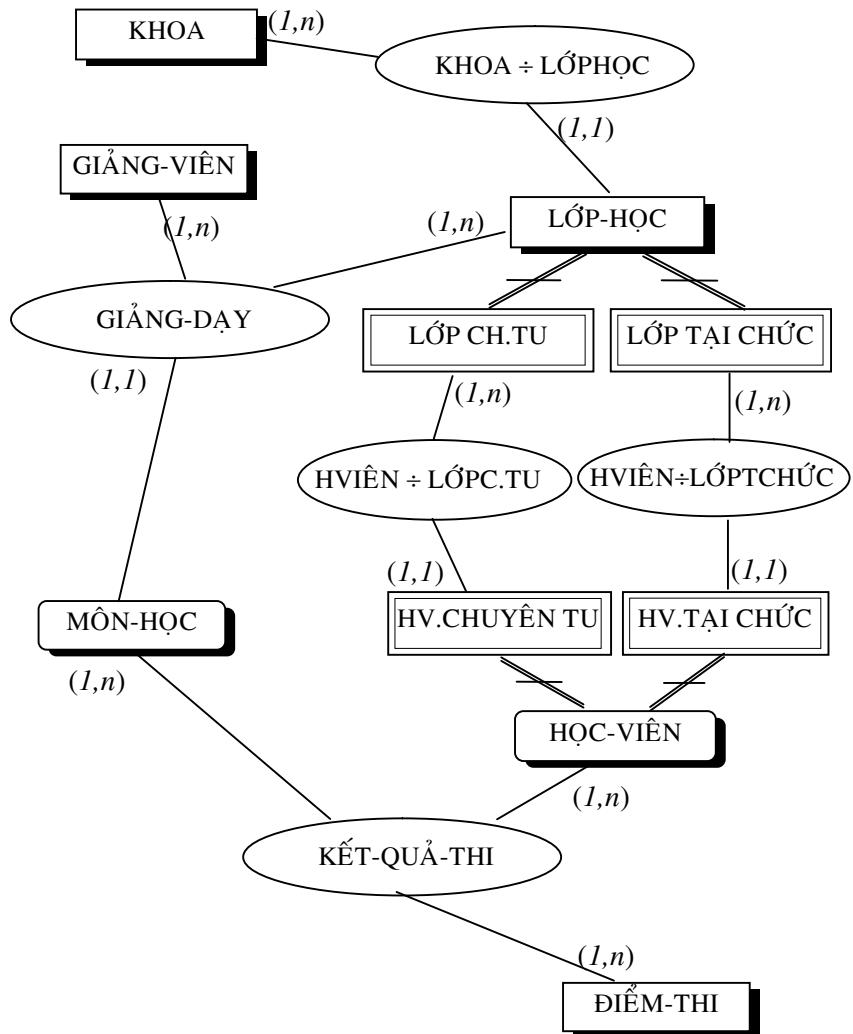
Bản số của mỗi nhánh của mối kết hợp (*Relationship Cardinal*): Mỗi nhánh (hay mỗi chiều) của mối kết hợp là mối kết hợp nối một loại thực thể với một mối kết hợp. Trong nhánh này cần xác định số lượng tối thiểu và số lượng tối đa các thực thể của nhánh đó sẽ tham gia vào một thực thể của mối kết hợp. Hai đại lượng này - đặt trong cặp dấu ngoặc tròn - được gọi là bản số của mối kết hợp. Ví dụ, trong mối kết hợp 3 ngôi nhà trên, tại nhánh nối loại thực thể HỌC-VIÊN với mối kết hợp KẾT-QUẢ-THI là (1,n), bởi vì sẽ có ít nhất một học viên tham gia kỳ thi và nhiều nhất là tất cả số học viên học môn đó cùng dự thi.

Vào khoảng năm 1980, mô hình dữ liệu *thực thể-kết hợp* đã được mở rộng thêm một số khái niệm mới như "loại thực thể chuyên biệt hóa" (*Specialized Entity*) và "mối kết hợp đệ quy" (*Recursive Relationship*). Mô hình này cùng với mô hình dữ liệu quan hệ và mô hình hướng đối tượng được sử dụng khá phổ biến trong việc thiết kế các CSDL hiện nay, bởi sự kết hợp này làm cho mô hình dữ liệu thể hiện được nhiều nghĩa của những loại dữ liệu trong CSDL hơn.

Hình 2.4.1 trình bày một mô hình *thực thể - kết hợp* cho CSDL quản lý học viên gồm các loại thực thể KHOA, LỚP-HỌC, MÔN-HỌC, GIÁNG-VIÊN, HỌC-VIÊN. Mỗi kết hợp HỌC-VIÊN-LỚP-HỌC giữa 2 loại thực thể HỌC-VIÊN và LỚP-HỌC. Tại nhánh HỌC-VIÊN bản số của nó là (1,1), điều này nhà phân tích và thiết kế đã khẳng định tình trạng thực tế là một học viên phải theo học (hay có tên trong danh sách) ít nhất là một lớp và cũng chỉ thuộc tối đa một lớp. Bản số của nhánh LỚP-HỌC là (1,n) nghĩa là một lớp (nếu đã có tên trong danh sách lớp học) thì có ít nhất 1 học viên và tối đa có thể là nhiều (n) học viên theo học.



Hình 2.4.1 Mô hình dữ liệu thực thể - kết hợp của CSDL quản lý học viên.



Hình 2.4.2. Mô hình thực thể kết hợp với các thực thể chuyên biệt hóa.

Nếu yêu cầu quản lý đòi hỏi phải phân tích rạch ròi thành 2 loại lớp học chính quy và tại chức, và các lớp học tại chức có thể sẽ không phải học một số môn học hoặc số đơn vị học trình ít hơn so với các lớp chính quy, khi đó mô hình dữ liệu *thực thể - kết hợp* được thể hiện như trong sơ đồ 2.4.2. Thực thể chuyên biệt hóa được thể hiện trong mô hình bằng hình chữ nhật có khung (*Frame*) đôi; một đường kết hợp bằng nét đôi có dấu gạch ngang thể hiện mối quan hệ là chuyên biệt hoá thành các loại thực thể con chi tiết hơn. Các tên trong các hình hộp chữ nhật thể hiện các thuộc tính của loại thực thể đó.

2.5. Mô hình dữ liệu hướng đối tượng.

Mô hình dữ liệu hướng đối tượng (*Object Oriented Data Model*) ra đời từ cuối những năm 80 và đầu những năm 90. Đây là loại mô hình tiên tiến nhất hiện nay dựa trên cách tiếp cận hướng đối tượng đã quen thuộc trong các phương pháp lập trình hướng đối tượng, nó sử dụng các khái niệm như lớp (*class*), sự kế thừa (*inheritance*), kế thừa bội (tức là kế thừa từ nhiều lớp cơ sở *multi-inheritance*). Đặc trưng cơ bản của cách tiếp cận này là tính đóng gói (*encapsulation*), tính đa hình (*polymorphism*) và tính tái sử dụng (*Reusability*).

Lớp là một kiểu dữ liệu có cấu trúc bao gồm các thành phần dữ liệu và các phương thức xử lý thao tác trên cấu trúc dữ liệu đó. Nó là một kiểu (hay cấu trúc) dữ liệu được trừu tượng hóa, bởi vì các tác động (còn gọi là các phương thức - *method*) là để phục vụ hoặc thao tác trên kiểu dữ liệu này. Dữ liệu và phương thức hòa quyện vào nhau thành một thể thống nhất: dữ liệu cần có những cách thức xử lý thỏa đáng, và phương thức xử lý được đưa vào trong kiểu dữ liệu đó là để phục vụ cho các đối tượng có cấu trúc như thế. Người ta gọi sự thống nhất đó là sự đóng gói.

Ví dụ, trong việc định nghĩa phép cộng (+) hai số phức c1 và c2 để cho một số phức kết quả là:

COMPLEX (c1.Real+c2.Real, c1.Image+c2.Image)

để người ta có thể sử dụng phép cộng (+) một cách tự nhiên như việc gán kết quả đó cho biến phức $c = c1 + c2$, hoàn toàn tự nhiên và trong sáng hơn rất nhiều so với việc phải viết một thủ tục Add_COMPLEX (COMPLEX &c1, COMPLEX &c2, COMPLEX &c) để cộng hai số phức và kết quả được gán vào tham đối thứ 3 của hàm thông qua tham chiếu địa chỉ của biến c theo cách lập trình hướng thủ tục trước đó. Và, hiển nhiên rằng, cách thức cộng như trên là chỉ được áp dụng cho các đối tượng số phức.

Tương tự, cũng có thể định nghĩa phép toán cộng (+) trong lớp ma trận (MATRIX) để cộng hai ma trận có cùng bậc. Khi đó giả sử A và B là hai ma trận $m * n$; Phép gán tổng hai ma trận A và B cho biến ma trận C có thể được viết:

$C = A + B$

rõ ràng là tự nhiên và dễ hiểu hơn đối với các ứng dụng lớp ma trận so với việc gọi thủ tục Add_MATRIX (MATRIX &A, MATRIX &B, MATRIX &C, int m, int n) ...

Phương pháp tiếp cận hướng đối tượng trong mô hình dữ liệu mặc dù còn mới mẻ nhưng hiện nay đang được nhiều người quan tâm nghiên cứu phát triển và áp dụng. Các hệ quản trị CSDL hướng đối tượng hiện nay vẫn chưa nhiều, một số còn chưa thuần nhất (nghĩa là việc lập trình là hướng đối tượng nhưng CSDL vẫn chủ yếu dựa trên mô hình quan hệ).

Tóm tắt, trong chương này đã trình bày một số vấn đề cơ bản:

- ❖ Sơ đồ tổng quát của một hệ quản trị CSDL.
- ❖ Tính độc lập giữa dữ liệu và chương trình. Tổ chức lưu trữ dữ liệu trên các thiết bị nhớ thứ cấp có thể bị thay đổi bằng cách thay đổi nơi lưu trữ dữ liệu hay thêm vào một số chỉ mục mới để cải thiện hiệu quả khai thác CSDL mà không làm thay đổi lược đồ và các lược đồ dữ liệu con cũng như hệ thống chương trình ứng dụng CSDL. Đó chính là tính độc lập vật lý (*Physical Independence*) giữa dữ liệu với chương trình. Việc thay đổi lược đồ khái niệm bằng cách thêm hay bớt đi một số field của một số bảng trong CSDL mà không làm ảnh hưởng tới các lược đồ con, do đó không cần phải viết lại các chương trình ứng dụng. Đó là tính độc lập của dữ liệu đối với chương trình ở mức lôgic (*Logical Independence*).
- ❖ Và một số cách tiếp cận CSDL:
 1. *Mô hình dữ liệu mạng*. Mô hình được biểu diễn bởi một đồ thị có hướng mà mỗi đỉnh được minh họa bằng một hình chữ nhật, thể hiện một loại mẫu tin, và mỗi cung có hướng thể hiện một loại liên hệ.
 2. *Mô hình dữ liệu phân cấp*. Mô hình dữ liệu này được biểu diễn bởi một cây (*Tree*), mà mỗi nút của cây thể hiện một loại mẫu tin. Giữa 2 loại mẫu tin chỉ tồn tại một mối liên hệ duy nhất. Quan hệ giữa nút cấp trên và các nút con cấp dưới là $1 \div n$ và $1 \div 1$ theo chiều ngược lại.
 3. *Mô hình dữ liệu quan hệ*. Các khái niệm được sử dụng là thuộc tính (*Attribute*), quan hệ (*Relation*), lược đồ quan hệ (*Relation Schema*), bộ giá trị (*Value Tuple*), khóa (*Key*). Tuy nhiên, những khái

niệm này cùng với một số khái niệm chi tiết khác sẽ được trình bày trong các chương còn lại - đó cũng là nội dung chính của tài liệu này.

4. Mô hình dữ liệu thực thể - kết hợp. Mục này đã đề cập tới khá nhiều khái niệm: Loại thực thể (*Entity type*), Thực thể (*Entity*), Thuộc tính của loại thực thể (*Entity Attributes*); Khóa của thực thể (*Entity Key*); Loại mối kết hợp (*Relationship*); Số ngôi của mối kết hợp (*Relationship Degree*); Thuộc tính của mối kết hợp (*Relationship Attributes*); Bản số (*Cardinal*) của mỗi nhánh của mối kết hợp; Thực thể chuyên biệt hóa (*Specialized Entity*); Mỗi kết hợp đệ quy (*Recursive Relationship*) và Đồ thị biểu diễn CSDL theo cách tiếp cận này.

5. Mô hình dữ liệu hướng đối tượng. Nền tảng là các lớp (*class*) đó là một cấu trúc dữ liệu được đóng gói cùng các thao tác tác động trên các đối tượng của lớp. Do tính kế thừa (*Inheritance*) mà có thể xây dựng được các lớp mới dẫn xuất từ các lớp cơ sở đã có. Đến lượt nó lại trở thành cơ sở để xây dựng các lớp mới khác (Cứ như thế, các lớp sẽ “bung nổ” mãi mãi, ngày một lớn hơn với nhiều chức năng (hoặc công cụ) hoàn thiện hơn.

BÀI TẬP THỰC HÀNH.

Bài tập 1: Dựa vào những khái niệm đã học, Anh/Chị hãy biểu diễn CSDL có các loại mẫu tin Phòng, Nhân viên, Công việc, lý lịch đã trình bày trong mô hình mạng theo cách tiếp cận phân cấp.

- Loại liên hệ là phân cấp:
- Phòng có nhiều nhân-viên; mỗi nhân-viên chỉ thuộc 1 phòng duy nhất.
- Công-việc có nhiều nhân-viên cùng làm; mỗi nhân-viên chỉ làm một công-việc duy nhất.
- Mỗi nhân-viên có một lý-lịch; mỗi lý-lịch chỉ thuộc 1 nhân-viên duy nhất.

Bài tập 2: Dựa vào những khái niệm đã học, Anh/Chị hãy biểu diễn CSDL về Tổng điều tra số dân toàn quốc 0 giờ ngày 01 tháng 01 năm 1989 có các loại mẫu tin tinh-thành-phố, Quận-huyện, Phường-xã, Địa-bàn, hộ-điều tra và nhân-khẩu đã trình bày trong mô hình phân cấp theo cách tiếp cận mạng.

- Loại liên hệ phân mạng là loại "*thuộc về*"
- Nhân khẩu (*thì*) thuộc một hộ-điều tra.
- Hộ-điều tra (*thì*) thuộc một địa-bàn.
- Địa-bàn điều tra (*thì*) thuộc một phường-xã.
- Phường-xã thuộc một quận-huyện
- Quận-huyện thuộc một tỉnh/thành phố.

Bài tập 3: Hệ thống thông tin quản lý lưu trữ các văn bản pháp quy tại một cơ quan quản lý Nhà nước có CSDL được phân tích và thiết kế theo cách tiếp cận "*Thực thể-kết hợp*" gồm các loại thực thể và các mối kết hợp sau:

1. CÔNG-VĂN-ĐẾN (thuộc tính bao gồm: ngày phát hành; Số và ký hiệu công văn; trích yếu nội dung; ngày nhận; số trang văn bản; ghi chú).
2. CÔNG-VĂN-ĐI (thuộc tính bao gồm: ngày phát hành; Số và ký hiệu công văn; trích yếu nội dung; người ký; số trang văn bản; Số tờ trình ký; Ngày trình ký; ghi chú).
3. CÔNG-VĂN-ĐẾN và CÔNG-VĂN-ĐI đều là CÔNG-VĂN, là hai loại thực thể chuyên biệt hóa của loại thực thể CÔNG-VĂN !
4. DANH-SÁCH-CHUYÊN-VIÊN (thuộc tính bao gồm: mã chuyên viên; Tên chuyên viên; Phòng ban đang công tác; Ghi chú bổ sung, nếu có)

CHƯƠNG III

MÔ HÌNH DỮ LIỆU QUAN HỆ CỦA E.F.CODD

3.1. Các khái niệm cơ bản.

Phần mở đầu của Chương III sẽ trình bày kỹ hơn về các khái niệm đã được nhắc tới trong Chương II về cách tiếp cận mô hình dữ liệu kiểu quan hệ và sẽ coi đó như là những cơ sở nền tảng để tiếp tục nghiên cứu các phần tiếp theo.

3.1.1. Thuộc tính (Attribute):

Thuộc tính là một tính chất riêng biệt của một đối tượng cần được lưu trữ trong CSDL để phục vụ cho việc khai thác dữ liệu về đối tượng.

Ví dụ 3.1.1:

- ❖ Đối tượng KHOA (tương ứng với loại thực thể KHOA trong mô hình thực thể kết hợp) có các thuộc tính Mã-khoa, Tên-khoa.
- ❖ Loại thực thể LỚP-HỌC có một số thuộc tính Mã-lớp, Tên-lớp, Niêm-khoa, Số-học-viên.
- ❖ Loại thực thể MÔN-HỌC có một số thuộc tính Mã-môn, Tên-môn, Số-Đv-Học-Trình.
- ❖ Loại thực thể HỌC-VIÊN có một số thuộc tính Mã-khoa, Mã-học-viên, Tên-học-viên, Ngày-sinh, Quê-quán.
- ❖ Loại thực thể GIẢNG-VIÊN có một số thuộc tính Mã-giảng-viên, Tên-giảng-viên, Cấp-học-vị, Chuyên-ngành. v.v...

Các thuộc tính được đặc trưng bởi một tên gọi, kiểu giá trị và miền giá trị của chúng. Trong giáo trình này, để trình bày một cách tổng quát và nếu không cần lưu ý đến ngữ nghĩa thì tên của các thuộc tính thường được ký hiệu bằng các chữ cái in hoa đầu tiên trong bảng chữ cái la tinh: A, B, C, D, ... Những chữ cái X, Y, W, Z, ... dùng thay cho một nhóm (hay tập hợp) gồm nhiều thuộc tính. Đôi khi còn dùng các ký hiệu chữ cái với các chỉ số A1, A2,, An để chỉ các thuộc tính trong trường hợp tổng quát hay muốn đề cập đến số ngôi (hay số lượng các thuộc tính) của một quan hệ.

Trong các ứng dụng thực tế, người phân tích - thiết kế thường đặt tên thuộc tính một cách gợi nhớ; nhưng để làm rõ hơn ý nghĩa của những tên gọi, người ta có thể đặt tên khá dài cho các thuộc tính với các chữ in hoa đầu từ hoặc viết cách nhau bởi dấu gạch chân (*Underscore*: _).

Trong các ví dụ của tài liệu này, các tên thuộc tính được viết bằng tiếng Việt gồm nhiều từ Việt nối với nhau bởi dấu trurus (-) có chữ cái đầu tiên được viết in hoa nhằm mục đích chuyển tải cả ngữ nghĩa của tên thuộc tính. Điều này không có gì sai, bởi vì hiện nay có một số hệ quản trị CSDL cho phép làm như vậy (*MicroSoft Access, SQL-Server* cho phép đặt tên dài tới 255 ký tự và có thể có chứa các khoảng trắng, các ký tự tiếng Việt có dấu và các ký tự đặc biệt khác). Những tên thuộc tính hoặc tên quan hệ như vậy, khi sử dụng trong Micro Soft Access hoặc SQL-Server phải viết chúng trong cặp dấu ngoặc vuông ([]), khi sử dụng trong ORACLE phải viết trong cặp dấu nháy kép (" – Quotes). Trong tài liệu này chúng ta sử dụng ký pháp của SQL-Server.

Trong cài đặt cụ thể với một hệ quản trị CSDL cần lưu ý đến khía cạnh đặt tên cho các bảng cũng như tên của thuộc tính. Trong hầu hết các ngôn ngữ lập trình nói chung và một số ngôn ngữ quản trị CSDL nói riêng, tên đối tượng (tên biến, tên quan hệ hay tên thuộc

tính v.v...) đều chỉ được phép viết bằng các chữ cái la tinh, chữ số và/hoặc dấu gạch chân (*underscore* '_'), bắt đầu bằng chữ cái hoặc dấu gạch chân, với độ dài tên theo quy định. Theo lý thuyết, người ta vẫn khuyên rằng không nên đặt tên thuộc tính quá dài (bởi vì nó làm cho việc viết các câu lệnh truy vấn trở nên vất vả hơn) và cũng không nên đặt tên thuộc tính quá ngắn (vì nó không cho thấy ngữ nghĩa của thuộc tính của quan hệ), đặc biệt là không đặt trùng tên hai thuộc tính mang ngữ nghĩa khác nhau thuộc hai đối tượng khác nhau. Chẳng hạn, nếu có hai đối tượng HỌC-VIÊN và GIẢNG-VIÊN đều có thuộc tính TÊN thì nên đặt tên thuộc tính rõ ràng là Tên_hoc-vien của loại đối tượng HỌC-VIÊN và Tên-giáo-viên cho đối tượng GIẢNG-VIÊN, bởi vì 2 thuộc tính TÊN đó mang ngữ nghĩa khác nhau trong 2 quan hệ khác nhau.

Mỗi thuộc tính đều phải thuộc một kiểu dữ liệu (*Data Type*) nhất định. Kiểu dữ liệu có thể là vô hướng (đó là các kiểu dữ liệu cơ bản như chuỗi - *String* hoặc *Text* hoặc *Character*, số - *Number*, Luận lý - *Logical*, ...) hoặc các kiểu dữ liệu có cấu trúc được định nghĩa dựa trên các kiểu dữ liệu đã có sẵn. Một số kiểu dữ liệu vô hướng sau đây thường được sử dụng trong các hệ quản trị CSDL :

- *Text* (hoặc *Character*, *String*, hoặc *Char*) – kiểu văn bản.
- *Number* (hoặc *Numeric*, hoặc *float*) – kiểu số
- *Logical* (hoặc *Boolean*) – kiểu luận lý
- *Date/Time* – kiểu thời gian : ngày tháng năm + giờ phút
- *Memo* (hoặc *VarChar*) – kiểu văn bản có độ dài thay đổi.

Mỗi hệ quản trị CSDL có thể gọi tên các kiểu dữ liệu nói trên bằng các tên gọi khác nhau, ngoài ra còn bổ sung thêm một số kiểu dữ liệu riêng của mình. Chẳng hạn, MicroSoft Access có kiểu dữ liệu OLE để chứa các đối tượng nhúng như hình ảnh, âm thanh, audio, video ... ORACLE có kiểu dữ liệu LONG cho phép chứa dữ liệu có kích thước lớn tới 2 tỷ bytes.

Mỗi thuộc tính có thể chỉ chọn lấy những giá trị trong một tập hợp con của kiểu dữ liệu. Tập hợp các giá trị mà một thuộc tính A có thể nhận được gọi là miền giá trị (*domain*) của thuộc tính A và được ký hiệu là MGT(A) hoặc *Dom*(A).

Ví dụ 3.1.2:

Học viên đang theo học tại trường ĐH.KHTN thì tuổi của họ nhiều nhất là 60 và tuổi ít nhất là 18, vừa mới tốt nghiệp PTTH. Mặc dù nói rằng Năm-sinh của học viên là một số nguyên, nhưng không phải số nguyên nào cũng có thể được chọn để gán vào thuộc tính Năm-sinh. Giá trị năm sinh của học viên chỉ cần lưu 2 chữ số sau của năm sinh tức là chỉ cần một byte để ghi nhận những năm sinh của họ trong thế kỷ 20: từ năm 40 đến năm 82. Với miền giá trị chỉ chứa từ 40 đến 82, như vậy chỉ cần dùng 1 byte để lưu là đủ.

Nếu kiểu dữ liệu của thuộc tính A là có cấu trúc thì miền giá trị của A là tích Đê-các (hoặc tập con của tích Đê-các - *Cartesian*) của các miền giá trị thành phần.

Ví dụ 3.1.3:

Ta có kiểu dữ liệu ngày tháng năm theo dương lịch với mô tả bằng ngôn ngữ Pascal như sau:

```
Type DATE = Record
  Day      : 1 .. 31;
  Month    : 1 .. 12;
  Year     : 0 .. 2500;
End;
```

Tích Đê-các của 3 miền giá trị của các thành phần là [1 .. 31] x [1 .. 12] x [0 .. 2500]. Nếu thuộc tính A thuộc kiểu DATE thì MGT(A) $\subset [1..31] \times [1..12] \times [0..2500]$, bởi vì $\{30,02,1999\}$ không phải là một ngày tháng năm hợp lệ nên tổ hợp đó không thuộc MGT(A). Ngày

nay, hầu hết các ngôn ngữ quản trị dữ liệu đều có định nghĩa kiểu này như một kiểu cơ bản. Tổ hợp 3 giá trị thành phần luôn luôn được kiểm tra tính đúng đắn trước khi được coi là một giá trị kiểu ngày tháng. Hai phép toán số học có thể tác động trên kiểu DATE là phép cộng (+) hoặc trừ (-) một DATE với một số nguyên để cho kết quả là một giá trị kiểu DATE; hiệu 2 giá trị kiểu DATE là số ngày trôi qua giữa 2 ngày tháng năm đó.

Trong nhiều hệ quản trị CSDL, người ta thường đưa thêm vào miền giá trị của các thuộc tính một giá trị đặc biệt gọi là giá trị rỗng (NULL). Tùy theo ngữ cảnh mà giá trị này có thể đặc trưng cho một giá trị không thể xác định được hoặc một giá trị chưa được xác định ở vào thời điểm nhập tin nhưng có thể được xác định vào một thời điểm khác.

Nếu thuộc tính có kiểu dữ liệu là vô hướng thì nó được gọi là thuộc tính đơn hoặc thuộc tính nguyên tố; nếu thuộc tính có kiểu dữ liệu có cấu trúc thì ta nói rằng nó là thuộc tính kép (hay không phải là nguyên tố).

3.1.2. Quan hệ (Relation):

Một quan hệ R có n ngôi được định nghĩa trên tập các thuộc tính $U = \{A_1, A_2, \dots, A_n\}$ (thứ tự của các thuộc tính là không quan trọng) và kèm theo nó là một tân từ, tức là một quy tắc để xác định mỗi quan hệ giữa các thuộc tính A_i và được ký hiệu là $R(A_1, A_2, \dots, A_n)$. Tập thuộc tính của quan hệ R đôi khi còn được ký hiệu là R^+ .

Với A_i là một thuộc tính có miền giá trị là $MGT(A_i)$, như vậy $R(A_1, A_2, \dots, A_n)$ là tập con của tích Đè-các: $MGT(A_1) \times MGT(A_2) \times \dots \times MGT(A_n)$.

Quan hệ còn được gọi bằng thuật ngữ khác là bảng (*Table*).

Ví dụ 3.1.4:

KHOA (Mã-khoa, Tên-khoa), là một quan hệ 2 ngôi.
Tân từ: "Mỗi khoa có một tên gọi và một mã số duy nhất để phân biệt với tất cả các khoa khác của trường".

Ví dụ 3.1.5:

LỚP-HỌC (Mã-lớp, Tên-lớp, Niên-khoa, Số-học-viên, Mã-khoa) là quan hệ 5 ngôi với tân từ: "Mỗi lớp học trong trường có một mã số quy ước duy nhất để phân biệt với tất cả các lớp học khác trong trường; có một tên gọi của lớp học, một số lượng học viên theo học và thuộc một khoa của trường".

Ví dụ 3.1.6:

MÔN-HỌC (Mã-môn, Tên-môn, Số-đv-học-trình) là quan hệ 3 ngôi.
Tân từ: "Mỗi môn học có một tên gọi cụ thể, được học trong một số đơn vị học trình nhất định và ứng với môn học là một mã số duy nhất để phân biệt với mọi môn học khác".

Ví dụ 3.1.7:

HỌC-VIÊN (Mã-học-viên, Tên-học-viên, Ngày-sinh, Quê-quán, Mã-lớp) là quan hệ 5 ngôi.
Tân từ: "Mỗi học viên có một họ và tên, ngày sinh, quê quán, ... và được cấp một mã số duy nhất để phân biệt với mọi học viên khác trong trường; học viên được ghi danh vào một lớp học duy nhất trong trường".

Ví dụ 3.1.8:

GIẢNG-VIÊN (Mã-giảng-viên, Tên-giảng-viên, Cấp-học-vị, Chuyên-ngành).
Đây là quan hệ 4 ngôi.
Tân từ: "Mọi giảng viên đều có họ tên, cấp học vị thuộc một chuyên ngành nhất định và được gán cho một mã số duy nhất, gọi là Mã giảng viên, để phân biệt với mọi giảng viên khác trong trường".

3.1.3. Bộ giá trị (Tuple):

Một bộ giá trị là các thông tin của một đối tượng thuộc quan hệ. Bộ giá trị cũng thường được gọi là mẫu tin hay bản ghi (*record*) hoặc dòng của bảng (*Row*). Về mặt hình thức, một bộ q là một vectơ gồm n thành phần thuộc tập hợp con của tích Đê-các miền giá trị của các thuộc tính và thỏa mãn tân từ đã cho của quan hệ:
 $q = (a_1, a_2, \dots, a_n) \in MGT(A_1) \times MGT(A_2) \times \dots \times MGT(A_n)$.

Ví dụ 3.1.9:

Đây là 4 bộ giá trị dựa trên các thuộc tính của quan hệ HỌC-VIÊN:

q1 = (SV001,	Nguyễn Văn Nam,	27/03/1970,	Cần Thơ,	QTKD1)
q2 = (SV005,	Vũ Thị Tuyết Mai,	26/02/1968,	Đồng Nai,	KTKC1)
q3 = (SV014,	Hồng Đăng,	30/04/1975,	Đồng Nai,	CNTK3)
q4 = (SV015,	Lê Hoài Nhớ,	23/03/1965,	Long An,	CNTK4)

Để lấy thành phần A_i (tức là giá trị thuộc tính A_i) của bộ giá trị q, ta viết $q.A_i$. Phép trích rút này được gọi là phép chiếu một bộ lên thuộc tính A_i . $Q1.[Tên-học-viên] = 'Nguyễn Văn Nam'$.

☞ Lưu ý: Trong MicroSoft Access, hằng văn bản được viết trong cặp dấu nháy kép (""), còn trong SQL-Server và trong ORACLE, hằng văn bản được viết trong cặp dấu nháy đơn (' ').

3.1.4. Lược đồ quan hệ (Relation schema):

Lược đồ quan hệ là sự trừu tượng hóa của quan hệ, một sự trừu tượng hóa ở mức độ cấu trúc của một bảng hai chiều. Khi nói tới lược đồ quan hệ tức là đề cập tới cấu trúc tổng quát của một quan hệ; khi đề cập tới quan hệ thì điều đó được hiểu rằng đó là một bảng có cấu trúc cụ thể hoặc một định nghĩa cụ thể trên một lược đồ quan hệ với các bộ giá trị của nó.

Lược đồ cơ sở dữ liệu \mathcal{Q} là tập hợp các lược đồ quan hệ con $\{R_i\}$.

3.1.5. Thể hiện của quan hệ (Occurrence of a Relation)

Thể hiện (hoặc còn gọi là *tình trạng*) của quan hệ R, ký hiệu bởi T_R , là tập hợp các bộ giá trị của quan hệ R vào một thời điểm. Tại những thời điểm khác nhau thì quan hệ sẽ có những thể hiện khác nhau. Thể hiện (hay tình trạng) của các lược đồ quan hệ con T_{R_i} gọi là tình trạng của lược đồ cơ sở dữ liệu \mathcal{Q} .

Ví dụ 3.1.10:

Các thể hiện của quan hệ LỐP-HỌC và MÔN-HỌC:

- Quan hệ LỐP-HỌC:

Mã-lớp	Tên-lớp	Niên-khoa	Số-Hviên	Mã-khoa
QTKD1	Quản trị kinh doanh QT01	96-99	145	QTKD
KTCK1	Tài chánh - Kế toán KT4	96-99	230	TCKT
KTCK2	Tài chánh - Kế toán KT5	97-2000	120	TCKT
CNTK3	Cử nhân Cao Đẳng Tin học K3	98-2000	172	CNTT
CNTK4	Cử nhân Cao Đẳng Tin học K4	99-2001	241	CNTT

- Quan hệ MÔN-HỌC:

Mã-môn	Tên-môn	Số-đv-học-trình
TCKT	Tài chính - kế toán	4
KTCT	Kinh tế chính trị	4
MSACB	Hệ quản trị CSDL (Cơ bản)	5
TOANC	Toán Cơ sở	4
LTCBC	Lập trình căn bản C	5
CSDL1	Nhập môn Cơ sở dữ liệu	5
LTHDT	Lập trình hướng đối tượng	5

3.1.6. Khóa - Siêu khóa - Khóa chỉ định - Khóa chính - Khóa ngoại

Có nhiều cách khác nhau để định nghĩa khóa:

Định nghĩa 3.1.6.1

Khóa (*Key*) của lược đồ quan hệ R định nghĩa trên tập các thuộc tính $U = \{A_1, A_2, \dots, A_n\}$ là một tập con $K \subseteq U$ thỏa mãn các tính chất sau: với mọi bộ giá trị q_1, q_2 của R đều tồn tại một thuộc tính $A \in K$ sao cho $q_1.A \neq q_2.A$. Điều này có nghĩa là không tồn tại hai bộ nào có giá trị bằng nhau trên mọi thuộc tính của K. Mở rộng phép chiếu của bộ lên tập thuộc tính K ta có thể viết $q_1.K \neq q_2.K$. Như vậy mỗi giá trị của khóa K phải là xác định duy nhất trên quan hệ R.

Theo định nghĩa trên, nếu $K' \subseteq K \subseteq U$ là khóa của lược đồ quan hệ R thì K cũng là khóa của R, bởi vì $q_1.K' \neq q_2.K'$ thì cũng có $q_1.K \neq q_2.K$. Như vậy trong lược đồ quan hệ có thể có rất nhiều khóa. Việc xác định tất cả các khóa của một lược đồ quan hệ là rất khó khăn.

Định nghĩa này là chưa chặt chẽ. Chúng ta có thể định nghĩa khóa tốt hơn và một cách hình thức như sau:

Định nghĩa 3.1.6.2

Quan hệ R định nghĩa trên tập các thuộc tính $U = \{A_1, A_2, \dots, A_n\}$ $K \subseteq U$ là khóa của quan hệ R nếu thỏa 2 điều kiện sau đây:

- (i) K xác định được giá trị của A_j với mọi $j = 1, 2, \dots, n$
- (ii) Không tồn tại $K' \subseteq K$ mà K' có thể xác định được giá trị của A_j với mọi $j = 1, 2, \dots, n$

Nghĩa là K là tập con nhỏ nhất mà giá trị của nó có thể xác định duy nhất một bộ giá trị của quan hệ. Khóa của quan hệ theo định nghĩa 3.1.6.2 được gọi là khóa chỉ định (*Candidate*) và là khóa nội của quan hệ. Trong các phần tiếp theo, nếu không có chú thích gì thêm, thì các khóa chỉ định đều được gọi chung là các khóa.

K là siêu khóa của quan hệ R nếu $K' \subseteq K$ là một khóa của quan hệ. Một lược đồ quan hệ Q của quan hệ R luôn luôn có ít nhất một siêu khóa và có thể có nhiều siêu khóa.

Ví dụ 3.1.11:

Quan hệ LỚP-HỌC (Mã-lớp, Tên-lớp, Niên-khoa, Số-học-viên, Mã-khoa)

Lược đồ LỚP-HỌC có khóa là Mã-lớp và một số siêu khóa sau:

$$K1 = \{ \text{Mã-lớp}, \text{Tên-lớp} \}$$

$$K2 = \{ \text{Mã-lớp}, \text{Tên lớp}, \text{Số-học-viên} \}$$

$$K3 = \{ \text{Mã-lớp}, \text{Số-học-viên} \}$$

$$K4 = \{ \text{Mã-lớp}, \text{Niên-khoa} \}$$

Ý nghĩa thực tế của khóa là dùng để nhận diện một bộ trong một quan hệ, nghĩa là, khi cần truy tìm một bộ q nào đó ta chỉ cần biết giá trị của thành phần khóa của q là đủ để dò tìm và hoàn toàn xác định được nó trong quan hệ.

Trong thực tế, đối với các loại thực thể tồn tại khách quan (ví dụ: sinh viên, giảng viên, nhân viên, hàng hóa, ...) người thiết kế cơ sở dữ liệu thường gán thêm cho chúng một thuộc tính giả gọi là mã số để làm khóa chỉ định (ví dụ: mã số sinh viên, mã số giảng viên, mã số nhân viên, mã số hàng hóa, ...). Trong khi đó, các lược đồ quan hệ biểu diễn cho sự trừu tượng hóa thường có khóa chỉ định là một tổ hợp của hai hay nhiều thuộc tính của nó.

Trong trường hợp lược đồ quan hệ Q có nhiều khóa chỉ định, khi cài đặt trên một hệ quản trị CSDL người sử dụng có thể chọn một trong số các khóa chỉ định để tạo chỉ mục (*Index*) chi phối việc truy cập đến các bộ giá trị. Khóa chỉ định này được gọi là khóa chính (*Primary key*). Các khóa còn lại gọi là các *khóa tương đương*. Khóa chính chỉ thật sự có ý nghĩa trong quá trình khai thác cơ sở dữ liệu và xét trên phương diện lý thuyết, khóa chính hoàn toàn không có vai trò gì khác so với các khóa chỉ định còn lại.

Một số hệ quản trị CSDL như MicroSoft Access, Paradox, Oracle, Informix, DB2 ... có cài đặt cơ chế tự động kiểm tra tính duy nhất trên khóa chính. Tức là, nếu thêm một bộ mới q_2 có giá trị khóa chính trùng với giá trị khóa chính của một bộ q_1 nào đó đã có trong quan hệ thì hệ thống sẽ báo lỗi và yêu cầu nhập lại một giá trị khác.

Người ta cũng qui ước rằng:

- Trong một bộ của một quan hệ các thuộc tính khóa không chứa giá trị rỗng.
- Không được phép sửa đổi giá trị của thuộc tính khóa. Nếu muốn sửa đổi giá trị thuộc tính khóa của một bộ q , người sử dụng phải hủy bỏ bộ q và sau đó thêm mới một bộ q' với giá trị khóa đã được sửa đổi.

Các thuộc tính có tham gia vào một khóa được gọi là thuộc tính khóa. Trong lược đồ quan hệ, các thuộc tính khóa sẽ được gạch dưới. Ngược lại, các thuộc tính không tham gia vào một khóa nào gọi là thuộc tính không khóa.

Ví dụ 3.1.12:

KHOA (Mã-khoa, Tên-khoa)

LỚP-HỌC (Mã-lớp, Tên-lớp, Niên-khoa, Số-học-viên, Mã-khoa)

MÔN-HỌC (Mã-môn, Tên-môn, Số-đv-học-trình).

HỌC-VIÊN (Mã-học-viên, Tên-học-viên, Ngày-sinh, Quê-quán, Mã-lớp).

GIẢNG-VIÊN (Mã-giảng-viên, Tên-giảng-viên, Cấp-học-vị, Chuyên-ngành).

KQUẢ_THI (Mã-học-viên, Mã-môn, Lần-thi, Ngày-thi, Điểm-thi, Ghi-chú).

* Khóa ngoại (Foreign Key)

Giả sử có hai quan hệ R và S. Một tập thuộc tính K của quan hệ R được gọi là khóa ngoại của quan hệ R nếu K là khóa nội của quan hệ S.

Ví dụ 3.1.13:

Mã-khoa trong quan hệ LỚP-HỌC là khóa ngoại vì nó là khóa nội của quan hệ KHOA.

Mã-lớp trong quan hệ HỌC-VIÊN là khóa ngoại của quan hệ HỌC-VIÊN vì nó là khóa nội của quan hệ LỚP-HỌC.

3.1.7. Phụ thuộc hàm (Functional Dependency)

Quan hệ R được định nghĩa trên tập thuộc tính $U = \{ A_1, A_2, \dots, A_n \}$. $X, Y \subset U$ là 2 tập con của tập thuộc tính U .

Nếu tồn tại một ánh xạ $f: X \rightarrow Y$ thì ta nói rằng X xác định hàm Y, hay Y phụ thuộc hàm vào X và ký hiệu là $X \rightarrow Y$. X được gọi là vế trái của phụ thuộc hàm (*Determinant of Y*). Chúng ta sẽ tìm hiểu kỹ hơn về phụ thuộc hàm trong Chương IV, mục 4.3, Bài 5.

3.1.8. Ràng buộc toàn vẹn (Integrity Constraint, Rule)

Ràng buộc toàn vẹn (viết tắt là RBTV) là một quy tắc định nghĩa trên một (hay nhiều) quan hệ do môi trường ứng dụng quy định. Đó chính là quy tắc để đảm bảo tính nhất quán của dữ liệu trong CSDL.

Mỗi RBTV được định nghĩa bằng một thuật toán trong CSDL.

Ví dụ 3.1.16:

Quan hệ CCVC (Mã-CBVC, Họ-tên-CBVC, Hết-số-lương)

- *Quy tắc:* Hết số lương của cán bộ viên chức (CBVC) phải lớn hơn hay bằng 1.00 và nhỏ hơn hay bằng 10.00.

- *Thuật toán:* $\forall cc \in CCVC$ thì $cc.Hết-số-lương \geq 1$ & $cc.Hết-số-lương \leq 10$.

Các khái niệm cũng như các vấn đề chủ yếu của RBTV sẽ được trình bày chi tiết trong Chương IV.

3.2. Các thao tác cơ bản trên các quan hệ.

Trong chương này chúng ta chỉ đề cập tới những khái niệm cơ bản do đó các phép toán khác trên các quan hệ sẽ được trình bày chi tiết trong chương V. Ba thao tác cơ bản trên một quan hệ, mà nhờ đó CSDL được thay đổi, đó là Thêm (*Insert*), Xóa (*Delete*) và Sửa (*Update*) các bộ giá trị của quan hệ.

3.2.1 Phép thêm một bộ mới vào quan hệ.

Việc thêm một bộ giá trị mới t vào quan hệ R (A_1, A_2, \dots, A_n) làm cho thể hiện T_R của nó tăng thêm một phần tử mới: $T_R = T_R \cup t$. Dạng hình thức của phép thêm bộ mới là:

$INSERT (R; A_{i1}=v_1, A_{i2}=v_2, \dots, A_{im}=v_m)$

trong đó, $A_{i1}, A_{i2}, \dots, A_{im}$ là các thuộc tính, và v_1, v_2, \dots, v_m là các giá trị thuộc $MGT(A_{i1}), MGT(A_{i2}), \dots, MGT(A_{im})$ tương ứng.

Cần lưu ý rằng các thuộc tính không có tên trong danh sách gán giá trị của bộ t trong câu lệnh *INSERT* sẽ có giá trị là *NULL*, tức là giá trị không xác định.

Ví dụ 3.1.17:

Quan hệ:

HỌC-VIÊN (Mã-học-viên, Tên-học-viên, Ngày-sinh, Quê-quán, Mã-lớp).

Thêm bộ mới:

$q5 = (\text{SV002}, \text{Hoàng Thị Chính}, 17/05/1967, \text{Hà nội}, \text{QTKD1})$

vào quan hệ HỌC-VIÊN bởi phép thêm như sau:

$\text{INSERT } (\text{HỌC-VIÊN}; [\text{Mã-học-viên}]=\text{Hoàng Thị Chính}, [\text{Ngày-sinh}]=17/05/1967, [\text{Quê-quán}]=\text{Hà nội}, [\text{Mã-lớp}]=\text{QTKD1}).$

Thể hiện T_{HỌC-VIÊN} giờ đây là:

$q1 = (\text{SV001}, \text{Nguyễn Văn Nam}, 27/03/1970, \text{Cần Thơ}, \text{QTKD1})$
$q5 = (\text{SV002}, \text{Hoàng Thị Chính}, 17/05/1967, \text{Hà nội}, \text{QTKD1})$
$q2 = (\text{SV005}, \text{Vũ Thị Tuyết Mai}, 26/02/1968, \text{Đồng Nai}, \text{KTKC1})$
$q3 = (\text{SV014}, \text{Hồng Đăng}, 30/04/1975, \text{Đồng Nai}, \text{CNTK3})$
$q4 = (\text{SV015}, \text{Lê Hoài Nhớ}, 23/03/1965, \text{Long An}, \text{CNTK4})$

Xin lưu ý là quan hệ HỌC-VIÊN có khóa là Mã-học-viên, do đó bản ghi mới $q5$ được đẩy lên vị trí thứ 2 theo thứ tự tăng dần giá trị của khóa. Cũng vì lý do này, phép thêm bản ghi mới còn được gọi là phép chèn. [7].

Nếu xem thứ tự của các thuộc tính là cố định và giá trị v_1, v_2, \dots, v_m là hoàn toàn tương ứng thì phép chèn có thể viết dưới dạng tường minh như sau:

$\text{INSERT } (R; v_1, v_2, \dots, v_m).$

Phép chèn có thể không thực hiện được hoặc làm mất tính nhất quán của dữ liệu trong CSDL vì các lý do:

- Giá trị khóa của bộ mới là rỗng (NULL) hoặc trùng với giá trị khóa của một bộ đã có trong CSDL. Trong trường hợp này hệ quản trị CSDL không cho bổ sung.

- Bộ mới không phù hợp với lược đồ quan hệ. Trường hợp này có thể xảy ra khi người sử dụng lầm lẫn thứ tự, kiểu hoặc độ lớn của các thuộc tính. Hệ quản trị CSDL có thể không cho bổ sung nếu không tương thích kiểu giá trị, hoặc vẫn cho bổ sung bộ mới nhưng tính nhất quán dữ liệu không được đảm bảo.

- Một số giá trị của bộ mới không thuộc miền giá trị của thuộc tính tương ứng. Trong trường hợp này, nếu quan hệ đã được đảm bảo tính nhất quán bởi các RBTv về miền giá trị thì hệ quản trị CSDL sẽ không cho bổ sung, nếu không có RBTv như vậy thì tính nhất quán của dữ liệu bị vi phạm mà hệ quản trị CSDL không phát hiện được.

3.2.2 Phép loại bỏ bộ khỏi quan hệ.

Phép loại bỏ (hoặc xóa bỏ) một bộ t của quan hệ sẽ lấy đi (những) bộ t khỏi thể hiện của quan hệ. $T_R = T_R \setminus t$. Phép loại bỏ được viết một cách hình thức như sau:

DELETE (R; $A_{i1}=v_1, A_{i2}=v_2, \dots A_{im}=v_m$).

trong đó $A_{ij}=v_j$ ($j = 1, 2, \dots, m$) được coi như những điều kiện thỏa một số thuộc tính của bộ t để loại bỏ một bộ ra khỏi quan hệ.

Ví dụ 3.1.18:

Quan hệ:

HỌC-VIÊN (Mã-học-viên, Tên-học-viên, Ngày-sinh, Quê-quán, Mã-lớp)

Với phép loại bỏ như sau:

DELETE (HỌC-VIÊN; [Quê-quán]=Đồng nai).

Thì các bộ:

q2 = (SV005, Vũ Thị Tuyết Mai, 26/02/1968, Đồng Nai, KTKC1)
q3 = (SV014, Hồng Đăng, 30/04/1975, Đồng Nai, CNTK3)

sẽ bị loại bỏ ra khỏi quan hệ HỌC-VIÊN bởi vì cùng có chung Quê-quán là Đồng nai. Thể hiện T_{HỌC-VIÊN} lúc này là:

q1 = (SV001, Nguyễn Văn Nam, 27/03/1970, Cần Thơ, QTKD1)
q5 = (SV002, Hoàng Thị Chính, 17/05/1967, Hà nội, QTKD1)
q4 = (SV015, Lê Hoài Nhớ, 23/03/1965, Long An, CNTK4)

3.2.3 Phép sửa đổi giá trị của các thuộc tính của quan hệ.

Dữ liệu của CSDL đôi khi cũng cần phải được đổi mới theo thời gian hoặc sửa lại cho đảm bảo tính chính xác hoặc nhất quán của dữ liệu. Do đó thao tác sửa dữ liệu (*Update*) là rất cần thiết. Một số hệ quản trị CSDL đưa ra nhiều câu lệnh khác nhau để sửa đổi dữ liệu: EDIT, CHANGE, BROW, UPDATE (như DBase, FoxPro v.v...). Trong ngôn ngữ hình thức, mục này đưa ra một dạng của phép sửa đổi giá trị các bộ của quan hệ:

UPDATE (R; A_{i1}=c₁, A_{i2}=c₂, ... A_{im}= c_m; A_{i1}=v₁, A_{i2}=v₂, ... A_{im}= v_m).

Trong đó R là quan hệ cần thực hiện sửa đổi; A_{ij}= c_j (j = 1, 2, ..., m) là điều kiện tìm kiếm bộ giá trị để sửa và A_{ij}= v_j (j = 1, 2, ..., m) là giá trị mới của bộ.

Ví dụ 3.1.19:

Quan hệ

HỌC-VIÊN (Mã-học-viên, Tên-học-viên, Ngày-sinh, Quê-quán, Mã-lớp)

Với phép sửa đổi giá trị như sau:

UPDATE (HỌC-VIÊN; [Mã-học-viên]=SV015, [Quê-quán]=Sông Bé)

thì giá trị của bộ q4 được sửa lại thành:

q4 = (SV015, Lê Hoài Nhớ, 23/03/1965, Sông Bé, CNTK4)

CHƯƠNG IV

RÀNG BUỘC TOÀN VẸN TRÊN MỘT CSDL

Ràng buộc toàn vẹn (*Integrity Constraint / Rule*) viết tắt là: RBTV) và kiểm tra sự vi phạm ràng buộc toàn vẹn là hai trong những vấn đề rất quan trọng trong quá trình phân tích, thiết kế và khai thác CSDL. Trong quá trình phân tích - thiết kế cơ sở dữ liệu, nếu không quan tâm đúng mức đến những vấn đề trên, thì có thể dẫn đến những hậu quả rất nghiêm trọng về tính an toàn và toàn vẹn dữ liệu, đặc biệt trong những CSDL tương đối lớn.

Chương này được chia làm 3 mục nhỏ: Mục thứ nhất trình bày các khái niệm cơ bản của ràng buộc toàn vẹn; mục thứ hai sẽ giới thiệu các cách phân loại ràng buộc toàn vẹn và phần thứ ba sẽ đề cập đến một công cụ để biểu diễn một số các ràng buộc toàn vẹn: Phụ thuộc hàm.

4.1. Ràng buộc toàn vẹn, các yếu tố của RBTV.

4.1.1. Định nghĩa.

Ràng buộc toàn vẹn là một điều kiện bất biến không được vi phạm trong một CSDL.

Trong một CSDL, luôn luôn tồn tại rất nhiều mối liên hệ ảnh hưởng qua lại lẫn nhau giữa các thuộc tính của một quan hệ, giữa các bộ giá trị trong một quan hệ và giữa các thuộc tính của các bộ giá trị trong các quan hệ với nhau. Các mối quan hệ phụ thuộc lẫn nhau này chính là những điều kiện bất biến mà tất cả các bộ của những quan hệ có liên quan trong cơ sở dữ liệu đều phải thỏa mãn ở bất kỳ thời

điểm nào. Ràng buộc toàn vẹn còn được gọi là các quy tắc quản lý (*Rules*) được áp đặt lên trên các đối tượng của thế giới thực.

Ví dụ 4.1.1:

Trong CSDL về quản lý học viên của một trường học đã cho trong các ví dụ của mục 3.1 trong Chương III (Bài 4), chúng ta có một số ràng buộc toàn vẹn như sau:

- R_1 : Mỗi lớp học phải có một mã số duy nhất để phân biệt với mọi lớp học khác trong trường.
- R_2 : Mỗi lớp học phải thuộc một KHOA của trường.
- R_3 : Mỗi học viên có một mã số riêng biệt, không trùng với bất cứ học viên nào khác.
- R_4 : Mỗi học viên phải đăng ký vào một lớp của trường.
- R_5 : Mỗi học viên được thi tối đa 3 lần cho mỗi môn học.
- R_6 : Tổng số học viên của một lớp phải lớn hơn hoặc bằng số lượng đếm được của lớp tại một thời điểm.

Khóa nội, Khóa ngoại, giá trị NOT NULL ... là những RBTv về miền giá trị của các thuộc tính. Những RBTv vừa nêu trên cũng mới chỉ là những RBTv đơn giản trong CSDL nhỏ về quản lý học viên. Trong thực tế, tất cả các RBTv của một cơ sở dữ liệu phải được người phân tích thiết kế phát hiện đầy đủ và mô tả một cách chính xác, rõ ràng trong hồ sơ phân tích, thiết kế.

Trong một CSDL, ràng buộc toàn vẹn được xem như một công cụ để diễn đạt ngữ nghĩa của cơ sở dữ liệu đó. Trong suốt quá trình khai thác cơ sở dữ liệu, các RBTv đều phải được thỏa mãn ở bất kỳ thời điểm nào nhằm đảm bảo cho CSDL luôn luôn ở trạng thái an toàn và nhất quán về dữ liệu.

Các hệ quản trị CSDL thường có các cơ chế tự động kiểm tra các RBTv về miền giá trị của Khóa nội, Khóa ngoại, giá trị NOT

NULL qua khai báo cấu trúc các bảng (mô hình dữ liệu của quan hệ) hoặc thông qua những thủ tục kiểm tra và xử lý vi phạm RBTV do những người phân tích - thiết kế cài đặt. Việc kiểm tra RBTV có thể được tiến hành vào một trong các thời điểm sau:

- Kiểm tra ngay khi thực hiện một thao tác cập nhật CSDL (thêm, sửa, xóa). Thao tác cập nhật chỉ được xem là hợp lệ nếu như nó không vi phạm bất cứ một RBTV nào, nghĩa là nó không làm mất tính toàn vẹn dữ liệu của CSDL. Nếu vi phạm RBTV, thao tác cập nhật bị coi là không hợp lệ và sẽ bị hệ thống hủy bỏ (hoặc có một xử lý thích hợp nào đó).

- Kiểm tra định kỳ hay đột xuất, nghĩa là việc kiểm tra RBTV được tiến hành một cách độc lập đối với thao tác cập nhật dữ liệu. Đối với những trường hợp vi phạm RBTV, hệ thống sẽ có những xử lý ngầm định hoặc yêu cầu người sử dụng xử lý những sai sót một cách tường minh.

4.1.2. Các yếu tố của ràng buộc toàn vẹn:

Khi xác định một RBTV cần chỉ rõ:

- (1) Điều kiện (tức là nội dung) của RBTV, từ đó xác định cách biểu diễn.
- (2) Bối cảnh xảy ra RBTV : trên một hay nhiều quan hệ, cụ thể trên các quan hệ nào.
- (3) Tầm ảnh hưởng của RBTV. Khả năng tính toàn vẹn dữ liệu bị vi phạm, và
- (4) Hành động cần phải có khi phát hiện có RBTV bị vi phạm.

4.1.2.1 Điều kiện của RBTV:

Điều kiện của RBTV là sự mô tả, và biểu diễn hình thức nội dung của nó, có thể được biểu diễn bằng ngôn ngữ tự nhiên, thuật giải (bằng mã giả - *Pseudo Code*, ngôn ngữ tựa Pascal), ngôn ngữ đại số tập hợp, đại số quan hệ v.v hoặc bằng các phụ thuộc hàm (sẽ được trình bày chi tiết trong mục 3 của chương này).

Ví dụ 4.1.2:

Giả sử có một CSDL quản lý hóa đơn bán hàng gồm các bảng sau:

HÓA ĐƠN (Số-hóa đơn, Số-chủng-loại-mặt-hàng, Tổng-trị-giá).

DM_HÀNG (Mã-hàng, Tên-hàng, Đơn-vị-tính).

CHITIẾT_HĐ (Số-hóa đơn, Mã-hàng, Số-lượng-đặt, Đơn-giá, Trị-giá).

Điều kiện của ràng buộc toàn vẹn có thể biểu diễn như sau:

(-) R₁ : “Mỗi hóa đơn có một Số hóa đơn riêng biệt, không trùng với hóa đơn khác”:

$$\forall hđ_1, hđ_2 \in \text{HÓA ĐƠN}, hđ_1 \neq hđ_2 \Rightarrow hđ_1.\text{Số-hóa đơn} \neq hđ_2.\text{Số-hóa đơn}.$$

(-) R₂ : “Số-chủng-loại-mặt-hàng = số bộ của CHITIẾT_HĐ có cùng Số-hóa đơn”:

$\forall hđ \in \text{HÓA ĐƠN}$ thì:

$$hđ.\text{Số-chủng-loại-mặt-hàng} = \text{COUNT} (cthđ \in \text{CHITIẾT_HĐ}, cthđ.Số-hóa đơn = hđ.Số-hóa đơn)$$

(-) R₃ : “Tổng các trị giá của các mặt hàng trong CHITIẾT_HĐ có cùng Số-hóa đơn phải bằng Tổng-trị-giá ghi trong HÓA ĐƠN”:

$\forall hđ \in \text{HÓA ĐƠN}$ thì:

$$hđ.\text{Tổng-trị-giá} = \text{SUM} (cthđ.Trị-giá) \text{ đối với các } cthđ \in \text{CHITIẾT_HĐ} \\ \text{sao cho: } cthđ.\text{Số-hóa đơn} = hđ.\text{Số-hóa đơn}.$$

(-) R₄ : “Mỗi bộ của CHITIẾT_HĐ phải có mã hàng thuộc về danh mục hàng”:

$$\text{CHITIẾT_HĐ} [\text{Mã-hàng}] \subseteq \text{DM_HÀNG}[\text{Mã-hàng}]$$

hoặc biểu diễn bằng cách khác:

$\forall cthd \in CHIET_HD, \exists hh \in DM_HANG$
sao cho: $cthds.Mahang = hh.Mahang$.

☞ Ghi chú: Chúng ta quy ước: chữ COUNT (...) nghĩa là đếm số bộ giá trị của một quan hệ thỏa mãn điều kiện đã cho trong dấu ngoặc tròn. Chữ SUM (...) nghĩa là lấy tổng các giá trị của các bộ trên một quan hệ thỏa mãn điều kiện đã cho trong dấu ngoặc tròn. Các ngôn ngữ quản trị CSDL đều có các thủ tục hàm này để hỗ trợ tính toán. Chương VI sẽ trình bày ngôn ngữ truy vấn CSDL có cấu trúc (SQL) trong đó có đề cập tới các hàm này.

4.1.2.2 Bối cảnh của RBTM :

Bối cảnh có thể định nghĩa trên một quan hệ cơ sở hay nhiều quan hệ cơ sở. Đó là những quan hệ mà RBTM được áp dụng trên đó.

Như trong ví dụ 4.1.2, bối cảnh của ràng buộc toàn vẹn R_1 chỉ là một quan hệ HÓAĐƠN; bối cảnh của ràng buộc toàn vẹn R_2 và R_3 là hai quan hệ HÓAĐƠN và CHIET_HD; bối cảnh của ràng buộc toàn vẹn R_4 là hai quan hệ CHIET_HD và DM_HÀNG.

4.1.2.3 Xác định tầm ảnh hưởng của RBTM:

Một RBTM có thể liên quan đến một số quan hệ, và chỉ khi có thao tác cập nhật (Thêm, Sửa, Xóa) mới có nguy cơ dẫn đến vi phạm RBTM, do đó cần xác định rõ thao tác nào dẫn đến việc cần phải kiểm tra RBTM.

Trong quá trình phân tích, thiết kế một CSDL, người phân tích cần lập bảng xác định tầm ảnh hưởng cho mỗi ràng buộc toàn vẹn nhằm xác định khi nào thì phải tiến hành kiểm tra các ràng buộc toàn

vẹn đó. Bảng này gồm 4 cột: cột 1 là cột chủ từ chứa tên các quan hệ liên quan tới RBTV; 3 cột tiếp theo là thao tác Thêm / Sửa / Xóa bô giá trị của quan hệ. Nếu RBTV cần được kiểm tra nguy cơ dẫn tới vi phạm thì tại ô (giao điểm dòng và cột) đó người ta đánh dấu bằng dấu gạch chéo (x) hoặc dấu cộng (+), và có thể chỉ rõ thêm các thuộc tính nào nếu được cập nhật mới dẫn đến vi phạm RBTV bằng cách liệt kê chúng dưới dấu (x) hoặc dấu (+). Nếu RBTV không có nguy cơ bị vi phạm khi cập nhật CSDL thì đánh dấu trừ (-) vào ô tương ứng. Nếu không bị vi phạm vì không được phép sửa đổi thì ký hiệu là trừ với dấu sao ($-^{(*)}$).

Ví dụ 4.1.3:

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_1

<i>Quan hệ</i>	<i>Thêm</i>	<i>Sửa</i>	<i>Xóa</i>
HÓAĐƠN	+ (Số-hóa-dơn)	- (*)	+

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_2

<i>Quan hệ</i>	<i>Thêm</i>	<i>Sửa</i>	<i>Xóa</i>
HÓAĐƠN	-	+ (Số-chủng-loại-mặt-hàng)	+
CHITIẾT_HĐ	+	-	+

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_3

<i>Quan hệ</i>	<i>Thêm</i>	<i>Sửa</i>	<i>Xóa</i>
HÓAĐƠN	-	+ (Tổng-tri-giá)	+
CHITIẾT_HĐ	+	+ (Trị-giá)	+

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_4

<i>Quan hệ</i>	<i>Thêm</i>	<i>Sửa</i>	<i>Xóa</i>
CHITIẾT_HĐ	+ (Mã-hàng)	- (*)	+
DM_HÀNG	-	- (*)	+

Trong thực tế, ràng buộc toàn vẹn R_1 là không cần thiết, bởi thuộc tính Số-hóa-đơn là khóa của quan hệ HÓAĐƠN, do vậy nó luôn luôn phải là duy nhất và không được phép chứa giá trị rỗng; đồng thời không được phép sửa đổi (Xem lại Chương III, mục 3.1, điểm 3.1.6 - *một số quy ước về khóa*).

Sau khi xây dựng các bảng tầm ảnh hưởng của từng RBTW trên các quan hệ liên quan, cần phải tổng hợp lại bằng cách xây dựng một bảng tầm ảnh hưởng tổng hợp các RBTW nhằm xác định tất cả các RBTW cần phải kiểm tra trên từng quan hệ. Bảng này gồm cột chủ từ là các RBTW, các cột còn lại là các thao tác Thêm (T), Sửa (S) và Xóa (X) của từng quan hệ nằm trong bối cảnh của các RBTW trong CSDL.

Ví dụ 4.1.4:

Lập bảng tầm ảnh hưởng tổng hợp của các RBTW trong CSDL quản lý hóa đơn bán hàng nêu trên:

Q.HỆ		HÓAĐƠN			CHITIẾT_HĐ			DM_HÀNG		
RBTW	T	S	X	T	S	X	T	S	X	
R_1	+ (Số-hd)	- (*)	+							
R_2	-	+ (Số-loại-MH)	+	+	- (*)	+				
R_3	-	+ (Tổng-TG)	+	+	+ (Trị-giá)	+				
R_4			+		-	+	-	- (*)	+	

Nhìn vào bảng tổng hợp trên chúng ta có thể thấy quan hệ HÓAĐƠN khi thêm và xóa một bộ giá trị, phải kiểm tra ràng buộc toàn vẹn R_1 , R_2 và R_3 ; khi sửa giá trị thuộc tính Số-chủng-loại-mặt-hàng thì phải kiểm tra ràng buộc toàn vẹn R_2 và khi sửa giá trị thuộc tính Tổng-trị-giá thì phải kiểm tra ràng buộc toàn vẹn R_3 . Quan hệ CHITIẾT_HĐ khi được cập nhật cần kiểm tra 2 RBTW: R_2 và R_3 ; Quan

hệ DM_HÀNG cần kiểm tra ràng buộc toàn vẹn R₄ khi xóa một bộ giá trị.

4.1.2.4 Hành động khi RBTV bị vi phạm:

Khi một RBTV bị vi phạm cần có những hành động thích hợp. Thông thường có 2 giải pháp:

(1) Đưa ra thông báo và yêu cầu sửa chữa dữ liệu của các thuộc tính cho phù hợp với quy tắc đảm bảo tính nhất quán dữ liệu. Thông báo phải đầy đủ và tạo được sự thân thiện với người sử dụng. Giải pháp này là phù hợp cho việc xử lý thời gian thực.

(2) Từ chối thao tác cập nhật. Giải pháp này là phù hợp đối với việc xử lý theo lô (*Batch processing*). Việc từ chối cũng phải được lưu lại bằng những thông báo đầy đủ, rõ ràng vì sao thao tác bị từ chối và cần phải sửa lại những dữ liệu nào.

4.2 Phân loại ràng buộc toàn vẹn.

Trong một CSDL có thể phát hiện nhiều RBTV, tuy nhiên có thể chia chúng thành hai loại chính theo bối cảnh RBTV:

- i) Ràng buộc toàn vẹn trong bối cảnh là một quan hệ cơ sở.
- ii) Ràng buộc toàn vẹn có bối cảnh trên nhiều quan hệ cơ sở.

Chúng ta sẽ xem xét chi tiết các loại RBTV chi tiết trong từng bối cảnh nêu trên.

4.2.1 Ràng buộc toàn vẹn có bối cảnh là 1 quan hệ cơ sở.

Trên một quan hệ cơ sở có thể tồn tại nhiều RBTV thuộc các loại: RBTV về miền giá trị của thuộc tính, RBTV về giá trị giữa thuộc tính này với (các) thuộc tính khác (*gọi là liên thuộc tính*) và giữa các giá trị của các bộ giá trị khác nhau (*gọi là liên bộ - liên thuộc tính*)

4.2.1.1 Ràng buộc toàn vẹn về miền giá trị của thuộc tính.

Trong hầu hết các CSDL quan hệ, loại RBTV này là rất phổ biến. Như chúng ta đã biết, thuộc tính được đặc trưng không chỉ bởi kiểu giá trị mà nó còn bị giới hạn bởi miền giá trị trong kiểu dữ liệu đó. Do đó, khi thực hiện các thao tác cập nhật Thêm / Sửa bộ giá trị mới cho quan hệ đều phải kiểm tra RBTV này.

Ví dụ 4.2.1:

Trong quan hệ KQUẢ-THI mô tả trong ví dụ 3.1.12, do quy định mỗi học viên chỉ được thi một môn học tối đa là 3 lần, hiển nhiên là điểm thi của mỗi môn học trong mọi lần thi không bị âm và không vượt quá 10. Có 2 ràng buộc toàn vẹn về miền giá trị trong quan hệ này:

$$R_1: \forall kq \in KQUẢ-THI \text{ thì } 0 \leq kq.\text{Lần-thi} \leq 3$$

$$R_2: \forall kq \in KQUẢ-THI \text{ thì } 0 \leq kq.\text{Điểm-thi} \leq 10$$

Giả sử các giảng viên có “châm chước” thêm rằng điểm thi lần sau không nhỏ hơn điểm thi lần trước đó. Chúng ta có thêm ràng buộc toàn vẹn về miền giá trị:

$$R_3: \forall kq \in KQUẢ-THI / kq.\text{Điểm-thi} (\text{lần trước}) \leq kq.\text{Điểm-thi} \leq 10.0$$

4.2.1.2 Ràng buộc toàn vẹn liên thuộc tính.

Đó là loại RBTV có liên quan tới nhiều thuộc tính của một quan hệ. Thông thường đó là các phụ thuộc tính toán, hoặc một suy diễn từ giá trị của một hay nhiều thuộc tính trong cùng một bộ giá trị.

Ví dụ 4.2.2:

Quan hệ CHITIẾT_HĐ trong CSDL quản lý hóa đơn bán hàng cho trong ví dụ 4.1.2 tại mục 4.1 trình bày trên, có RBTV liên thuộc tính là:

$$\forall cthđ \in \text{CHITIẾT_HĐ} / cthđ.\text{Trị-giá} = cthđ.\text{Số-lượng-đặt} * cthđ.\text{Đơn-giá}.$$

Ví dụ 4.2.3:

Quan hệ danh sách cán bộ - công chức Nhà nước CBCC với tập các thuộc tính: { Mã-đơn-vị, Mã-CBCC, Họ-tên, Giới-tính, Ngày-sinh, Ngày-tuyển-dụng, Ngạch-CBCC, Bậc, Hệ-số-lương, Ngày-xếp-lương }.

Với quy định nam từ 18 đến 60 và nữ từ 18 đến 55 tuổi và phải từ 18 tuổi trở lên mới được tuyển vào làm công chức Nhà nước. Chúng ta có các RBTV về miền giá trị liên thuộc tính như sau:

R₁: $\forall cc \in \text{CBCC} / \text{nếu } cc.\text{Giới-tính} = \text{Nam} \text{ thì } (\text{Now}() - cc.\text{Ngay_sinh}) / 365 \text{ trong khoảng 18 và 60. Nếu } cc.\text{Giới-tính} = \text{Nữ} \text{ thì } (\text{Now}() - cc.\text{Ngay_sinh}) / 365 \text{ trong khoảng 18 và 55.}$

R₂: $\forall cc \in \text{CBCC} / (cc.\text{Ngày-tuyển-dụng} - cc.\text{Ngày-sinh}) / 365 \geq 18 \text{ và } cc.\text{Ngày-tuyển-dụng} \leq \text{Now}().$

Ghi chú:

- Now() là lấy ngày tháng năm hiện tại và một năm trung bình có 365 ngày;
- Hiệu 2 giá trị ngày tháng là số ngày trôi qua giữa 2 ngày đó.

4.2.1.3 Ràng buộc toàn vẹn liên bộ, liên thuộc tính.

Đây là loại RBTV có liên quan tới nhiều bộ và có thể tới nhiều thuộc tính của (các) bộ giá trị trong một quan hệ.

Ví dụ 4.2.4:

Trong ví dụ 4.2.1 vừa nêu, chúng ta thấy điểm thi không chỉ liên quan đến thuộc tính Lần-thi mà còn liên quan tới điểm thi của lần thi trước đó nếu đã thi 1 hay 2 lần rồi. RBTV đầy đủ phải được diễn đạt bằng thuật toán như sau:

R3: $\forall kq \in KQUA\text{-THI} / \text{Nếu } kq.\text{Lần-thi} = 1 \text{ thì } 0 \leq kq.\text{Điểm-thi} \leq 10.0$

hoặc:

$\text{Nếu } kq.\text{Lần thi} > 1 \text{ thì } \exists kq' \in KQUA\text{-THI}$

sao cho $kq'.\text{Lần-thi} = kq.\text{Lần-thi} - 1$ và $kq.\text{Điểm-thi} \geq kq'.\text{Điểm-thi}$.

Ví dụ 4.2.5:

Giả thiết thêm rằng, để xác định hệ số lương của một cán bộ - công chức Nhà nước căn cứ vào Ngạch công chức và Bậc được xếp, chúng ta có thêm bảng quy định Ngạch, bậc và hệ số lương cán bộ công chức Nhà nước theo Nghị định 25CP của Thủ tướng Chính phủ:

NGẠCH-BẬC-LƯƠNG (Mã-ngạch, Bậc, Hệ-số-lương).

Tân từ: Ứng với một Ngạch công chức và một Bậc cụ thể thì có một hệ số lương tương ứng (từ 1.0 đến 10.0).

Việc biểu diễn ràng buộc toàn vẹn Hệ-số-lương phụ thuộc vào Ngạch và Bậc của quan hệ CBCC nêu trong ví dụ 4.2.3 nêu trên bằng thuật giải có thể trở nên rắc rối. Người ta đã đưa thêm một cách biểu diễn mới để làm cho RBTV trở nên rõ ràng hơn, đó là cách biểu diễn RBTV bằng phụ thuộc hàm mà chúng ta sẽ trình bày rõ hơn trong mục 4.3 của chương này.

4.2.2 Ràng buộc toàn vẹn định nghĩa trên nhiều quan hệ cơ sở.

4.2.2.1 Ràng buộc toàn vẹn về phụ thuộc tồn tại.

Ràng buộc toàn vẹn về phụ thuộc tồn tại (*Existential Dependency* hay *Referential Dependency*) còn được gọi là *phụ thuộc về khóa ngoại*. Đây là loại RBTV khá phổ biến trong các CSDL bởi các quan hệ trong một CSDL luôn luôn có mối quan hệ mật thiết với nhau. Bộ giá trị của quan hệ này được thêm vào một cách hợp lệ nếu tồn tại một bản ghi tương ứng của một quan hệ khác.

Phụ thuộc tồn tại xảy ra nếu có một trong hai trường hợp sau:

- (i) Có sự hiện diện của khóa ngoại.
- (ii) Có sự lồng khóa giữa các quan hệ.

Ví dụ 4.2.6:

Trong thể hiện của quan hệ CHITIẾT-HĐ, sự tồn tại của mỗi bộ giá trị $cthd$ đều phụ thuộc vào sự tồn tại của một bộ giá trị hd trong thể hiện của quan hệ HÓAĐƠN sao cho $hd.Số-hóa-đơn = cthd.Số-hóa-đơn$, và phụ thuộc cả vào sự tồn tại của một bộ giá trị mh trong thể hiện của quan hệ DM_HÀNG sao cho $mh.Mã-hàng = cthd.Mã-hàng$.

Biểu diễn các RBTV này như sau:

- RBTV₁ : “Mỗi bộ của CHITIẾT_HĐ phải có một hóa đơn với Số-hóa-đơn tương ứng”:

$$\forall cthd \in \text{CHITIẾT_HĐ}, \exists hd \in \text{HÓAĐƠN} \\ \text{sao cho } cthd.Số-hóa-đơn = hd.Số-hóa-đơn.$$

hoặc biểu diễn bằng cách khác:

$$\text{CHITIẾT_HĐ} [\text{Số-hóa-đơn}] \subseteq \text{HÓAĐƠN} [\text{Số-hóa-đơn}]$$

- RBTV₂ : “Mỗi bộ của CHITIẾT_HĐ phải có mã hàng thuộc về danh mục hàng”:

$\forall \text{cth}\in \text{CHITIẾT_HĐ}, \exists \text{hh} \in \text{DM_HÀNG} \text{ sao cho } \text{cth}.Mã-hàng = \text{hh}.Mã-hàng$
hoặc biểu diễn bằng cách khác:

$$\text{CHITIẾT_HĐ} [\text{Mã-hàng}] \subseteq \text{DM_HÀNG} [\text{Mã-hàng}]$$

Ví dụ 4.2.7 :

Trong CSDL về quản lý CBCC nêu trong ví dụ 4.2.3 và 4.2.5 ở trên, RBTV về phụ thuộc tồn tại giữa 2 quan hệ CBCC và NGẠCH-BẬC-LƯƠNG được xác định:

$$\forall \text{cbcc} \in \text{CBCC}, \exists \text{ng} \in \text{NGẠCH-BẬC-LƯƠNG}$$

sao cho $(\text{cbcc}.Mã-ngạch = \text{ng}.Mã-ngạch) \wedge (\text{cbcc}.Bậc = \text{ng}.Bậc)$

hoặc biểu diễn bằng cách khác:

$$\text{CBCC} [\text{Mã-ngạch}, \text{Bậc}] \subseteq \text{NGẠCH-BẬC-LƯƠNG} [\text{Mã-ngạch}, \text{Bậc}]$$

Ví dụ 4.2.8 :

Trong CSDL về quản lý học viên đã nêu trong ví dụ 3.1.12, các RBTV về phụ thuộc tồn tại gồm:

- RBTV₁ : “Mỗi LỚP-HỌC đều phải thuộc một KHOA nhất định”:
 $\forall \text{lh} \in \text{LỚP-HỌC}, \exists \text{kh} \in \text{KHOA} \text{ sao cho } \text{lh}.Mã-khoa = \text{kh}.Mã-khoa.$

hoặc biểu diễn qua phép chiếu quan hệ:

$$\text{LỚP-HỌC} [\text{Mã-khoa}] \subseteq \text{KHOA} [\text{Mã-khoa}].$$

- RBTV₂ : “Mỗi HỌC-VIÊN đều phải thuộc một LỚP-HỌC nhất định”:

$$\forall \text{hv} \in \text{HỌC-VIÊN}, \exists \text{lh} \in \text{LỚP-HỌC} \text{ sao cho } \text{hv}.Mã-lớp = \text{lh}.Mã-lớp.$$

hoặc biểu diễn qua phép chiếu quan hệ:

$$\text{HỌC-VIÊN} [\text{Mã-lớp}] \subseteq \text{LỚP-HỌC} [\text{Mã-lớp}].$$

- RBT₃ : “Mỗi KQUẢ-THI đều phải là của một HỌC-VIÊN nhất định”:

$$\forall kq \in KQUẢ-THI, \exists hv \in HỌC-VIÊN
sao cho kq.Mã-học-viên = hv.Mã-học-viên.$$

hoặc biểu diễn qua phép chiếu quan hệ:

$$KQUẢ-THI [Mã-học-viên] \subseteq HỌC-VIÊN [Mã-học-viên].$$

- RBT₄ : “Mỗi môn thi trong KQUẢ-THI đều phải có tên trong danh sách các môn học” :

$$\forall kq \in KQUẢ-THI, \exists mh \in MÔN-HỌC sao cho kq.Mã-môn = mh.Mã-môn$$

hoặc biểu diễn qua phép chiếu quan hệ:

$$KQUẢ-THI [Mã-môn] \subseteq MÔN-HỌC [Mã-môn]$$

Chúng ta có thể thấy về phái của phép toán tập con (\subseteq) là phép chiếu trên thuộc tính khóa nội của một quan hệ, còn về trái là phép chiếu trên tập các thuộc tính khóa ngoại của một quan hệ khác. Chính vì lẽ đó mà người ta còn gọi RBT₄ loại này là RBT₄ về khóa ngoại. Phát biểu tổng quát về loại RBT₄ này là như sau:

R và S là hai quan hệ định nghĩa trên các tập thuộc tính R^+ và S^+ . $K_R \subseteq R^+$ là tập các thuộc tính khóa nội của quan hệ R; $K_S \subseteq S^+$ là tập các thuộc tính khóa nội của quan hệ S; và $W \subseteq S^+$ là tập các thuộc tính khóa ngoại của S đối với R. Khi đó ta có phụ thuộc tồn tại của S vào R và được biểu diễn thông qua phép chiếu:

$$S [W] \subseteq R [W].$$

Nếu $W \subseteq K_S$, thì ta nói rằng có sự *lồng khóa* giữa hai quan hệ R và S.

Trong bảng tóm ảnh hưởng của loại RBT₄ này, các thao tác *Thêm* và *Sửa* một bộ giá trị của quan hệ R (về phái của phụ thuộc tồn

tại) không gây ra sự vi phạm RBTV (trừ khi có sự lồng khóa giữa R với một quan hệ khác), chỉ có thao tác *Xóa* bỏ một bộ giá trị của R mới cần có sự kiểm tra RBTV. Ngược lại, thao tác *Xóa* một bộ giá trị của S không gây ra sự vi phạm RBTV (trừ khi có sự lồng khóa của một quan hệ khác vào S), thao tác *Thêm* một bộ giá trị mới vào S luôn luôn phải được kiểm tra RBTV này; nếu W là các thuộc tính khóa ngoại của S thì việc *Sửa* bộ giá trị của S trên các thuộc tính khóa ngoại W vẫn phải kiểm tra RBTV; nếu có sự lồng khóa thì việc sửa không đòi hỏi kiểm tra RBTV vì theo quy ước là không được sửa giá trị của thuộc tính khóa.

Bảng tóm ảnh hưởng có 2 dạng ứng với 2 trường hợp trên như sau:

a. Ứng với trường hợp khóa ngoại:

<i>Quan hệ</i>	<i>Thêm</i>	<i>Sửa</i>	<i>Xóa</i>
R	-	- (*)	+
S	+	+ (W)	-

a. Ứng với trường hợp lồng khóa:

<i>Quan hệ</i>	<i>Thêm</i>	<i>Sửa</i>	<i>Xóa</i>
R	-	- (*)	+
S	+	- (*)	-

4.2.2.2 Ràng buộc toàn vẹn liên bộ - liên quan hệ.

Khi có sự hiện diện của 1 thuộc tính mang tính chất tổng hợp (tức là giá trị của thuộc tính có thể được tính toán từ giá trị của các thuộc tính khác trên một hay nhiều bộ giá trị của các quan hệ trong CSDL), hay phụ thuộc tồn tại lồng khóa thì có RBTV liên quan hệ - liên bộ.

Ví dụ 4.2.9 :

Xét CSDL về quản lý học viên nêu trong ví dụ 3.1.12 (Chương III, mục 3.1), RBTV liên quan hệ - liên bộ có thể được xác định: “Với mọi bộ giá trị của LỚP-HỌC, nếu Số lượng học viên lớn hơn 0 thì số lượng này phải lớn hơn hay bằng tổng số bộ giá trị đếm được của các học viên có cùng Mã lớp”. Đây chính là RBTV R_6 đã nêu trong ví dụ 4.1.1.

Biểu diễn hình thức của RBTV này như sau:

$\forall lh \in LỚP-HỌC$ thì:

nếu $lh.\text{Số-học-viên} > 0$ thì:

$lh.\text{Số-học-viên} = \text{COUNT } (hv \in HỌC-VIÊN, hv.\text{Mã-lớp} = lh.\text{Mã-lớp})$.

Ví dụ 4.2.10 :

Xét CSDL quản lý hóa đơn bán hàng đã cho trong ví dụ 4.1.2 với 3 quan hệ:

1) HÓAĐƠN (Số-hóa đơn, Số-chủng_loại_mặt_hàng, Tổng-trị-giá).

2) CHITIẾT_HĐ (Số-hóa đơn, Mã-hàng, Số-lượng-đặt, Đơn-giá, Trị-giá).

3) DM_HÀNG (Mã-hàng, Tên-hàng, Đơn-vị-tính).

- RBTV₁ : “Số_chủng_loại_mặt_hàng = số bộ của CHITIẾT_HĐ có cùng Số-hóa đơn” :

$\forall hd \in HÓAĐƠN$ thì:

$hd.\text{Số-chủng-loại-mặt-hàng} = \text{COUNT } (cthd \in CHITIẾT_HĐ, cthd.\text{Số-hóa đơn} = hd.\text{Số-hóa đơn})$

- RBTV₂ : “Tổng tất cả các Trị-giá của các mặt hàng trong CHITIẾT_HĐ có cùng Số-hóa đơn phải bằng Tổng-trị-giá của hóa đơn đó trong HÓAĐƠN”:

$\forall hd \in HÓAĐƠN$ thì $hd.\text{Tổng-trị-giá} = \text{SUM } (cthd.\text{Trị-giá})$

đối với các $cthd \in CHITIET_HD$
 sao cho : $cthd.Số-hóa-đơn = hd.Số-hóa-đơn.$

Chúng ta có thể nhận thấy trong CSDL này có sự dư thừa thông tin một cách cố ý. Đó là thuộc tính toán *Trị-giá* của các mặt hàng trong chi tiết hoá đơn bán hàng. Một trong những phương pháp kiểm định tính đúng đắn của dữ liệu được nhập vào là tổ chức nhập “*thừa*” dữ liệu tính toán được (*Computable Value*) rồi so sánh với công thức tính toán. Nếu có sự sai sót nào trong các thành phần có liên quan trong công thức thì lôgic biểu thức sẽ không còn phù hợp nữa.

Bây giờ để đạt được dạng chuẩn (*Normal Form*) tốt hơn cho quan hệ CHITIET-HD, chúng ta có thể loại bỏ thuộc tính Trị-giá. Khi đó RBT_{V2} được viết lại là:

$\forall hd \in HÓAĐƠN$ thì
 $hd.Tổng-trị-giá = \text{SUM } (cthd.Số-lượng-đặt * cthd.Đơn-giá)$
 đối với các $cthd \in CHITIET_HD$
 sao cho : $cthd.Số-hóa-đơn = hd.Số-hóa-đơn.$

4.2.2.3 RBT_V do sự hiện diện của chu trình.

Bây giờ chúng ta biểu diễn cấu trúc CSDL dưới dạng đồ thị như sau: Mỗi nút của đồ thị biểu diễn một quan hệ hoặc một thuộc tính.

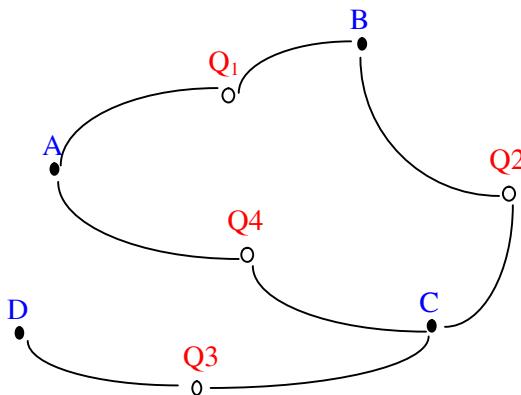
- (i) Quan hệ được biểu diễn bằng nút tròn trắng (o), và
- (ii) Thuộc tính được biểu diễn bởi một nút tròn đen nhỏ hơn (•).
- (iii) Tất cả các nút đều được chỉ rõ bằng tên của quan hệ hoặc thuộc tính. Thuộc tính thuộc một quan hệ được biểu diễn bởi một cung nối giữa nút tròn trắng và nút tròn đen đó.

Nếu trên đồ thị chúng ta thấy xuất hiện một đường khép kín thì ta nói rằng trong lược đồ CSDL có sự hiện diện của chu trình. Sự hiện

diện này làm nảy sinh một vấn đề mới: Xác định khả năng đảm bảo tính nhất quán của dữ liệu, đó là một *RBT* do sự hiện diện của chu trình.

Ví dụ 4.2.11:

Hãy xét lược đồ CSDL với các lược đồ quan hệ con: $Q_1(A, B)$, $Q_2(B, C)$, $Q_3(C, D)$ và $Q_4(A, C)$. Biểu diễn lược đồ CSDL bằng đồ thị như sau:



Hình 4.2.1 Biểu diễn lược đồ CSDL bằng đồ thị để phát hiện chu trình

Với $a \in A$ thì qua quan hệ Q_1 ta có thể xác định được một giá trị $b \in B$. Vẫn với giá trị $a \in A$ đó nhưng xác định qua con đường Q_4 (để có $c \in C$) rồi qua Q_2 để có b' . b' và b có nhất thiết phải giống nhau hay không? Một CSDL là nhất quán nếu qua các cách khác nhau chỉ xác định được 1 giá trị duy nhất của thuộc tính liên quan. Trong trường hợp này có 3 vấn đề nảy sinh:

- Hai cách xác định (hoặc 2 con đường) mang ý nghĩa hoàn toàn giống nhau.

- Có một con đường phụ thuộc vào con đường kia, nghĩa là tập hợp một số thuộc tính thông qua một con đường là tập con của các thuộc tính đó thông qua con đường kia.

- Hai con đường độc lập nhau thì chu trình ở đây là chu trình giả, và như vậy không có ràng buộc toàn vẹn do sự hiện diện của chu trình.

Ví dụ 4.2.12:

Giả sử các quan hệ Q1, Q2, Q3, và Q4 được định nghĩa trên các thuộc tính như sau:

Q1 (Mã-nhân-viên, Mã-phòng)

Q2 (Mã-phòng, Mã-đề-án)

Q3 (Mã-đề-án, Tên-đề-án)

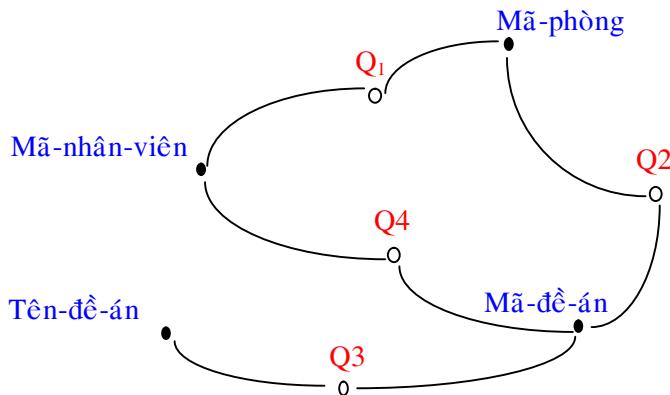
Q4 (Mã-đề-án, Mã-nhân-viên)

Đồ thị biểu diễn CSDL này như trong hình 4.2.2:

Giả thiết 1: Với tân từ sau:

Mỗi nhân viên (thể hiện qua Mã-nhân-viên) được phân công vào tất cả các đề án do phòng đó (thể hiện qua Mã-phòng) phụ trách.

(i) Hai con đường mang ý nghĩa hoàn toàn giống nhau. Đường dài hơn Q1 kết nối với Q2 (ký hiệu là $Q1 \times Q2$) trùng với con đường ngắn hơn Q4 khi cùng xác định một đề án (*thể hiện qua Mã-đề-án*) mà một nhân viên (*thể hiện qua Mã-nhân-viên*) tham gia vào. Đây là một chu trình giả, có thể hủy bỏ (bằng cách loại bỏ quan hệ Q4).



Hình 4.2.2. Lược đồ CSDL được biểu diễn dưới dạng đồ thị

(ii) Nếu vẫn muốn giữ lại quan hệ Q4, tức là không hủy bỏ chương trình, thì phải có một RBTV với thuật toán sau:

$Q1 \bowtie Q2 [Mã-nhân-viên, Mã-đề-án] = Q4 [Mã-nhân-viên, Mã-đề-án]$

Giả thiết 2: Với tân từ sau:

Mỗi nhân viên (*thể hiện qua Mã-nhân-viên*) được phân công vào một số đề án do phòng đó (*thể hiện qua Mã-phòng*) phụ trách.

Con đường ngắn Q4 phụ thuộc vào con đường dài $Q1 \bowtie Q2$, bởi vì một nhân viên có thể không tham gia vào tất cả các đề án do phòng mình phụ trách. Chu trình trên mang ý nghĩa thực sự. Ràng buộc toàn vẹn được thể hiện bởi thuật toán sau:

$Q4 [Mã-nhân-viên, Mã-đề-án] \subseteq Q1 \bowtie Q2 [Mã-nhân-viên, Mã-đề-án]$

Giả thiết 3: Với tân từ sau:

Mỗi nhân viên (*thể hiện qua Mã-nhân-viên*) được phân công vào những đề án (*thể hiện qua Mã-phòng*) bất kỳ.

Hai con đường hoàn toàn độc lập nhau, không liên quan gì tới nhau, mang ý nghĩa hoàn toàn khác nhau, do đó không có RBTV.

4.3. Biểu diễn RBTV bằng phu thuộc hàm.

Trong Chương III, mục 3.1, điểm 3.1.7 đã trình bày khái niệm cơ bản về phụ thuộc hàm (*Functional Dependency*) trong một quan hệ. Phụ thuộc hàm có tầm quan trọng rất lớn trong việc phân tích và thiết kế mô hình dữ liệu. Mục này sẽ trình bày kỹ hơn về vấn đề này.

Khái niệm: Quan hệ R được định nghĩa trên tập thuộc tính $U = \{A_1, A_2, \dots, A_n\}$. $A, B \subset U$ là 2 tập con của tập thuộc tính U . Nếu tồn tại một ánh xạ $f: A \rightarrow B$ thì ta nói rằng A xác định hàm B , hay B phụ thuộc hàm vào A , và ký hiệu là $A \rightarrow B$.

Định nghĩa hình thức của phụ thuộc hàm như sau:

Quan hệ $Q(A, B, C)$ có phụ thuộc hàm A xác định B (ký hiệu là $A \rightarrow B$) nếu:

$\forall q, q' \in Q$, sao cho $q.A = q'.A$ thì $q.B = q'.B$

(Nghĩa là: ứng với 1 giá trị của A thì có một giá trị duy nhất của B)

B là vế phải của phụ thuộc hàm. A là vế trái của phụ thuộc hàm, còn gọi là *Determinant* của B.

$A \rightarrow B$ được gọi là phụ *thuộc hàm hiển nhiên* nếu $B \subseteq A$. Nghĩa là, một tập A lớn hơn và bao tập con B thì A xác định được giá trị của các thuộc tính trong tập B là điều hiển nhiên.

$A \rightarrow B$ được gọi là phụ *thuộc hàm nguyên tố*, hoặc nói cách khác, B được gọi là phụ *thuộc hàm đầy đủ* (*fully functional dependence* hoặc *on all parts of*) vào A nếu $\forall A' \subsetneq A$ chúng ta đều không có phụ thuộc hàm $A' \rightarrow B$.

Ví dụ 4.3.1 :

Trong lược đồ CSDL quản lý hóa đơn bán hàng đã cho trong ví dụ 4.1.2 và 4.2.10, quan hệ HÓAĐƠN (Số-hóa đơn, Số-chủng-loại-mặt-hàng, Tổng-trị-giá) có các phụ thuộc hàm sau:

- f1: Số-hóa đơn \rightarrow Số_chủng_loại_mặt_hàng;
- f2: Số-hóa đơn \rightarrow Tổng-trị-giá;

bởi vì Số-hóa đơn là khóa của lược đồ quan hệ HÓAĐƠN. Nếu biết số hóa đơn thì ta có thể xác định được tất cả các thông tin còn lại liên quan đến hóa đơn đó, trong đó có thông tin về Số_chủng_loại_mặt_hàng và Tổng-trị-giá tất cả các mặt hàng của hóa đơn. Các phụ thuộc hàm trên đều là nguyên tố.

Quan hệ CHITIẾT_HĐ (Số-hóa đơn, Mã-hàng, Số-lượng-đặt, Đơn-giá, Trị-giá) có các phụ thuộc hàm sau:

- f1: Số-hóa đơn, Mã-hàng \rightarrow Số-lượng-đặt.
- f2: Số-hóa đơn, Mã-hàng \rightarrow Đơn-giá.
- f3: Số-hóa đơn, Mã-hàng \rightarrow Trị-giá.
- f4: Số-lượng-đặt, Đơn-giá \rightarrow Trị-giá.

Thuộc tính Đơn-giá phụ thuộc hàm không đầy đủ vào khóa (Số hóa-đơn, Mā-hàng), bởi vì nó chỉ phụ thuộc vào mặt hàng (thông qua Mā-hàng).

Qua ví dụ vừa nêu chúng ta thấy, trên một lược đồ quan hệ có thể tồn tại nhiều phụ thuộc hàm. Tập các phụ thuộc hàm thường được ký hiệu bằng chữ \mathcal{F} .

Gọi \mathcal{F} là tập các phụ thuộc hàm đối với lược đồ quan hệ R định nghĩa trên tập thuộc tính U và $X \rightarrow Y$ là một phụ thuộc hàm; $X, Y \subseteq U$. Ta nói rằng $X \rightarrow Y$ được *suy diễn lôgic* từ \mathcal{F} nếu R thỏa các phụ thuộc hàm của \mathcal{F} thì cũng thỏa $X \rightarrow Y$ và ký hiệu là:

$$F \mid = X \rightarrow Y.$$

Ví dụ 4.3.2:

Với $\mathcal{F} = \{ X \rightarrow Y, X \rightarrow Z, Y \rightarrow T \}$

Thì ta có các phụ thuộc hàm $X \rightarrow YZ$ và $X \rightarrow T$.

Gọi \mathcal{F}^+ là bao đóng (*Closure*) của \mathcal{F} , tức là tập các phụ thuộc hàm được *suy diễn lôgic* từ \mathcal{F} . Nếu $\mathcal{F} = \mathcal{F}^+$ thì ta nói \mathcal{F} là họ đầy đủ (*full family*) của các phụ thuộc hàm.

Bài toán thành viên (*MemberShip*) nêu vấn đề phụ thuộc hàm $X \rightarrow Y$ có phải là được suy diễn lôgic từ \mathcal{F} hay không (tức là $X \rightarrow Y \in \mathcal{F}^+ ?$) là một bài toán khó giải. Nó đòi hỏi chúng ta phải có một hệ luật dẫn để suy diễn lôgic các phụ thuộc hàm.

Năm 1974, Armstrong đã đưa ra hệ tiên đề (còn gọi là hệ luật dẫn Armstrong, hay các tính chất của phụ thuộc hàm) (D.Maier - 1983 [2]) như sau:

$X, Y, Z, W \subseteq U$. Phụ thuộc hàm có các tính chất sau đây:

(i) *Tính phản xạ*:

Nếu $Y \subseteq X$ thì $X \rightarrow Y$.

(ii) *Tính tăng trưởng*:

Nếu $X \rightarrow Y$ và $Y \rightarrow Z$ thì $X \rightarrow YZ$.

(iii) *Tính bắc cầu*:

Nếu $X \rightarrow Y$ và $Y \rightarrow Z$ thì $X \rightarrow Z$.

Và người ta đã chứng minh rằng hệ tiên đề Armstrong là đúng đắn và đầy đủ thông qua 3 bối đề (*ở đây không chứng minh*):

Bối đề 1: *Hệ tiên đề Armstrong là đúng, nghĩa là, với F là tập phụ thuộc hàm đúng trên quan hệ R , nếu $X \rightarrow Y$ là một phụ thuộc hàm được suy diễn logic từ F nhờ hệ tiên đề Armstrong, thì $X \rightarrow Y$ cũng đúng trên quan hệ R .*

Bối đề 2: *Từ hệ tiên đề Armstrong suy ra một số luật bổ sung sau đây:*

(iv) *Tính phân rã (hoặc luật tách)*:

Nếu $X \rightarrow YZ$ thì $X \rightarrow Y$ và $X \rightarrow Z$.

(v) *Tính hợp (hoặc luật hợp)*:

Nếu $X \rightarrow Y$ và $X \rightarrow Z$ thì $X \rightarrow YZ$.

(vi) *Tính tựa bắc cầu, hoặc bắc cầu giả*:

Nếu $X \rightarrow Y$ và $YZ \rightarrow W$ thì $XZ \rightarrow W$.

Định nghĩa: Bao đóng (*Closure*) của tập các thuộc tính X đối với tập các phụ thuộc hàm F (ký hiệu là X_F^+) là tập tất cả các thuộc tính A có thể suy dẫn từ X nhờ tập bao đóng của các phụ thuộc hàm F^+ :

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

Và Bổ đề 3: $X \rightarrow Y$ được suy diễn logic từ F nhờ hệ tiên đề Armstrong nếu và chỉ khi $Y \in X^+_F$.

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

Đây là nền tảng để chứng minh các lý thuyết CSDL quan hệ, đặc biệt là trong “*bài toán thành viên*” - kiểm tra xem một phụ thuộc hàm $f : X \rightarrow Y$ có thuộc bao đóng của tập \mathcal{F} hay không? ...

Ví dụ 4.3.3:

Cho lược đồ quan hệ R (A,B,C,D,E,G,H) và tập các phụ thuộc hàm $\mathcal{F} = \{AB \rightarrow C, B \rightarrow D, CD \rightarrow E, CE \rightarrow GH, G \rightarrow A\}$. Áp dụng hệ tiên đề Armstrong, tìm một chuỗi suy diễn $AB \rightarrow E$.

Giải:

1. $AB \rightarrow C$ (cho trước - phụ thuộc hàm f_1)
2. $AB \rightarrow AB$ (tính chất phản xạ)
3. $AB \rightarrow B$ (luật tách)
4. $B \rightarrow D$ (cho trước - phụ thuộc hàm f_2)
5. $AB \rightarrow D$ (bắc cầu 3 & 4)
6. $AB \rightarrow CD$ (hợp 1 & 5)
7. $CD \rightarrow E$ (cho trước - phụ thuộc hàm f_3)
8. $AB \rightarrow E$ (bắc cầu 6 & 7). Kết thúc.

Ví dụ 4.3.4:

Cho lược đồ quan hệ R (A,B,C,D,E,G,H,I,J) và tập các phụ thuộc hàm $\mathcal{F} = \{AB \rightarrow E, AG \rightarrow J, BE \rightarrow I, E \rightarrow G, GI \rightarrow H\}$. Tìm chuỗi suy diễn $AB \rightarrow GH$ (Bài tập mẫu của David Maier tr. 51)

Giải:

1. $AB \rightarrow E$ (cho trước - phụ thuộc hàm f_1)
2. $AB \rightarrow AB$ (phản xạ)
3. $AB \rightarrow B$ (luật tách)
4. $AB \rightarrow BE$ (hợp của 1 & 3)
5. $BE \rightarrow I$ (cho trước - phụ thuộc hàm f_3)
6. $AB \rightarrow I$ (bắc cầu 4 & 5)
7. $E \rightarrow G$ (cho trước - phụ thuộc hàm f_4)
8. $AB \rightarrow G$ (bắc cầu 1 & 7)
9. $AB \rightarrow GI$ (hợp 6 & 8)
10. $GI \rightarrow H$ (cho trước - phụ thuộc hàm f_5)
11. $AB \rightarrow H$ (bắc cầu 9 & 10)
12. $AB \rightarrow GH$ (hợp 8 & 11)

Thuật toán tìm bao đóng của X dựa trên tập phụ thuộc hàm F đối với quan hệ R được mô tả bằng ngôn ngữ tựa Pascal như sau (Xem D.Maier - 1983 tr.64-69 [4]):

```

Procedure Closure (X, F)
Begin
  OldDep := Ø; NewDep := X;
  While NewDep <> OldDep Do
    Begin
      OldDep := NewDep;
      For every FD:  $W \rightarrow Z \in F$  Do
        If  $W \subseteq NewDep$  Then NewDep := NewDep  $\cup Z$ ;
    End;
    Return NewDep;
  End;

```

Ứng dụng để giải bài toán tìm khóa của quan hệ:

Định nghĩa 3.1.6.2 trong Chương III, mục 3.1 về khóa được viết lại bằng phụ thuộc hàm như sau:

R là lược đồ quan hệ định nghĩa trên tập các thuộc tính $U = \{A_1, A_2, \dots, A_n\}$, với tập các phụ thuộc hàm $F = \{f_1, f_2, \dots, f_m\}$ xác định trên R. K $\subseteq U$ là khóa của R nếu thỏa mãn hai điều kiện sau đây:

- (i) $K \rightarrow U$.
- (ii) $\exists K' \subset K$ mà $K' \rightarrow U$.

Bây giờ chúng ta hãy biểu diễn lược đồ quan hệ R (U) bằng đồ thị có hướng như sau:

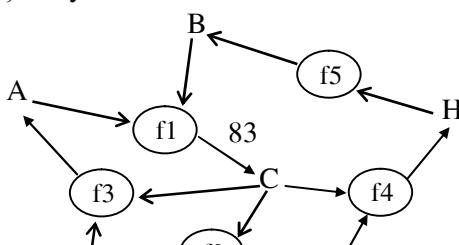
- * Mỗi nút của đồ thị là tên một thuộc tính của R.
- * Cung nối 2 thuộc tính A và B thể hiện phụ thuộc hàm $A \rightarrow B$
- * Thuộc tính mà chỉ có các mũi tên đi ra (tức là chỉ nằm trong vế trái của các phụ thuộc hàm) được gọi là nút gốc.
- * Thuộc tính mà tới nó chỉ có các cung đi tới (tức là chỉ nằm trong vế phải của các phụ thuộc hàm) được gọi là nút lá.

Như vậy khóa của lược đồ quan hệ phải bao phủ tập các nút gốc, đồng thời không chứa bất kỳ nút lá nào của đồ thị.

Xuất phát từ tập các nút gốc (X), dựa trên tập các phụ thuộc hàm F, chúng ta tìm bao đóng X_F^+ . Nếu $X_F^+ = U$ thì X là khoá, ngược lại thì bổ sung một thuộc tính không thuộc nút lá vào X rồi tìm bao đóng. Cứ như thế cho đến khi tìm được bao đóng của X bằng U.

Ví dụ 4.3.5:

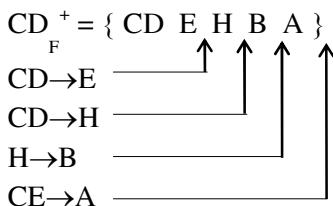
Cho R (A, B, C, D, E, H) với $F = \{AB \rightarrow C, CD \rightarrow E, EC \rightarrow A, CD \rightarrow H, H \rightarrow B\}$. Hãy tìm khóa của R.



Trong đồ thị trên chúng ta thấy chỉ có nút D là nút gốc, các nút còn lại đều không phải nút lá. Khóa của R phải chứa thuộc tính D.

$D_F^+ = \emptyset$, bởi vì không tìm thấy phụ thuộc hàm nào có vế trái chỉ có một mình D.

Vì CD có mặt trong vế trái của 2 phụ thuộc hàm do đó ghép thêm C vào tập các nút gốc để xét khóa.



Như vậy $CD \rightarrow \{C,D,E,H,B,A\}$ do đó CD là khóa của R.

Ví dụ 4.3.6:

Cho quan hệ GIÁNG-DẠY (MS_CBGD, MS_MH, T_CBGD, HH_CBGD, ML, TSSV) với tập phụ thuộc hàm:

$$\begin{aligned} F = \{ & \quad MS_CBGD \rightarrow T_CBGD; \\ & \quad MS_MH \rightarrow HH_CBGD, ML; \\ & \quad HH_CBGD \rightarrow ML; \\ & \quad MS_CBGD \rightarrow HH_CBGD; \end{aligned}$$

$$\left. \begin{array}{l} \text{MS_CBGD, MS_MH} \rightarrow \text{TSSV} \\ \end{array} \right\}$$

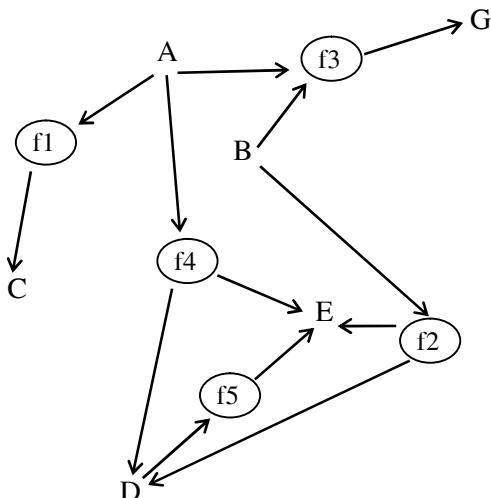
Ở đây MS_CBGD là mã số cán bộ giảng dạy; MS_MH là mã số môn học; T_CBGD là tên cán bộ giảng dạy; HH_CBGD là học hàm của cán bộ giảng dạy; ML là mã số lớp học; và TSSV là tổng số sinh viên theo học môn MS_MH do giảng viên MS_CBGD phụ trách.

Bài toán: Xác định khóa của quan hệ GIÁNG-DẠY.
(TS. Đồng thị Bích Thủy, đề thi năm 1992).

Lời giải: Để cho đơn giản, chúng ta hãy ký hiệu tên các thuộc tính của quan hệ trên lần lượt là A, B, C, D, E, G tương ứng. Khi đó quan hệ GIÁNG-DẠY và tập phụ thuộc hàm F được viết ngắn gọn lại là:

$$\begin{aligned} R &(A, B, C, D, E, G) \\ F = &\{ A \rightarrow C; B \rightarrow D, E; D \rightarrow E; A \rightarrow ED; AB \rightarrow G \} \end{aligned}$$

Đồ thị biểu diễn các phụ thuộc hàm như sau:



Chúng ta nhận thấy trên đồ thị, hai thuộc tính A và B là các nút gốc. E, C và G là các nút lá. Khóa của quan hệ phải chứa các thuộc tính ở các nút gốc.

Xét $X = \{ A \}$

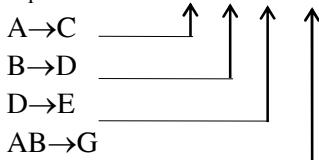
$X_F^+ = \{ A, C, D, E \} //$ Còn thiếu thuộc tính B và G

Xét $X = \{ B \}$

$X_F^+ = \{ D, E \} //$ Còn thiếu thuộc tính A,B,C và G

Lấy $X = \{ A, B \}$

$X_F^+ = \{ AB, C, D, E, G \} = U$



Vậy AB là khóa của R. Tức là, khóa của quan hệ GIẢNG-DẠY là { MS_CBDG, MS_MH }

Phụ thuộc hàm cùng với các loại phụ thuộc đa trị (*Multivalued Dependence*) và phụ thuộc chiếu/kết (*Projection/Join Dependence*) là những khái niệm rất quan trọng trong lý thuyết phân tích và thiết kế CSDL.

Tóm tắt..

Chương IV đã trình bày một số khái niệm cơ bản về:

1. Ràng buộc toàn vẹn (Integrity Constraint): Đó là một điều kiện bất biến không được vi phạm trong một CSDL. Đó là quy tắc (Rule) áp đặt lên trên các đối tượng của thế giới thực.

2. Các yếu tố của ràng buộc toàn vẹn: Khi xác định một RBTV cần chỉ rõ:

a- Điều kiện (tức là nội dung) của RBTV, từ đó xác định cách biểu diễn. Đó là sự mô tả, và biểu diễn hình thức nội dung của nó. Điều kiện có thể được biểu diễn bằng ngôn ngữ tự nhiên, thuật giải (bằng mã giả - *Pseudo Code*, ngôn ngữ tựa Pascal), ngôn ngữ đại số tập hợp, đại số quan hệ v.v hoặc bằng các phụ thuộc hàm.

b- Bối cảnh xảy ra RBTV: trên một hay nhiều quan hệ, cụ thể trên các quan hệ nào.

c- Tâm ảnh hướng của RBTV: Khả năng vi phạm tính toàn vẹn dữ liệu.

Một RBTV có thể liên quan đến một số quan hệ, và chỉ khi có thao tác cập nhật (Thêm, Sửa, Xóa) mới có nguy cơ dẫn đến vi phạm RBTV, do đó cần xác định rõ thao tác nào dẫn đến việc cần phải kiểm tra RBTV.

d- Hành động cần phải có khi phát hiện có RBTV bị vi phạm.

Khi một RBTV bị vi phạm thì cần có những hành động thích hợp. Có 2 giải pháp là đưa ra thông báo và yêu cầu sửa chữa dữ liệu của các thuộc tính cho phù hợp với quy tắc đảm bảo tính nhất quán dữ liệu; hoặc từ chối thao tác cập nhật. Giải pháp thứ hai là phù hợp đối với việc xử lý theo lô (*Batch processing*). Việc từ chối cũng phải được lưu lại bằng những thông báo đầy đủ, rõ ràng vì sao thao tác bị từ chối và cần phải sửa lại những dữ liệu nào.

3. Phân loại ràng buộc toàn vẹn.

Trong một CSDL có thể phát hiện nhiều RBTV, tuy nhiên có thể chia chúng thành hai loại chính theo bối cảnh RBTV:

- a) Ràng buộc toàn vẹn trong bối cảnh là một quan hệ cơ sở.*
- b) Ràng buộc toàn vẹn có bối cảnh trên nhiều quan hệ cơ sở.*

Với loại Ràng buộc toàn vẹn có bối cảnh là 1 quan hệ cơ sở có thể có các loại:

+ Ràng buộc toàn vẹn về miền giá trị của thuộc tính: Mặc dù mỗi thuộc tính đều phải thuộc một kiểu giá trị xác định, tuy nhiên chúng có thể chỉ lấy những giá trị nào đó mà thôi.

+ Ràng buộc toàn vẹn liên thuộc tính: Giá trị của các thuộc tính có thể phụ thuộc lẫn nhau. Chẳng hạn giá trị tính toán trị giá một mặt hàng phụ thuộc vào số lượng và đơn giá của mặt hàng đó.

+ Ràng buộc toàn vẹn liên bộ, liên thuộc tính. Đây là loại RBTY có liên quan tới nhiều bộ và có thể tới nhiều thuộc tính của (các) bộ giá trị trong một quan hệ.

Với loại Ràng buộc toàn vẹn có bối cảnh là nhiều quan hệ cơ sở có thể có các loại:

+ Ràng buộc toàn vẹn về phụ thuộc tồn tại: còn được gọi là *phụ thuộc về khóa ngoại*. Đây là loại RBTY khá phổ biến trong các CSDL bởi các quan hệ trong một CSDL luôn luôn có mối quan hệ mật thiết với nhau. Bộ giá trị của quan hệ này được thêm vào một cách hợp lệ nếu tồn tại một bản ghi tương ứng của một quan hệ khác.

Phụ thuộc tồn tại xảy ra nếu có một trong hai trường hợp sau:

(i) Có sự hiện diện của khóa ngoại.

(ii) Có sự lồng khóa giữa các quan hệ.

+ Ràng buộc toàn vẹn liên bộ - liên quan hệ. Khi có sự hiện diện của 1 thuộc tính mang tính chất tổng hợp (tức là giá trị của thuộc tính có thể được tính toán từ giá trị của các thuộc tính khác trên một hay nhiều bộ giá trị của các quan hệ trong CSDL), hay phụ thuộc tồn tại lồng khóa thì có RBTY liên quan hệ - liên bộ. Chẳng hạn, “Với mỗi lớp LỚP-HỌC, nếu Số lượng học viên lớn hơn 0 thì số lượng này phải lớn hơn hay bằng tổng số bộ giá trị đếm được của các HỌC-VIÊN có cùng Mã lớp”.

+ Ràng buộc toàn vẹn do có sự xuất hiện của chu trình trong đồ thị biểu diễn CSDL.

4. Biểu diễn RBTV bằng các phu thuộc hàm. Trong phần này các khái niệm sau đây đã được đưa ra:

(a) Định nghĩa hình thức của phu thuộc hàm: Quan hệ $Q(A, B, C)$ có phu thuộc hàm A xác định B (ký hiệu là $A \rightarrow B$) nếu:

$$\forall q, q' \in Q, \text{ sao cho } q.A = q'.A \text{ thì } q.B = q'.B$$

(Nghĩa là: Ứng với 1 giá trị của A thì có một giá trị duy nhất của B)

A là vế trái của phu thuộc hàm, B là vế phải của phu thuộc hàm.

(b) Phu thuộc hàm hiển nhiên nếu $B \subseteq A$. Nghĩa là, một tập A lớn hơn và bao tập con B thì A xác định được giá trị của các thuộc tính trong tập B là điều hiển nhiên.

(c) Phu thuộc hàm nguyên tố hoặc phu thuộc hàm đầy đủ: B được gọi là phu thuộc hàm đầy đủ (*fully functional dependence*) vào A nếu $\forall A' \subsetneq A$ đều không có $A' \rightarrow B$.

(d) Đặc biệt là hệ tiên đề Armstrong, hay các tính chất của phu thuộc hàm:

(i) Tính phản xa (Reflexivity):

Nếu $Y \subseteq X$ thì $X \rightarrow Y$.

(ii) Tính tăng trưởng (Augmentation):

Nếu $X \rightarrow Y$ thì $XZ \rightarrow YZ$.

(iii) Tính bắc cầu (Transitivity):

Nếu $X \rightarrow Y$ và $Y \rightarrow Z$ thì $X \rightarrow Z$.

Và một số luật bổ sung rút ra từ các tính chất trên:

(iv) Tính phân rã, hoặc luật tách (Projectivity):

Nếu $X \rightarrow YZ$ thì $X \rightarrow Y$ và $X \rightarrow Z$.

(v) Tính hợp, hoặc luật hợp (Additivity):

Nếu $X \rightarrow Y$ và $X \rightarrow Z$ thì $X \rightarrow YZ$.

(vi) *Tính tựa bắc cầu, hoặc bắc cầu giả (PseudoTransitivity):*

Nếu $X \rightarrow Y$ và $YZ \rightarrow W$ thì $XZ \rightarrow W$.

(e) *Bao đóng của tập các phụ thuộc hàm:* Hệ tiên đề Armstrong là đúng và đầy đủ. Nhờ hệ luật dẫn này người ta đã giải quyết được bài toán thành viên của tập các phụ thuộc hàm: thay vì đi tìm bao đóng (*Closure*) của tập phụ thuộc hàm F (ký hiệu là F^+ , đó là tập các phụ thuộc hàm có thể được suy dẫn lôgic từ F) để kiểm tra xem một phụ thuộc hàm $X \rightarrow Y$ có thuộc F^+ hay không, người ta chỉ cần đi tìm bao đóng của X dựa trên tập các phụ thuộc hàm F (ký hiệu là X_F^+ , đó là tập các thuộc tính có thể suy diễn lôgic từ X nhờ F), rồi kiểm tra xem Y có thuộc X_F^+ hay không.

Trong mục này người ta cũng đã đưa ra thuật toán bằng ngôn ngữ giả để xác định bao đóng của tập thuộc tính dựa trên tập các phụ thuộc hàm.

(f) *Thuật toán xác định khóa của lược đồ quan hệ:* Biểu diễn lược đồ quan hệ bằng đồ thị có hướng. Tìm các nút gốc của đồ thị. Khóa của quan hệ phải chứa các nút gốc này. Bắt đầu từ tập nút gốc, bằng cách bổ sung các nút không phải là lá của đồ thị, người ta tìm được bao đóng của tập này dựa trên tập các phụ thuộc hàm. Nếu bao đóng đó bằng tập khóa của quan hệ thì tập thuộc tính đó là khóa của lược đồ quan hệ.

Nội dung ôn tập: Phụ thuộc hàm, Hệ tiên đề Armstrong, Bao đóng của tập các phụ thuộc hàm, Bao đóng của tập thuộc tính X dựa trên tập các phụ thuộc hàm F , và cách xác định khóa của lược đồ quan hệ.

BÀI TẬP THỰC HÀNH.

Bài tập 1: Cho lược đồ CSDL về hệ thống kế toán của một doanh nghiệp với các quan hệ sau:

1. DM-TK (Mã-TK, Tên-TK)

Tân từ: Danh mục các tài khoản hạch toán kế toán theo chế độ kế toán hiện hành của Nước CHXHCN Việt nam bao gồm các tài khoản. Mỗi tài khoản có một tên gọi cụ thể và một mã số duy nhất để phân biệt với mọi tài khoản khác.

2. TK-ĐỐI-ÚNG (Mã-TK, TK-Đối-ứng);

Tân từ: Mỗi tài khoản, theo chế độ hạch toán hình chữ T, khi được phát sinh bên NỢ (hạch toán tăng) thì phải có một mã tài khoản đối ứng bên CÓ (hạch toán giảm) để đảm bảo cân đối tài khoản. Một tài khoản được ghi NỢ có thể có nhiều tài khoản khác nhau được ghi CÓ. Mã tài khoản NỢ và mã tài khoản đối ứng đều phải thuộc danh mục các tài khoản.

3. SỔ-CT (Loại-CT, Số-CT, NGÀY-CT, Diễn-Giải, Số-Tiền, TK-NỢ, TK-CÓ).

Tân từ: Trong phương pháp kế toán ghi sổ, các chứng từ ban đầu được ghi vào sổ theo dõi, gọi là sổ chứng từ. Mỗi chứng từ đều thuộc một loại chứng từ cụ thể (LOẠI-CT); có một số chứng từ (SỐ-CT) phân biệt với mọi chứng từ khác. Chứng từ được ghi rõ ngày tháng phát sinh (NGÀY-CT), diễn giải nội dung phát sinh (DIỄN-GIẢI), số tiền phát sinh (SỐ-TIỀN), mã tài khoản ghi NỢ (TK-NỢ) và mã tài khoản đối ứng ghi CÓ (TK-CÓ);

4. SỔ-CÁI (Mã-TK, NỢ-ĐK, CÓ-ĐK, PS-NỢ, PS-CÓ, NỢ-CK, CÓ-CK).

Tân từ: Từ sổ chứng từ (SỐ-CT), các chứng từ ghi sổ được tổng hợp theo từng loại tài khoản (Mã-TK) và lập thành sổ cái. Mỗi mã tài khoản (Mã-TK) trong SỔ-CÁI được phản ánh duy nhất 1 lần các số dư NỢ, dư CÓ đầu kỳ (NỢ-ĐK, CÓ-ĐK); số phát sinh NỢ, CÓ trong tháng (PS-NỢ, PS-CÓ), và số dư NỢ, dư CÓ cuối kỳ (NỢ-CK, CÓ-CK). Mã tài khoản phải có trong danh mục tài khoản (DM-TK) nêu trên.

Câu 1: Xác định khóa của các quan hệ trong CSDL nêu trên.

Câu 2: Xác định các RBTV của CSDL. Nêu rõ nội dung RBTV, bối cảnh và lập (các) bảng tầm ảnh hưởng của các RBTV của lược đồ CSDL.

Bài tập 2: Vận dụng hệ tiên đề Armstrong để tìm chuỗi suy diễn:

Cho $R(A,B,C,D,E,G,H)$ với

$$F = \{ AB \rightarrow C; B \rightarrow D; CD \rightarrow E; CE \rightarrow GH; G \rightarrow A \}$$

- (a) Tìm chuỗi suy diễn cho $AB \rightarrow E$.
- (b) Tìm chuỗi suy diễn cho $BG \rightarrow C$.
- (c) Tìm chuỗi suy diễn cho $AB \rightarrow G$.

(D. Maier, *The Theory of Relational DataBases*, pp.69)

Bài tập 3: Xác định khóa của các lược đồ quan hệ sau:

- 1) Q1 (A,B,C,D,E,H)
với $F = \{ AB \rightarrow C; CD \rightarrow E; AH \rightarrow B; B \rightarrow D; A \rightarrow D \}$
- 2) Q2 (A,B,C,D,M,N,P,Q)
với $F = \{ AM \rightarrow NB; BN \rightarrow CM; A \rightarrow P; D \rightarrow M; PC \rightarrow A; DQ \rightarrow A \}$
- 3) Q3 (M, N, P, Q, R, S, T, U, W)
với $F = \{ M \rightarrow W; MR \rightarrow T; T \rightarrow R; QR \rightarrow T; M \rightarrow U; MT \rightarrow P; NP \rightarrow Q; UW \rightarrow R \}$

(Bài tập của TS. Đồng Thị Bích Thủy, giáo trình môn CSDL).