TS. Đặng Thị Thu Hiền

CƠ SỞ DỮ LIỆU

Hà Nội, tháng 3 năm 2013

LỜI MỞ ĐẦU

Công nghệ thông tin là lĩnh vực đang phát triển mạnh mẽ, đặc biệt là những ứng dụng của nó đã trở thành công cụ đắc lực phục vụ cho các hoạt động của con người ở mọi lĩnh vực. Một trong những hỗ trợ phổ biến nhất của máy tính là quản lý thông tin. Mọi thông tin được quản lý trên máy tính đều phải thể hiện bằng các dữ liệu được lưu trữ trong một cơ sở dữ liệu nhất định.

Lý thuyết cơ sở dữ liệu đã được phát triển nhanh chóng theo tốc độ phát triển của công nghệ thông tin. Cơ sở dữ liệu là "linh hồn" của mọi ứng dụng công nghệ thông tin trong cuộc sống. Đánh giá được tầm quan trọng này, đối với những sinh viên ngành công nghệ thông tin thì lý thuyết về cơ sở dữ liệu là môn học không thể thiếu.

Với mục đích xây dựng giáo trình Cơ sở dữ liệu dùng cho các sinh viên đại học, cao đẳng tìm hiểu về lý thuyết cơ sở dữ liệu. Nhằm cung cấp những khái niệm cơ bản về lý thuyết cơ sở dữ liệu. Điều này cũng không ngoài mục đích đổi mới giáo trình – bài giảng nhằm nâng cao chất lượng đào tạo.

Nội dung giáo trình được chia làm 6 chương:

Chương 1: Các khái niệm cơ bản: Giới thiệu các khái niệm cơ bản về cơ sở dữ liệu nói chung, như khái niệm cơ sở dữ liệu, hệ quản trị cơ sở dữ liệu, các mô hình cơ sở dữ liệu.

Chương 2: Mô hình cơ sở dữ liệu quan hệ: Trình bày chi tiết về mô hình cơ sở dữ liệu quan hệ, các khái niệm, các phép toán của đại số tập hợp và đại số quan hệ.

Chương 3: Ngôn ngữ truy vấn dữ liệu: Trình bày về ngôn ngữ truy vấn dữ liệu SQL, các lệnh về truy vấn thông tin, các lệnh cập nhật, phân quyền.

Chương 4: Ràng buộc toàn vẹn, phụ thuộc hàm và khóa: Trình bày tổng quan về ràng buộc toàn vẹn, khái niệm về phụ thuộc hàm và các khái niệm về khóa và thuật toán tìm khóa.

Chương 5: Dạng chuẩn và chuẩn hóa: Trình bày về các dạng chuẩn, và các thuật toán chuẩn hóa lược đồ cơ sở dữ liệu.

Chương 6: Tối ưu hóa câu hỏi: Trình về các quy tắc để tối ưu hóa câu hỏi truy vấn, một số thuật toán và ví dụ minh họa.

Trong quá trình biên soạn giáo trình này tác giả nhận được sự giúp đỡ và tạo điều kiện của toàn thể các giảng viên trong khoa Công nghệ Thông tin – Đại học Giao Thông Vận tải. Tác giả xin gửi lời cảm ơn tới toàn thể các giảng viên về những góp ý quý báu đó. Cuốn giáo trình này cũng không thể được hoàn thành nếu thiếu những lời nhận xét đóng góp của các bạn bè đồng nghiệp ở các Trường, các Viện khác. Tác giả cũng xin ghi nhận và cảm ơn tới những người bạn này.

Trong quá trình biên soạn, tác giả đã cố gắng để hoàn thành tốt cuốn giáo trình. Tuy nhiên, cũng không thể tránh khỏi thiếu xót. Tác giả rất mong nhận được sự góp ý của độc giả để giáo trình ngày càng hoàn thiện hơn.

Hà Nội, tháng 03 năm 2013
Tác giả

CÁC TỪ VIẾT TẮT

CSDL	Database	Cơ sở dữ liệu
DBMS	Database Management System	Hệ quản trị cơ sở dữ liệu
SQL	Structured Query Language	Ngôn ngữ truy vấn dữ liệu có cấu trúc
DDL	Data Definition Language	Ngôn ngữ định nghĩa dữ liệu
DML	Data Manipulation Language	Ngôn ngữ thao tác dữ liệu
DCL	Data Control Language	Ngôn ngữ điều khiển dữ liệu
RBTV		Ràng buộc toàn vẹn
NF	Normal Form	Dạng chuẩn

Chương 1
CÁC KHÁI NIỆM CƠ BẢN

Chương này sẽ trình bày các khái niệm cơ bản của một hệ Cơ sở dữ liệu (CSDL), hệ quản trị CSDL, các mô hình cơ sở dữ liệu. Giúp người đọc có cái nhìn tổng quan về CSDL, hệ quản trị CSDL để tiếp tục tới các vấn đề chi tiết trong chương sau.

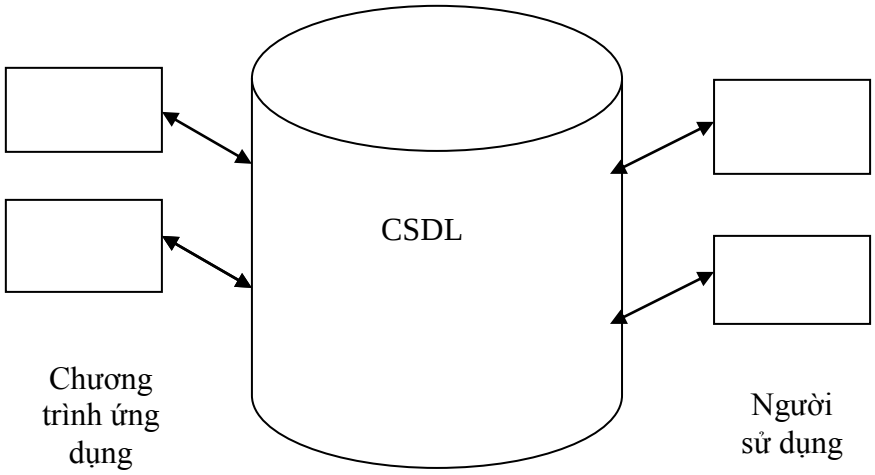
1.1. MỘT SỐ KHÁI NIỆM

1.1.1. Cơ sở dữ liệu

1.1.1.1. Định nghĩa

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.

Hệ cơ sở dữ liệu: Một hệ cơ sở dữ liệu gồm có bốn thành phần đó là: CSDL, người sử dụng hoặc các chương trình ứng dụng, phần mềm, phần cứng.



Hình 1.1. Sơ đồ hệ cơ sở dữ liệu

Qua định nghĩa này ta thấy 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 kết 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…

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.

1.1.1.2. Các đối tượng sử dụng CSDL

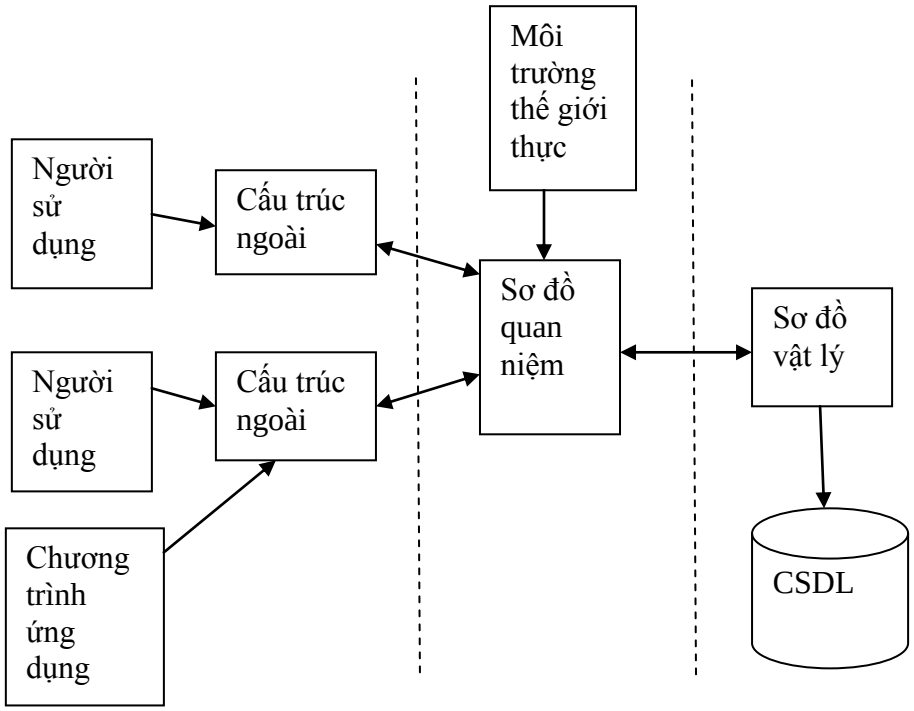
Những người sử dụng CSDL không chuyên về lĩnh vực tin học và CSDL, do đó CSDL cần có các công cụ để cho những người sử dụng không chuyên có thể sử dụng để khai thác CSDL khi cần thiết.

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.

1.1.1.3. Kiến trúc một hệ cơ sở dữ liệu

Theo kiến trúc ANSI-PARC, một hệ CSDL có 3 mức biểu diễn: Mức trong (còn gọi là mức vật lý – Physical), mức quan niệm (Conception hay Logical) và mức ngoài. Giữa các mức tồn tại ánh xạ quan niệm trong, án xạ quan niệm ngoài.



Hình 1.2. Kiến trúc tổng quát của hệ CSDL.

a) 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.

b) 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 ? Mối 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 kết 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 1.1 :

Người ta muốn xây dựng một hệ thống để quản lý các nhân viên của một công ty. Môi trường (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 dễ dẫn 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 dễ 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.

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.

c) 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 sừ 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 1.2

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, sơ lược 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.

1.1.2. Hệ quản trị cơ sở dữ liệu

1.1.2.1. 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 (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 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 II). Dù dựa trên mô hình dữ liệu nào, một hệ quản trị CSDL cũng phải có:

1) 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 kết 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 thên (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.

2) 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....

3) Có biện pháp bảo mật tốt khi có yêu cầu bảo mật.

4) 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.

5) 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.

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

7) 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.

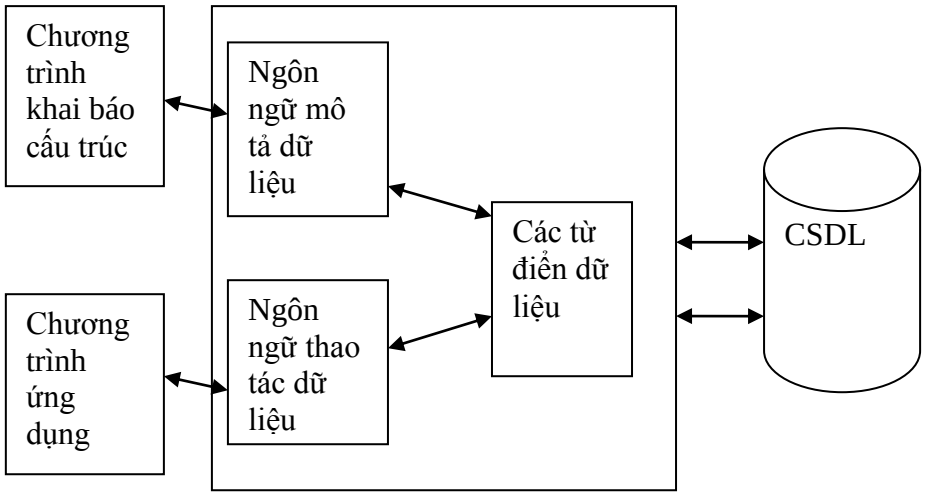
Vài nét về quá trình phát triển các hệ quản trị CSDL:

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.

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



Hình 1.3. Sơ đồ tổng quát của một hệ quản trị CSDL

Hình trên 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 kết 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).

1.1.2.3. Tính độc lập 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 1.2 – Sơ đồ tổng quát của một hệ 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 bớt 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 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".

1.2. CÁC MÔ HÌNH DỮ LIỆU

Mô hình dữ liệu là sự trừu tượng hóa môi trường thực, nó là sự biểu diễn dữ liệu ở mức quan niệm. Mỗi loại mô hình dữ liêu 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 - liên kết và mô hình dữ liệu hướng đối tượng. Phần này sẽ lần lượt giới thiệu các loại mô hình dữ liệu trên.

1.2.1. Mô hình 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: bản ghi (Record), loại bản ghi (Record Type) và loại liên kết (Set Type):

- *Loại bản ghi (Recorde 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 ảnh của thế giới thực có thể là Phòng, Nhân viên, Công việc, lý lịch ... do đó có các loại bản ghi đặc trưng cho từng đối tượng này. Trong đồ thị biểu diễn mô hình mạng mỗi loại bản ghi được biểu diễn bởi một hình chữ nhật

- *Bản ghi* (*Record*): Một thể hiện (Instance) của một loại bản ghi được gọi là bản ghi. Trong ví dụ trên loại bản ghi Phòng có các bản ghi là các phòng, ban trong đơn vị; loại bản ghi nhân viên có các bản ghi là các nhân viên đang làm việc tại các phòng ban của cơ quan...

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

Trong loại liên kết người ta còn chỉ ra số lượng các bản ghi tham gia trong mối liên kết. Có các loại liên kết sau:

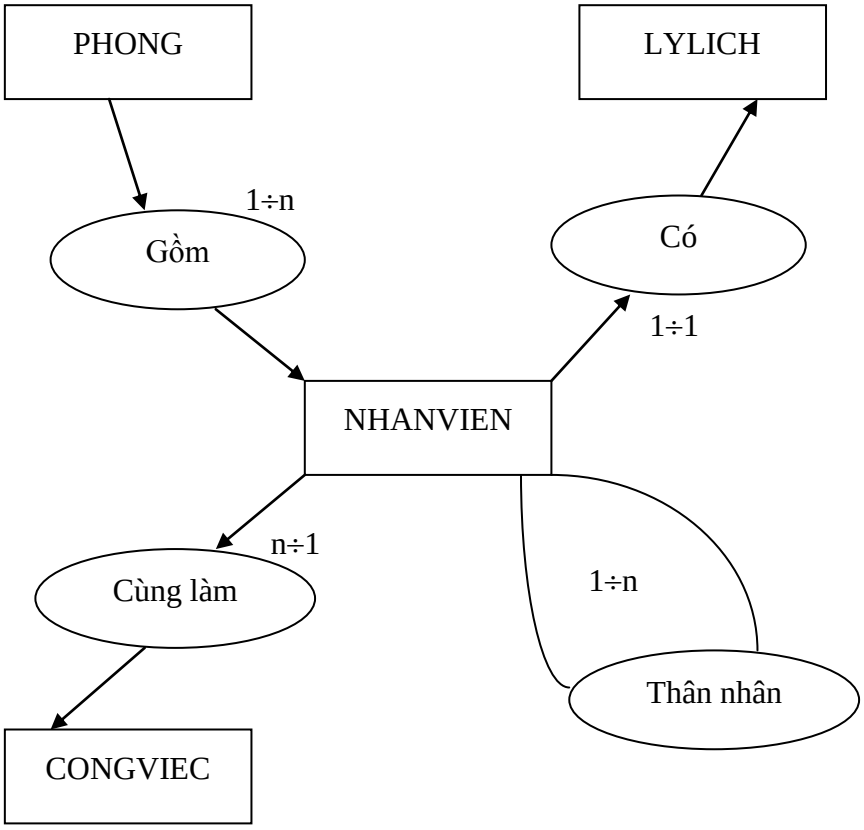
1 – 1 (One-to-One): Mỗi bản ghi của loại bản ghi chủ chủ liên kết với đúng 1 bản ghi của loại bản ghi thành viên. Ví dụ, mỗi nhân viên có duy nhất một lý lịch cá nhân.

1 – n (One-to-Many): Mỗi bản ghi của loại bản ghi chủ chủ liên kết với 1 hay nhiều bản ghi của loại bản ghi 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 -1 (Many-to-One): Nhiều bản ghi của loại bản ghi chủ chủ liên kết với đúng 1 bản ghi của loại bản ghi thành viên. Ví dụ, nhiều nhân viên cùng làm một công việc.

Đệ quy (Recursive): Một loại bản ghi chủ cũng có thể đồng thời là loại bản ghi thành viên với chính nó. Ta nói rằng loại liên kết này là đệ quy.

Hình dưới đây 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 bản ghi: PHONG, NHANVIEN, CONGVIEC và LYLICH, với bốn loại liên kết: mỗi phòng gồm 1 đến nhiều nhân viên; mỗi nhân viên có đúng 1 lý lịch; nhiều nhân viên cùng làm một công việc; một nhân viên có thể có 1 hay nhiều nhân viên là thân nhân của mình (như bố, mẹ, vợ, chồng, anh, chị em ruột cũng là nhân viên của công ty – đây là loại liên kết đệ quy).



Hình 1.4. Mô hình dữ liệu mạng (Network Model)

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 kết phức tạp của dữ liệu trong thực tế là rất hạn chế.

1.2.2. Mô hình 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 kết với nhau theo một mối quan hệ xác định.

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

- *Loại bản ghi*: giống khái niệm bản ghi trong mô hình dữ liệu mạng.
- *Loại mối liên kết:* Kiểu liên kết là phân cấp, theo cách:

Bản ghi thành viên chỉ đóng vai trò thành viên của một mối liên kết duy nhất, tức là nó thuộc một chủ duy nhất. Như vậy, mối liên kết từ bản ghi chủ tới các bản ghi thành viên là 1 – n, và từ bản ghi thành viên với bản ghi chủ là 1 –1.

Giữa 2 loại bản ghi chỉ tồn tại 1 mối liên kết duy nhất.

Ví dụ 1.3:

Giả sử để xây dựng hệ thống quản lý sinh viên của các trường Đại học tại Việt Nam. Chúng ta có thể dùng mô hình phân cấp để phân tích như sau:

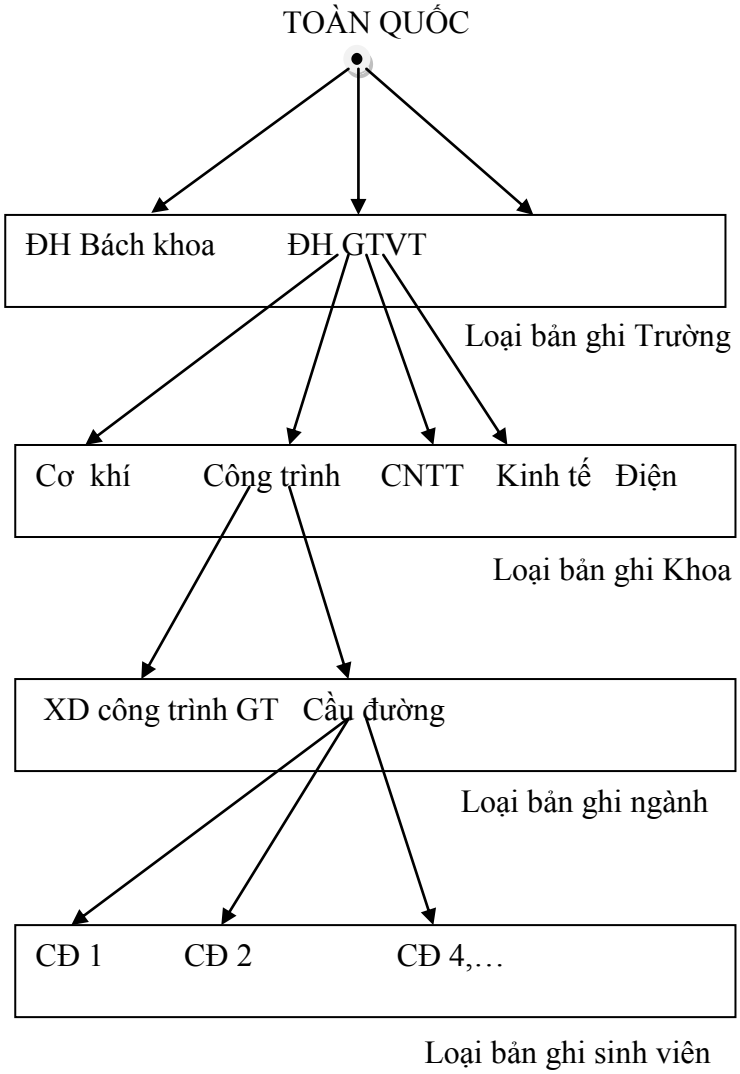
Có 4 loại bản ghi:

Loại bản ghi đặc trưng cho Trường gồm Mã trường, tên trường,…

Loại bản ghi đặc trưng cho Khoa gồm Mã trường, Mã Khoa, tên khoa,..

Loại bản ghi đặc trưng cho ngành gồm Mã trường, Mã khoa, Mã Ngành, tên ngành,..

Loại bản ghi đặc trưng cho sinh viên gồm Mã trường, Mã khoa, Mã Ngành, Mã sinh viên, tên sinh viên,..



Hình 1.5. Mô hình dữ liệu phân cấp (Hierachical Model)

1.2.3. Mô hình 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 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 liên kết 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.

1.2.4. Mô hình dữ liệu thực thể liên kết

Hệ thống ký hiệu:

Sơ đồ này được P.Chen giới thiệu vào năm 1976. Các khái niệm chủ yếu được sử dụng trong sơ đồ này là:

Thực thể (Entity): Là khái niệm mô tả một lớp các đối tượng có đặc trưng chung mà chúng ta cần quan tâm.

Các thực thể là đối tượng cụ thể hoặc trừu tượng:

Ví dụ 1.4:

Ta có thực thể SINHVIEN (Sinh viên), KHACHHANG (Khách hàng), …

Trong sơ đồ thì thực thể thường được ký hiệu là hình chữ nhật

Thực thể yếu: X là thực thể yếu nếu sự tồn tại của X phụ thuộc vào sự tồn tại của thực thể Y. Được ký hiệu bằng hình chữ nhật kép 

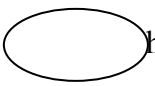
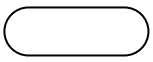
Giả sử để quản lý sinh viên của trường ĐHGT Vận Tải, ta có thực thể LOPHOC (lớp học) thì ta có hai thực thể yếu là BANGHAI (bằng hai) và LIENTHONG (liên thông). Vì phải có thực thể LOPHOC thì mới có hai thực thể BANGHAI, LIENTHONG.

Bản thể: là một đối tượng cụ thể của lớp các đối tượng đó:

Ví dụ 1.5: Sinh viên Đinh Gia Linh là đối tượng cụ thể của thực thể Sinh viên, hay khách hàng Nguyễn Văn An là đối tượng cụ thể của thực thể Khách hàng,…

Thuộc tính(Attribute): Là các tính chất, đặc điểm chung của của lớp đối tượng mà ta quan tâm. Nó là một giá trị dùng để mô tả một đặc trưng nào đó của một thực thể:

Thuộc tính có thể là đơn trị, đa trị (lặp), hoặc phức hợp. (Có thể hiểu thuộc tính giống như một cột trong bảng)

Các thuộc tính thường được ký hiệu là  hoặc 

Ví dụ 1.6: Với thực thể SINHVIEN ta thấy có các thuộc tính như MaSV (Mã sinh viên), HT (họ tên sinh viên), NS (ngày tháng năm sinh) ,…

Khoá (key): là một hoặc một tập các thuộc tính xác định duy nhất một bản thể trong một thực thể. Thuộc tính khoá hay còn gọi là thuộc tính định danh luôn được gạch chân để phân biệt.

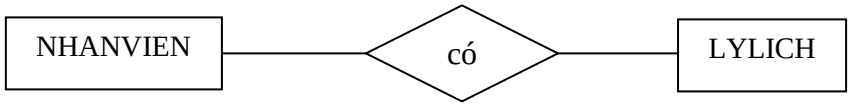
Ví dụ 1.7: Trong thực thể SINHVIEN ta thấy thuộc tính MaSV là khoá vì nó xác định duy nhất một sinh viên, hay nói cách khác không có hai sinh viên nào trùng mã với nhau.

Mối liên hệ (Entity Relationship)

Mối liên hệ giữa các thực thể thường được biểu diễn bằng hình thoi. Trong sơ đồ thực thể liên hệ có các loại quan hệ sau:

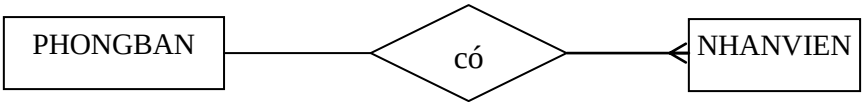
Quan hệ 1-1: là mối quan hệ mà mỗi bản thể trong thực thể này chỉ có nhiều nhất một bản thể được liên kết trong thực thể kia.

Ví dụ 1.8: Mối quan hệ giữa thực thể NHANVIEN và thực thể LYLICH. Mỗi nhân viên chỉ có một lý lịch duy nhất.

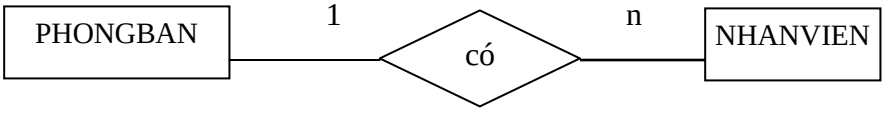


Quan hệ 1-n: là mối quan hệ mà một bản thể trong thực thể này có thể không liên kết hoặc liên kết với một hay nhiều bản thể trong thực thể kia.

Ví dụ 1.9: Mối quan hệ giữa thực thể NHANVIEN và thực thể PHONGBAN. Một phòng ban có thể có một hoặc nhiều nhân viên, mỗi nhân viên chỉ thuộc một phòng ban nhất định.

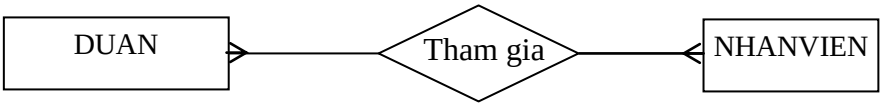


Hoặc có thể ký hiệu

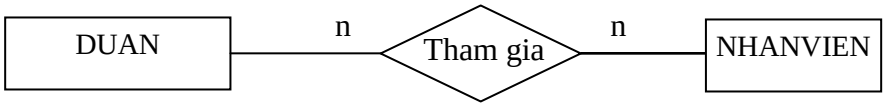


Quan hệ n-n: là mối quan hệ mà một bản thể trong thực thể có thể liên kết với nhiều bản thể trong thực thể khác và ngược lại.

Ví dụ 1.10: Mối quan hệ giữa thực thể NHANVIEN và thực thể DUAN (Dự án). Một nhân viên có thể tham gia nhiều dự án và một dự án có thể cần nhiều nhân viên.

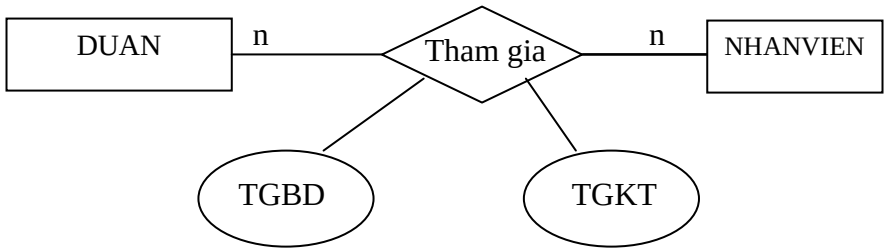


Hoặc có thể ký hiệu



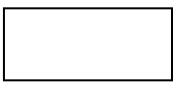
Thuộc tính của mối liên hệ (Relationship Attribute): Mỗi mối liên hệ cũng có thể có những thuộc tính riêng của chúng.

Ví dụ 1.11: Trong mối quan hệ giữa thực thể DUAN và NHANVIEN ta thấy có thuộc tính TGBD (*thời gian bắt đầu)* và TGKT (*thời gian kết thúc)* của mỗi nhân viên khi tham gia vào từng dự dự án, đó chính là hai thuộc tính của mối liên hệ *Tham gia*



Hệ thống lại các ký hiệu:

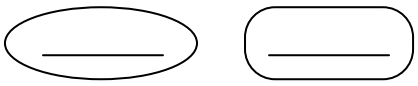
- Thực thể :



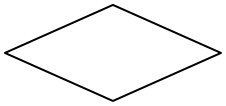
- Thực thể yếu:

- Thuộc tính

- Thuộc tính khoá/định danh



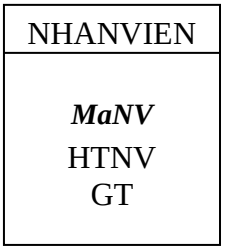
- Mối liên hệ



- Độ kết nối (lực lượng bằng)=1 ————————

- Độ kết nối =N ————————<

Chú ý: Để đơn giản, đôi khi trình bày các thuộc tính nằm luôn trong các thực thể



Cách chuyển mô hình thực thể liên hệ thành lược đồ quan hệ

- Các thực thể tương ứng chuyển thành các bảng/quan hệ

- Các thuộc tính tương ứng chuyển thành các thuộc tính/cột

- Các mối liên hệ ta thực hiện như sau:

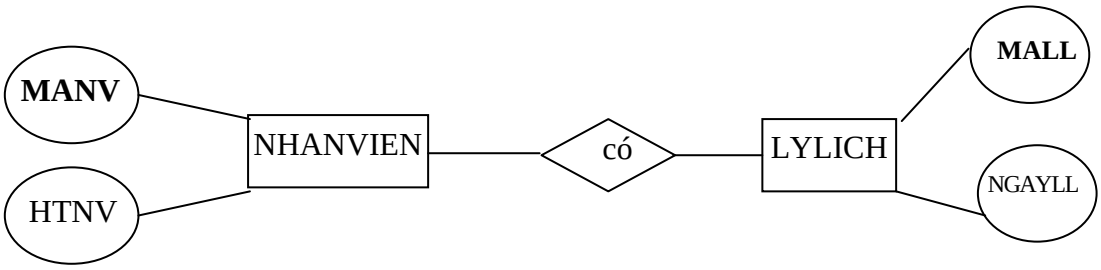
Nếu là mối liên hệ 1-1 thì: khoá của bảng bên 1 bất kỳ trở thành một thuộc tính kết nối (khoá ngoại) ở bảng bên kia.

Nếu là mối liên hệ 1-n thì: khoá của bảng bên 1 trở thành thuộc tính kết nối/khoá ngoại ở bảng bên nhiều.

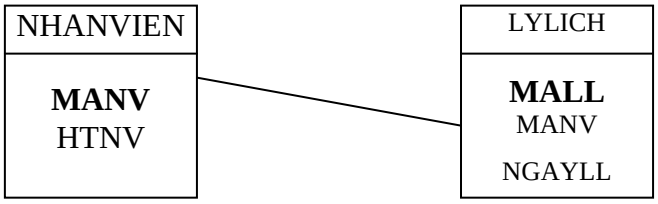
Nếu là mối liên hệ n-n thì: ta phải tạo thêm một bảng trung gian, bảng này có thuộc tính khoá là hai thuộc tính khoá của hai bảng tương ứng. Ngoài ra còn có thể có một số thuộc tính khác như thuộc tính của mối liên hệ,…

Ví dụ 1.12:

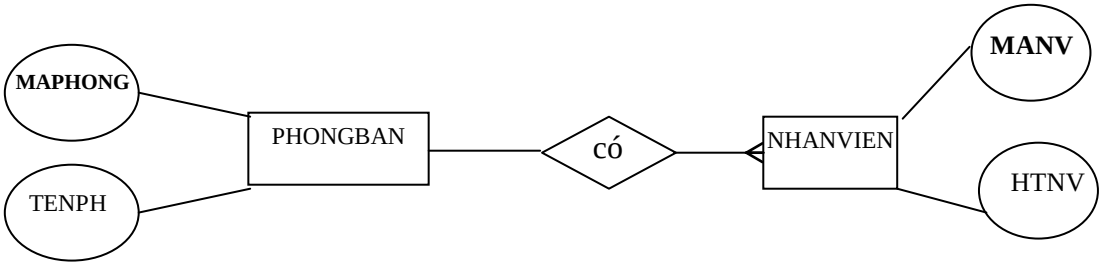
Mối liên hệ 1-1:



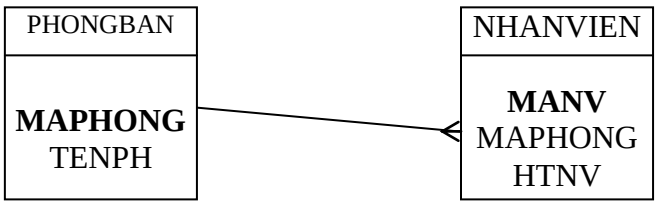
Ta sẽ chuyển thành hai bảng tương ứng như sau:



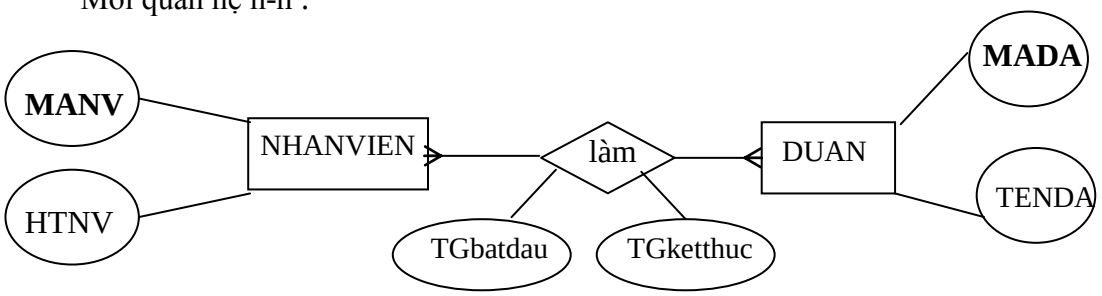
Mối liên hệ 1-n:



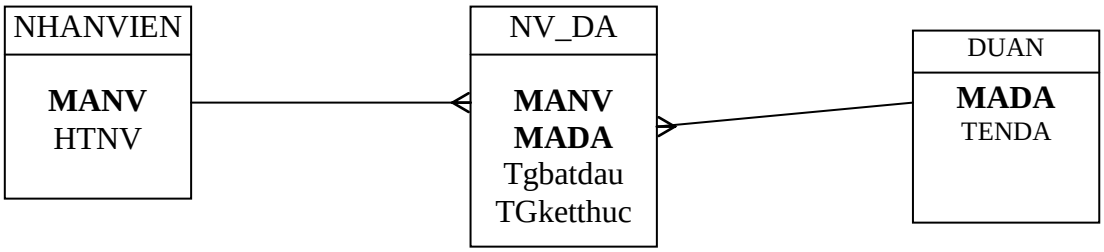
Ta chuyển thành sơ đồ quan hệ như sau:



Mối quan hệ n-n :



Ta chuyển thành các bảng sau :



1.2.5. Mô hình hướng đối tượng

Mô hình này 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. Nó sử dụng các khái niệm như đối tượng, lớp, tính thừa kế, tính thừa kế bội. Đặc trưng cơ bản của mô hình này là tính đóng gói, tính đa hình và tính tái sử dụng.

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 dữ liệu được trừu tượng hoá vì các tác động là để phục vụ hoặc thao tác trên kiểu dữ liệu này. Dữ liệu và phương thức được đồng nhất thành một: Dữ liệu cần có cách thức xử lý thoả đáng và phương thức xử lý được đưa vào trong kiểu dữ liệu đó để phục vụ cho các đối tượng có cấu trúc như thế.

BÀI TẬP CHƯƠNG 1

1.1. Nêu sơ đồ kiến trúc của một hệ cơ sở dữ liệu

1.2. Định nghĩa các thuật ngữ sau :

- Cơ sở dữ liệu

- Hệ cơ sở dữ liệu

- Hệ quản trị cơ sở dữ liệu

1.3. Hiểu và lấy ví dụ của các mô hình cơ sở dữ liệu.

1.4. Dựa vào những khái niệm đã học hãy biểu diễn CSDL có các loại bản ghi PHONG, NHANVIEN, CONGVIEC, LYLICH đã trình bày trong mô hình mạng theo cách tiếp cận phân cấp.

Loại liên kết 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.

1.5. Dựa vào những khái niệm đã học, hãy biểu diễn CSDL về quản lý Sinh viên đã trình bày trong mô hình phân cấp theo cách tiếp cận mạng.

Loại liên kết phân mạng là loại « thuộc về »

1.6. Hệ thống thông tin quản lý kho 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-liên kết » gồm các thực thể và các mối liên kết sau :

CÔNG-VĂN-ĐẾN (ngày phát hành ; Số công văn ; nội dung ; ngày nhận ; số trang ; ghi chú).

CÔNG-VĂN-ĐI (ngày phát hành ; Số công văn ; nội dung ; người ký ; số trang ; Số tờ trình ký ; Ngày trình ký ; ghi chú).

CÔNG-VĂN-ĐẾN và CÔNG-VĂN-ĐI đều là CÔNG-VĂN, là hai loại thực thể yếu/chuyên biệt hóa của loại thực thể CÔNG-VĂN

CHUYÊN-VIÊN (mã Cviên ; Tên Cviên ; Phòng ban ; Ghi chú).

GIẢI QUYẾT (mã Cviên, số-Cviệc, ngày nhận, thời hạn trả lời, KQ giải quyết).

Mỗi công văn đến (từ một đơn vị hay một tác nhân nào đó) có yêu cầu giải quyết thì công văn đó sẽ được chuyển cho một chuyên viên nghiên cứu và đề xuất hướng giải quyết trong một thời hạn nhất định.

Hãy biểu diễn CSDL trên theo cách tiếp cận “thực thể liên kết”.

Chương 2
MÔ HÌNH CƠ SỞ DỮ LIỆU QUAN HỆ

Chương này sẽ giới thiệu các khái niệm cơ bản của cách tiếp cận CSDL theo mô hình quan hệ của E.F.Codd và một số thao tác cơ bản trên các quan hệ.

Phần mở đầu của chương 2 sẽ trình bày kỹ hơn về các khái niệm đã được nhắc tới trong chương I 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.

2.1. CÁC KHÁI NIỆM CƠ BẢN

2.1.1. 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 2.1 :

Đối tượng KHOA (tương ứng với thực thể KHOA trong mô hình thực thể liên kết) có các thuộc tính Makhoa, Tenkhoa.

Đối tượng SINHVIEN có các thuộc tính Masv, Tensv, Que.

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 trừ (-) 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 cả 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.

Mỗi thuộc tính đều phải thuộc một kiểu 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 Charater, số - Number, 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 Logic

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 Dom(A).

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

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).

2.1.2. Quan hệ (Relation)

Một quan hệ R có n ngôi được định nghĩa trên tập các thuộc tính U = A1…An (thứ tự của các thuộc tính là không quan trọng) và kèm theo nó là một vị từ, tức là một quy tắc để xác định mối quan hệ giữa các thuộc tính Ai và được ký hiệu là R(A1…An).

Tập thuộc tính của quan hệ R đôi khi còn được ký hiệu là R^+.

Với Ai là một thuộc tính có miền giá trị là DOM(Ai), như vậy R(A1 … An) là tập con của tích Đề-các: DOM(A1) x ... x DOM(An).

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

Ví dụ 2.2: SINHVIEN (Masv, Tensv, Que) là một quan hệ 3 ngôi, với Masv là mã của sinh viên, Tensv là tên của sinh viên, Que (là quê quán của sinh vien).

Quy tắc: "Mỗi sinh viên có một mã số sinh viên duy nhất để phân biệt với các sinh viên khác trong trường".

Masv	Tensv	Que
Sv1	Nguyễn Văn Anh	Hà Nội
Sv2	Phạm Ngọc Bình	Hải phòng
Sv3	Nguyễn Hoa Cúc	Quảng Ninh
Sv4	Đinh Gia Linh	Hà Nội

Ví dụ 2.3 :

MONHOC (Mamon, Tenmon, Sodvhoctrinh) là quan hệ 3 ngôi.

Quy tắc : « 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 ».

2.1.3. 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 Đề-các miền giá trị của các thuộc tính và thỏa mãn quy tắc đã cho của quan hệ :

q =(a1, a2, ..., an) $\in$ (DOM(A1) x (DOM(A2) x ... x DOM(An) .

Ví dụ 2.4: Trong quan hệ SINHVIEN có các bộ giá trị sau :

q1 = (SV1, Nguyễn Văn Anh, Hà Nội)

q2 = (SV2, Phạm Ngọc Bình, Hải phòng)

q3 = (SV3, Nguyễn Hoa Cúc, Quảng Ninh)

q4 = (SV4, Đinh Gia Linh, Hà Nội)

Để lấy thành phần Ai (tức là giá trị thuộc tính Ai) của bộ giá trị q, ta viết q.Ai. Phép trích rút này được gọi là phép chiếu một bộ lên thuộc tính Ai.

2.1.4. 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 C là tập hợp các lược đồ quan hệ con $\{R_i\}$.

Đôi khi người ta có thể dùng lược đồ quan hệ và quan hệ thay thế cho nhau trong một số trường hợp.

2.1.5. 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_{Ri} gọi là tình trạng của lược đồ cơ sở dữ liệu C.

Ví dụ 2.5 : Ta có các thể hiện của quan hệ SINHVIEN như sau:

Masv	Tensv	Que
Sv1	Nguyễn Văn Anh	Hà Nội
Sv2	Phạm Ngọc Bình	Hải phòng
Sv3	Nguyễn Hoa Cúc	Quảng Ninh
Sv4	Đinh Gia Linh	Hà Nội

2.1.6. Khóa – Siêu khóa – Khóa dự tuyển – Khóa chính – Khóa ngoại

Có nhiều cách khác nhau để định nghĩa khóa:

Định nghĩa 2.1

Khóa của một quan hệ R là một hoặc một số thuộc tính của quan hệ có thể dùng để phân biệt hai bộ bất kỳ trong quan hệ.

Định nghĩa 2.2

Khóa của quan hệ R định nghĩa trên tập các thuộc tính U = A1… An là một tập con $K \subseteq U$ thỏa mãn các tính chất sau: với mọi bộ giá trị q1, q2 của R đều tồn tại một thuộc tính $A \in K$ sao cho $q1.A \neq q2.A$.

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 $q1.K \neq q2.K$. 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 quan hệ R thì K cũng là khóa của R, bởi vì q1.K' $\neq$ q2.K' thì cũng có q1.K $\neq$ q2.K. Như vậy trong quan hệ có thể có rất nhiều khóa.Việc xác định tất cả các khóa của một quan hệ là rất khó khăn.

Theo định nghĩa 2.2 khóa ở đây chưa phải là khóa nhỏ nhất. Chúng ta có thể định nghĩa khóa tốt hơn một cách hình thức như sau:

Định nghĩa 2.3: Khóa tối tiểu

K là khóa tối tiểu của quan hệ R nếu K là khóa của R và mọi K' là tập con thực sự của K đều không là khóa của R.

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.

Chúng ta quy ước rằng từ nay về sau trong giáo trình này, khi nói đến khóa nếu không nói gì thêm có nghĩa là khóa theo định nghĩa 2.3.

Định nghĩa 2.4: Khoá dự tuyển(Candidate): Khóa của quan hệ theo định nghĩa 2.3 được gọi là khóa dự tuyển và là khóa nội của quan hệ.

Trong các phần tiếp theo, nếu không có chú thích gì thêm, thì các khóa dự tuyển đều được gọi chung là các khóa.

Định nghĩa 2.5: Siêu khoá (Supper key): K là siêu khóa của quan hệ R nếu K' $\subseteq$ K là một khóa của quan hệ. Một quan hệ R luôn luôn có ít nhất một siêu khóa và có thể có nhiều siêu khóa.

Ví dụ 2.6

Quan hệ LOPHOC (Malop, Tenlop, Nienkhoa, Sohocvien, Makhoa)

Quan hệ LOPHOC có khóa là Malop và một số siêu khóa sau:

K1 = { Malop, Tenlop}

K2 = { Malop, Tenlop, Sohocvien }

K3 = { Malop, Sohocvien }

K4 = { Malop, Nienkhoa }

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.

Định nghĩa 2.6: Khoá chính (Primary key): Trong trường hợp lược đồ quan hệ Q có nhiều khóa dự tuyển, khi cài đặt trên một hệ quản trị CSDL người sử dụng có thể chọn một trong số các khóa dự tuyển để tạo chỉ mục (Index) chi phối việc truy cập đến các bộ. Khi đó khóa dự tuyển này được gọi là khóa chính. 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 dự tuyển còn lại.

Ví dụ 2.7

KHOA (Makhoa, Tenkhoa)

LOPHOC (Malop, Tenlop, Nienkhoa, Sohocvien, Makhoa)

MONHOC (Mamon, Tenmon, Sodvhoctrinh).

HOCVIEN (Mahocvien, Tenhocvien, Ngaysinh, Quequan, Malop).

GIANGVIEN (Magiangvien, Tengiangvien, Caphocvi, Chuyennganh).

KQUATHI (Mahocvien, Mamon, Lanthi, Ngaythi, Diemthi, Ghichu).

Định nghĩa 2.7: Khoá ngoại (Foreign key): Tập thuộc tính K là khoá ngoại của một quan hệ R nếu K không là khóa chính của quan hệ R nhưng lại là khóa chính của một quan hệ khác.

Trong ví dụ 2.7 ta thấy Makhoa trong quan hệ LOPHOC là khóa ngoại vì nó là khóa chính của quan hệ KHOA.

Malop trong quan hệ HOCVIEN là khóa ngoại của quan hệ HOCVIEN vì nó là khóa chính của quan hệ LOPHOC.

2.1.7. Phụ thuộc hàm (Functional Dependency)

Quan hệ R được định nghĩa trên tập thuộc tính U = A1…An. 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. Chúng ta sẽ tìm hiểu kỹ hơn về phụ thuộc hàm trong chương sau.

Ví dụ 2.8

Trong các quan hệ ở ví dụ 2.7 ta thấy có những phụ thuộc hàm sau:

Quan hệ KHOA, có phụ thuộc hàm *Makhoa* $\rightarrow$ *Tenkhoa*

Quan hệ LOPHOC, có phụ thuộc hàm Malop$\rightarrow${Tenlop, Nienkhoa, Sohocvien}.

Quan hệ MONHOC, có phụ thuộc hàm *Mamon* $\rightarrow${*Tenmon, Sodvhoctrinh*}

Quan hệ HOCVIEN có phụ thuộc hàm Mahocvien$\rightarrow${Tenhocvien, Ngaysinh, Quequan}

2.1.8. 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 2.9:

Quan hệ CCVC (MaCBVC, HotenCBVC, Hesoluong)

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

Thuật toán: "cc ∈ CCVC thì cc.Hesoluong >= 1 & cc.Hesoluong <= 10.

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

2.1.9. Các thao tác cơ bản trên quan hệ

Trong phần này chúng ta chỉ đề cập tới những khái niệm cơ bản còn các phép toán khác trên các quan hệ sẽ được trình bày chi tiết trong chương sau. 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.

2.1.9.1. Phép thêm một bộ mới vào quan hệ

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

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

trong đó, A_{i1}, A_{i2}, ... A_{im} là các thuộc tính, và v_1, v_2, ... v_m là các giá trị thuộc DOM(A_{i1}), DOM(A_{i2}) , ..., DOM(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 2.10 :

Quan hệ: SINHVIEN (Masv, Tensv, Que).

Thêm bộ q5 = (SV5, Đinh Gia Nhi, Hà nội) vào quan hệ SINHVIEN bởi phép thêm như sau:

INSERT (SINHVIEN; [Masv]=SV5, [Tensv]=Đinh Gia Nhi, [Que]=Hà nội)

Thể hiện $T_{SINHVIEN}$ giờ đây là:

Masv	Tensv	Que
Sv1	Nguyễn Văn Anh	Hà Nội
Sv2	Phạm Ngọc Bình	Hải phòng
Sv3	Nguyễn Hoa Cúc	Quảng Ninh
Sv4	Đinh Gia Linh	Hà Nội
Sv5	Đinh Gia Nhi	Hà Nội

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

INSERT (R; v_1, v_2, ... 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.

2.1.9.2. 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, ... A_{im}= v_m).

Trong đó A_{ij}= v_j (j = 1, 2, ..., 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 2.11: Quan hệ SINHVIEN (Masv, Tensv, Que).

Với phép loại bỏ như sau:

DELETE (SINHVIEN; [Que]=Hà nội).

Thì các bộ: có Masv là Sv1, Sv4, Sv5 sẽ bị loại bỏ khỏi quan hệ SINHVIEN

Thể hiện $T_{SINHVIEN}$ lúc này là:

Masv	Tensv	Que
Sv2	Phạm Ngọc Bình	Hải phòng
Sv3	Nguyễn Hoa Cúc	Quảng Ninh

2.1.9.3. 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; $A_{i1}= c_1$, $A_{i2} = c_2$, ... $A_{im}= c_m$; $A_{i1}= v_1$, $A_{i2} = v_2$, ... $A_{im}= v_m$).

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 2.12: Quan hệ SINHVIEN (Masv, Tensv, Que).

Với phép sửa đổi giá trị như sau:

UPDATE (SINHVIEN; [Masv]=SV1, [Que]=Hưng Yên)

thì giá trị của bộ q1 được sửa lại thành:

q1 = (SV1, Nguyễn Văn Anh, Hưng Yên)

2.2. CÁC PHÉP TOÁN TRÊN ĐẠI SỐ TẬP HỢP

Ngôn ngữ đại số quan hệ là ngôn ngữ biểu diễn câu hỏi về các quan hệ. Các phép toán cơ bản trên tập hợp được áp dụng trên tập các bộ giá trị của các quan hệ, đó là: Hợp (*Union*), Hiệu (Trừ - *Minus*), Giao (*Intersection*), Tích Đề-các (*Cartesian*) và phép chia (*Division*).

2.2.1. Phép hợp 2 quan hệ (Union)

Hợp của hai quan hệ R và S cùng xác định trên tập thuộc tính U, được ký hiệu là $R \cup S$, là một quan hệ Q xác định trên tập thuộc tính U, có cùng thứ tự thuộc tính như trong quan hệ R và S, được định nghĩa như sau: $Q = R \cup S = \{t \mid t \in R \text{ hoặc } t \in S\}$.

Nói một cách đơn giản, hợp của 2 quan hệ R và S là một quan hệ có cùng ngôi với quan hệ R và S với các bộ giá trị bằng gộp các bộ giá trị của cả R và S, những bộ giá trị trùng nhau chỉ được giữ lại 1 bộ.

Ví dụ 2.13: Quan hệ SINHVIEN1 gồm:

Masv	Tensv	Que
Sv1	Nguyễn Văn Anh	Hà Nội
Sv2	Phạm Ngọc Bình	Hải phòng
Sv3	Nguyễn Hoa Cúc	Quảng Ninh
Sv4	Đinh Gia Linh	Hà Nội

Quan hệ SINHVIEN2 gồm:

Masv	Tensv	Que
Sv4	Đinh Gia Linh	Hà Nội
Sv5	Phạm Anh Tuấn	Thanh Hóa

Hợp của SINHVIEN1 và SINHVIEN2 là một quan hệ gồm :

Masv	Tensv	Que
Sv1	Nguyễn Văn Anh	Hà Nội
Sv2	Phạm Ngọc Bình	Hải phòng
Sv3	Nguyễn Hoa Cúc	Quảng Ninh
Sv4	Đinh Gia Linh	Hà Nội
Sv5	Phạm Anh Tuấn	Thanh Hóa

Trong hai quan hệ SINHVIEN1 và SINHVIEN2 đều có bộ trùng nhau là Masv =Sv4, Tensv=Đinh Gia Linh, Que=Hà Nội nên ta chỉ giữ lại bộ này một lần trong quan hệ kết quả.

2.2.2. Giao của 2 quan hệ (Intersection)

Giao của hai quan hệ R và S cùng xác định trên tập thuộc tính U, được ký hiệu là $R \cap S$, là một quan hệ Q xác định trên tập thuộc tính U, có cùng thứ tự thuộc tính như trong quan hệ R và S, được định nghĩa như sau: $Q = R \cap S = \{ t \mid t \in R \text{ và } t \in S \}$

Nói một cách đơn giản, giao của 2 quan hệ R và S là một quan hệ có cùng ngôi với quan hệ R và S với các bộ giá trị là các bộ giống nhau của cả hai quan hệ R và S.

Ví dụ 2.14: Giao của hai quan hệ SINHVIEN1 và SINHVIEN2 là một quan hệ như sau:

Masv	Tensv	Que
Sv4	Đinh Gia Linh	Hà Nội

2.2.3. Phép trừ hai quan hệ (Minus)

Hiệu của hai quan hệ R và S cùng xác định trên tập thuộc tính U, được ký hiệu là R – S, là một quan hệ Q xác định trên tập thuộc tính U, có cùng thứ tự thuộc tính như trong quan hệ R và S, được định nghĩa như sau : $Q = R - S = \{ t \mid t \in R \text{ và } t \notin S \}$

Nói một cách đơn giản, hiệu của 2 quan hệ R và S là một quan hệ có cùng ngôi với quan hệ R và S với các bộ giá trị là các bộ giá trị thuộcquan hệ R mà không thuộc quan hệ S..

Ví dụ 2.15: Hiệu của quan hệ SINHVIEN1 và SINHVIEN2 là một quan hệ như sau:

Masv	Tensv	Que
Sv1	Nguyễn Văn Anh	Hà Nội
Sv2	Phạm Ngọc Bình	Hải phòng
Sv3	Nguyễn Hoa Cúc	Quảng Ninh

2.2.4. Tích Đề-các của 2 quan hệ (Cartesian)

R(A1... An) và S(B1...Bm) là hai quan hệ có số bộ giá trị hữu hạn. Tích Đề-các của hai quan hệ R và S, được ký hiệu là R x S, là một quan hệ Q xác định trên tập thuộc tính của R và S (với *n* + *m* thuộc tính) và được định nghĩa như sau:

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

Nói một cách đơn giản, tích Đề-các của 2 quan hệ R và S là một quan hệ Q có số ngôi bằng tổng số ngôi của R và S, với các bộ giá trị gồm 2 phần: phần bên trái là một bộ giá trị của R và phần bên phải là một bộ giá trị của S. Như vậy, nếu R có n_1 bộ giá trị và S có n_2 bộ giá trị, thì Q sẽ có $n_1 \times n_2$ bộ giá trị.

Ví dụ 2.16. Cho các quan hệ R và S như sau:

R

A	B	C
1	2	3
4	5	6
7	8	9

S

D	E	F
5	4	3
1	2	6

RxS

A	B	C	D	E	F
1	2	3	5	4	3
1	2	3	1	2	6
4	5	6	5	4	3
4	5	6	1	2	6
7	8	9	5	4	3
7	8	9	1	2	6

2.2.5. Phép chia hai quan hệ (Division)

R là quan hệ *n* ngôi và S là quan hệ *m* ngôi ($n > m$ và $S \neq \varnothing$), có *m* thuộc tính chung (giống nhau về mặt ngữ nghĩa, hoặc các thuộc tính có thể so sánh được) giữa R và S. Phép chia 2 quan hệ R và S, ký hiệu là $R \div S$, là một quan hệ Q có *n* - *m* ngôi được định nghĩa như sau: $Q = R \div S = \{ t \mid \forall u \in S, (t,u) \in R \}$

Sử dụng định nghĩa phép tích Đề-các, có thể định nghĩa phép chia hình thức hơn như sau: $R \div S = Q$ sao cho $Q \times S \subseteq R$ (với giả thiết thêm là thứ tự thuộc tính của R, S, Q là không quan trọng).

Ví dụ 2.17:

R

A	B	C	D
1	2	5	4
2	3	5	4
1	2	3	6
3	1	4	6
8	4	3	6
2	3	3	6

S

C	D
5	4
3	6

R÷S

A	B
1	2
2	3

Ví dụ 2.18: Cho quan hệ về khả năng lái các loại máy bay của các phi công:

KHANANG (Sohieuphicong, Sohieumaybay)

Sohieuphicong	Sohieumaybay
32	102
30	101
30	103
32	103
33	100
30	102
31	102
30	100
31	100

Câu hỏi: Cho biết các phi công có khả năng lái được cả 3 loại máy bay 100, 101, và 103 ?

Trả lời: Đó là thương của phép chia quan hệ KHANANG cho quan hệ MAYBAY (Sohieumaybay):

100
101
103

Và kết quả là quan hệ PHICONG (Sohieuphicong) có 1 bộ giá trị (30).

2.3. CÁC PHÉP TOÁN TRÊN ĐẠI SỐ QUAN HỆ

Trong phần này, tiếp tục tìm hiểu các phép toán đại số quan hệ phức tạp hơn trên các quan hệ. Đó là phép: chiếu (Projection), chọn (Select), Kết nối (*Join*) - gồm 2 loại: Kết tự nhiên (*Natural Join*) và Theta-Kết (θ-*Join*). Các phép toán khác trên các quan hệ gồm: Kết nối nội (*Inner Join*), Kết trái (*Left Join*) và Kết nối phải (*Right Join*) – hai phép toán này trong một số hệ quản trị CSDL gọi là Kết nối ngoài (*Outer Join*). Các phép toán nêu trên chính là tiền đề cho việc truy vấn CSDL bằng ngôn ngữ SQL sau này.

2.3.1. Phép chiếu (Projection)

Giả sử R(U) là một quan hệ xác định trên tập thuộc tính U = A1...An

$X \subseteq U$. Phép chiếu quan hệ R trên tập con các thuộc tính X là một quan hệ Q xác định trên tập thuộc tính X, ký hiệu là R[X], được định nghĩa như sau:

$Q = R[X]=\{q \mid \exists\ t \in R: q=t.X\}$ hoặc ký hiệu là $\Pi_{Ai1...Aim}(R)$.

Ngữ nghĩa: Trích từ R một số thuộc tính nào đó để tạo thành một quan hệ mới. Số ngôi của quan hệ mới này bằng số thuộc tính của tập con X. Các bộ giá trị của các cột được trích nếu giống nhau sẽ được loại bỏ để chỉ giữ lại một bộ duy nhất (trong thể hiện của quan hệ mới không có 2 bộ nào giống nhau).

Ví dụ 2.19: Giả sử cho quan hệ SINHVIEN1 ở trên.

Câu hỏi yêu cầu: Hãy đưa ra quê của các sinh viên. Khi đó ta chiếu lấy thuộc tính Que của quan hệ SINHVIEN như sau: Π_{QUE}(SINHVIEN1)

Que
Hà Nội
Hải phòng
Quảng Ninh

Ta thấy giá trị trên thuộc tính Que có 3 bản ghi là Hà Nội nên đã được loại bỏ chỉ giữ lại trong quan hệ kết quả 1 bộ giá trị Hà Nội.

2.3.2. Phép chọn (Selection)

Phép chọn cho phép chọn lựa chỉ những bản ghi thỏa mãn một điều kiện Đ nào đó để đưa vào quan hệ kết quả. Điều kiện Đ chính là một biểu thức lôgíc cho kết quả hoặc là đúng (*True*) hoặc là sai (*False*) khi đánh giá trên các bộ giá trị của quan hệ nguồn; nó là tổ hợp của các biểu thức lôgic cơ sở. Mỗi biểu thức cơ sở chứa một phép so sánh: nhỏ hơn (<), nhỏ hơn hay bằng (<=), lớn hơn (>), lớn hơn hay bằng (>=), bằng nhau (=) và khác (hoặc <>) có dạng:

Thuộc tính so sánh với thuộc tính.

Thuộc tính so sánh với hằng (*literal*)

Các biểu thức lôgic cơ sở được tổ hợp với nhau bởi các phép toán lôgic: phép "*và*" lôgic - hay còn gọi là phép nối liền ($\wedge$ - *conjunction*),phép "*hoặc*" lôgic - hay còn gọi là phép nối rời ($\vee$ - *disjunction*) và phép phủ định ($\neg$ - *not*).

Giả sử R (A1... An) là một quan hệ, và Đ là một điều kiện dựa trên tập con thuộc tính R^+. Đánh giá điều kiện Đ trên bộ giá trị t $\in$ R được ký hiệu là E ($t_Đ$) hoặc để đơn giản, ta có thể viết Đ(t).

Phép chọn các bản ghi của R thỏa mãn điều kiện Đ là một quan hệ Q có cùng ngôi với R, ký hiệu là R:Đ, được định nghĩa hình thức như sau: Q ={t$\in$R | Đ(t)=đúng}

Phép chọn cũng có thể được ký hiệu theo C.J.Date bởi dấu sigma (σ): $\sigma_F(R)$.

Ví dụ 2.20: Trên quan hệ SINHVIEN, câu hỏi yêu cầu hãy đưa ra các các sinh viên có Quê ở Hà Nội.

Khi đó ta chọn với điều kiện chọn là Que=Hà Nội như sau $\sigma_{QUE='Hà\ Nội'}$(SINHVIEN1)

Ta được kết quả:

Masv	Tensv	Que
Sv1	Nguyễn Văn Anh	Hà Nội
Sv4	Đinh Gia Linh	Hà Nội

2.3.3. Phép kết nối hai quan hệ (Join)

Giả sử có 2 quan hệ R(U), U= (A1 ...An) và S(V), V= (B1... Bm). t = (a_1, a_2, ..., a_n) là một bộ giá trị của R và u = (b_1, b_2, ..., b_m) là một bộ giá trị của S. Gọi *v* là bộ ghép nối *u* vào *t* (hay bộ giá trị *t* và *u* được "*xếp cạnh nhau*" để tạo thành bộ giá trị mới *v*) được định nghĩa như sau: *v* = (*t*, *u*) = (a_1, a_2, ..., a_n, b_1, b_2, ..., b_m).

A $\in$ U và B $\in$ V là hai thuộc tính có thể so sánh được.

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

Phép kết nối hai quan hệ (có thể nói tắt là *phép kết*) R với S trên các thuộc tính A và B với phép so sánh θ , với giả thiết là giá trị cột R[A] có thể so sánh được (qua phép so sánh θ) với mỗi giá trị của cột R[B], được định nghĩa qua:

$R \overset{A\theta B}{\bowtie} S = \{ v = (t, u) \mid t \in R , u \in S$ và $t.A\ \theta\ u.B\}$

Hoặc: $R \overset{A\theta B}{\bowtie} S = (R \times S) : (A\ \theta\ B)$.

Phép kết nối 2 quan hệ R và S có thể xem như được thực hiện qua 2 bước:

Bước 1: Thực hiện tích Đề-các hai quan hệ R và S.

Bước 2: Chọn các bộ giá trị thỏa mãn điều kiện A θ B.

Ngữ nghĩa: Định nghĩa trên cho ta kết quả của phép kết nối hai quan hệ R và S với phép so sánh θ trên 2 thuộc tính A và B là một quan hệ mới, Đó là kết quả cuối cùng của phép toán quan hệ (phép *Chọn*) trên quan hệ kết quả của phép toán tập hợp (tích *Đề-các*).

Nếu θ là phép toán so sánh bằng nhau (=) thì ta gọi đó là phép ***kết nối bằng (Equi Join)***. Nếu các thuộc tính so sánh là giống tên nhau thì trong kết quả của phép kết nối sẽ loại bỏ đi một trong 2 thuộc tính đó, khi đó phép kết nối được gọi là ***phép kết nối tự nhiên* (*Natural Join*)** và sử dụng ký hiệu cho phép toán là " ***** ".

Ví dụ 2.21: Cho quan hệ R và S

R

A	B	C
1	2	3
4	5	6
7	8	9

S

C	D	E
2	3	1
6	4	3
9	3	5

$R \overset{R.C \geq S.C}{\bowtie} S = Q$

A	B	C	C	D	E
1	2	3	2	3	1
4	5	6	2	3	1
4	5	6	6	4	3
7	8	9	2	3	1
7	8	9	6	4	3
7	8	9	9	3	5

Kết quả ***phép kết nối tự nhiên*** của 2 quan hệ R và S là quan hệ Q' với các bộ giá trị như sau:

R * S = Q'

A	B	C	D	E
4	5	6	4	3
7	8	9	3	5

2.3.4. Các phép toán kết nối khác

Mục này trình bày 3 phép toán kết nối mở rộng khác đặc biệt quan trọng, mà bản chất của chúng vẫn là kết nối. Chúng đã được cài đặt trong một số hệ quản trị CSDL như MicroSoft Access, SQL-Server, Oracle. Các phép kết nối đó là: Kết nối nội (*Inner Join*), Kết nối trái (*Left Join*) và Kết nối phải (*Right Join*).

2.3.4.1. Phép kết nối nội (Inner Join)

Thực chất là phép *kết nối bằng* đã trình bày trên. Tuy nhiên, ngay cả trong trường hợp hai thuộc tính so sánh có cùng tên thì kết quả phép kết nối vẫn giữ lại 2 tên thuộc tính đó.

Ví dụ 2.22:

Cho 2 quan hệ R(ABC) và S(ADEF) với các bộ giá trị như dưới đây. Kết quả của phép *kết nối nội* được cho trong bảng phía bên phải.

R	(A	B	C)	S	(A	D	E	F)	R	$\overset{R.A = S.A}{\bowtie}$	S	= Q	(A	B	C	A	D	E	F)
	1	2	3		3	4	6	7					1	2	3	1	5	3	2
	4	5	6		1	5	3	2					7	8	9	7	2	4	5
	7	8	9		7	2	4	5											

2.3.4.2. Phép kết nối trái (Left Join)

Giả sử có 2 quan hệ R(U), U= (A1... An) và S(V), V= (B1... Bm).

t = (a_1, a_2, ..., a_n) và u = (b_1, b_2, ..., b_m) là hai bộ giá trị của R và S. Gọi *v* là bộ ghép nối *u* vào *t* (hay bộ giá trị *t* và *u* được "*xếp cạnh nhau*") và ký hiệu là:

v = (*t*, *u*) = (a_1, a_2, ..., a_n, b_1, b_2, ..., b_m).

Bộ t_{NULL} = (NULL, NULL, ..., NULL) là một bộ đặc biệt của R gồm *n* giá trị của các thuộc tính A1, A2, ..., An đều là không xác định và u_{NULL} = (NULL, NULL, ..., NULL) là một bộ đặc biệt của S gồm *m* giá trị của các thuộc tính B1, B2, ..., Bm đều là không xác định.

A ∈ U và B ∈ V là hai thuộc tính có thể so sánh được.

Phép *kết nối trái* hai quan hệ R với S trên các thuộc tính A và B với phép so sánh bằng (=), với giả thiết là giá trị cột R[A] có thể so sánh tương đương được với mỗi giá trị của cột S[B], được định nghĩa là:

$R \overset{A=B}{\bowtie} S = \{v=(t,u) \mid (t \in R, u \in S \text{ và } t.A\ \theta\ u.B) \text{ hoặc } (t \in R, u=u_{NULL} \text{ với } t.A \notin S[B])\}$

nghĩa là, tất cả các bộ *v* có được nhờ cách đặt bộ giá trị của R và S xếp cạnh nhau, nếu có giá trị giống nhau trên 2 thuộc tính kết nối, và các bộ *v* có được nhờ cách đặt bộ của R với các bộ NULL của S, nếu không tìm được giá trị tương ứng của thuộc tính kết nối trên quan hệ S.

Ví dụ 2.23: Với hai quan hệ R và S cùng các bộ giá trị của chúng đã được cho trong ví dụ 2.22, kết quả của phép kết nối trái của R và S là:

R	$\overset{R.A=S.A}{\bowtie}$	S	= Q	(A	B	C	A	D	E	F)
				1	2	3	1	5	3	2
				4	5	6	-	-	-	-
				7	8	9	7	2	4	5

Ký hiệu dấu trừ (-) trong các thuộc tính của S được hiểu là giá trị không xác định (*giá trị Null*).

Dòng có giá trị thuộc tính A của R là 4 không tìm được giá trị của thuộc tính A tương ứng trong quan hệ S, nên phần còn lại của nó được để là không xác định. Qua bảng kết quả trình bày trên, chúng ta thấy ý nghĩa của phép toán này là nhằm xác định các bộ giá trị của quan hệ bên trái nhưng không có bộ giá trị tương ứng trong quan hệ phía bên phải.

2.3.4.3. Phép kết nối phải (Right Join)

Vẫn với các quan hệ R, S; các thuộc tính A, B; và các bộ giá trị v, t, u, t_{NULL}, u_{NULL} được xác định như trên.

Phép kết nối phải hai quan hệ R với S trên các thuộc tính A và B với phép so sánh =, với giả thiết là giá trị cột R[A] có thể so sánh tương đương được với mỗi giá trị của cột S[B], được định nghĩa là:

$$R \overset{A=B}{\bowtie\!\!\!\rangle} S = \{v=(t,u) | (t \in R, u \in S \text{ và } t.A\ \theta\ u.B) \text{ hoặc } (t = t_{NULL}, u \in S, \text{ với } t.A \notin S[B])\}$$

nghĩa là, tất cả các bộ *v* có được nhờ cách đặt bộ giá trị của R và S xếp cạnh nhau nếu chúng có giá trị giống nhau trên 2 thuộc tính kết nối, và các bộ NULL của R với các bộ của S, nếu không tìm được giá trị tương ứng của thuộc tính kết nối trên quan hệ R.

Ví dụ 2.24: Giả sử với các quan hệ R và S cùng các bộ giá trị của chúng được xác định như trong ví dụ 2.22 nêu trên. Kết quả của phép kết nối phải R với S là quan hệ với các bộ giá trị sau:

$$R \overset{R.A=S.A}{\bowtie\!\!\!\rangle} S = Q$$

A	B	C	A	D	E	F
1	2	3	1	5	3	2
7	8	9	7	2	4	5
-	-	-	3	4	6	7

Ký hiệu dấu trừ (-) trong các thuộc tính của R được hiểu là giá trị không xác định (*giá trị Null*).

Dòng có giá trị tại thuộc tính A của S là 3 không tìm được giá trị của thuộc tính A tương ứng trong quan hệ R, do đó phần đầu của nó được để là không xác định. Qua bảng kết quả trình bày trên, chúng ta thấy ý nghĩa của phép toán này là nhằm xác định các bộ giá trị của quan hệ bên phải không có bộ giá trị tương ứng trong quan hệ phía bên trái.

BÀI TẬP CHƯƠNG 2

2.1. Cho các quan hệ R1, R2, R3. Thực hiện các phép toán sau:

R1

A	B	C
1	2	3
4	5	6
7	8	9

R2

A	B	C
3	1	4
1	2	3
5	3	1

R3

C	D	E
1	4	2
3	2	1
6	3	4

1. R1 $\cup$ R2
2. R1 $\cap$ R2
3. R1 – R2
4. R2 – R1
5. R1 x R3
6. R1* R3
7. $\sigma_{A>2}$(R1* R3)
8. Π_{BC}(R2)*R3
9. Π_{AC}(R1)*Π_{CD}(R3)
10. $\sigma_{B>2}$(Π_{BC}(R2)*Π_{CD}(R3))

2.2. Cho CSDL quản lý hàng hóa gồm các quan hệ sau:

HANG(MaH, TenH, SLTon); SLTon: là số lượng hàng tồn trong kho

KHACH(MaK, TenK, Diachi); địa chỉ của khách giả sử chỉ là tên tỉnh.

HOADON(SoHD, ngayHD, MaK)

CHITIETHD(SoHD, MaH, SLBan, DGia); DGia là đơn giá bán của sản phẩm.

1. Hãy dùng đại số quan hệ viết biểu thức trả lời các câu hỏi sau:
2. Đưa ra danh sách địa chỉ của các khách hàng.
3. Đưa ra tên hàng và số lượng tồn của những mặt hàng
4. Đưa ra thông tin của các mặt hàng có số lượng tồn >5.
5. Đưa ra các thông tin khách hàng có địa chỉ ở Hà Nội
6. Đưa ra tên khách hàng mua hàng ngày 1/1/2013
7. Đưa ra Mã hàng, Tên hàng có đơn giá bán >200,000
8. Đưa ra tên khách hàng ở Hải Phòng mua hàng ngày 2/2/2013
9. Đưa ra tên hàng được bán trong ngày 2/2/2013
10. Đưa ra các mã hàng chưa từng được bán.
11. Đưa ra các mã khách chưa từng mua hàng từ ngày 12/12/2012.

Chương 3
NGÔN NGỮ TRUY VẤN DỮ LIỆU

3.1. KHÁI QUÁT VỀ NGÔN NGỮ TRUY VẤN DỮ LIỆU

Ta thấy một hệ quản trị CSDL (DBMS) phải có ngôn ngữ giao tiếp giữa người sử dụng với CSDL. Ngôn ngữ giao tiếp CSDL gồm các thành phần:

- Ngôn ngữ mô tả dữ liệu (*Data Definition Languege - DDL*): cho phép khai báo cấu trúc các bảng của CSDL, khai báo các mối quan hệ của dữ liệu và các quy tắc 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, xoá, sửa 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.

Những năm 1975-1976, IBM lần đầu tiên đưa ra hệ quản trị CSDL kiểu quan hệ mang tên SYSTEM-R với ngôn ngữ giao tiếp CSDL SEQUEL (Structured English Query language), đó một ngôn ngữ con để thao tác với CSDL.

Năm 1976 ngôn ngữ SEQUEL được cải tiến thành SEQUEL2. Khoảng năm 1978-1979 SEQUEL2 được cải tiến và đổi tên thành Ngôn ngữ truy vấn có cấu trúc và cuối năm 1979 hệ quản trị CSDL được cải tiến thành SYSTEM-R.

Năm 1986 Viện Tiêu chuẩn quốc gia Mỹ đã công nhận và chuẩn hoá ngôn ngữ SQL và sau đó Tổ chức Tiêu chuẩn Thế giới cũng đã công nhận ngôn ngữ này. Đó là chuẩn SQL-86.

SQL đã qua 3 lần chuẩn hoá lại (1989, 1992, 1996) để mở rộng các phép toán và tăng cường khả năng bảo mật và tính toàn vẹn dữ liệu.

Để trình bày cú pháp các câu lệnh SQL được gọn gàng và dễ hiểu, tài liệu này có đưa ra một số quy ước ký pháp như sau:

- Các biến cú pháp (Syntax Variables), người sử dụng phải điền cụ thể vào khi viết lệnh sẽ được viết bằng chữ thường (lowercase), trong cặp dấu(< >)
- Các thành phần tuỳ chọn được viết trong cặp dấu ngoặc vuông ([ ]).
- Việc lựa chọn một trong các khả năng được thể hiện bằng dấu (|).
- Thành phần bắt buộc phải chọn trong danh sách được viết trong cặp dấu móc đậm nét ({ }).

Lệnh SQL có thể được viết trên nhiều dòng và kết thúc lệnh bởi dấu chấm phẩy (;), tuy nhiên từ khoá, tên, hàm, tên thuộc tính, tên bảng, tên đối tượng thì không được phép viết tách xuống hàng. SQL không phân biệt chữ hoa và chữ thường.

Sau đây chúng ta sử dụng CSDL quản lý bán hàng để minh hoạ cho các câu lệnh.

Khach (Mak, tenk, diachi, dienthoai): gồm Mã khách, tên khách, địa chỉ của khách chỉ ghi tên tỉnh/Thành phố, điện thoại.

Loaihang(Maloai, tenloai): Gồm Mã loại hàng, tên loại hàng

Hang(mah, tenh, slton, maloai): Gồm Mã hàng, tên hàng, Số lượng hàng tồn còn trong kho hàng, mã loại hàng.

HoaDon(SoHD, ngayHD, Mak): Gồm Số hóa đơn bán, ngày hóa đơn bán, mã khách mua hàng.

ChitietHD(SoHD, mah, slb, dgia): Số hóa đơn, mã hàng bán, số lượng bán, đơn giá bán.

Để minh họa cho các câu lệnh này. Chúng ta có thể sử dụng bất kỳ hệ quản trị cơ sở dữ liệu nào. Nhưng trong giáo trình này sử dụng trên hệ quản trị CSDL SQL Server và Access.

3.2. CÂU LỆNH SELECT

Câu lệnh SELECT là một trong số các câu lệnh SQL cài đặt đầy đủ các phép toán quan hệ dựa trên các từ khoá cơ bản SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY. Đây là câu lệnh được sử dụng phổ biến nhất với mục đích tìm kiếm thông tin trong CSDL quan hệ cú pháp tổng quát của câu lệnh như sau:

SELECT [DISTINCT] <biểu thức 1>, <biểu thức 2>,…

[INTO <tên bảng mới>]

FROM <tên bảng 1>, <tên bảng 2>,…

[WHERE] <điều kiện chọn>

[GROUP BY <tên cột 1>,<tên cột 2>,…]

[HAVING <điều kiện in kết quả>]

[ORDER BY <tên cột 1> | <biểu thức số 1> [ASC| DESC],…];

Chúng ta sẽ lần lượt làm rõ từng phần của cú pháp ngôn ngữ.

3.2.1. Mệnh đề SELECT

SELECT [DISTICT] {*| <biểu thức 1> [AS<tên mới 1>], <biểu thức 2> [AS <tên mới 2>],…}

FROM <tên bảng>;

- Cho biết tên các khách hàng của của hàng

```
Select     Tenk
From           KHACH;
```

Kết quả là hiển thị ra một cột tên các khách hàng.

- Nếu không muốn lấy tên các khách hàng trùng nhau thì dùng từ khoá DISTINCT.

```
Select    Distinct   Tenk
From          KHACH;
```

- Muốn hiển thị hết tất cả các cột của bảng dùng ký tự đại diện "*"

```
Select    *
From      KHACH;
```

Kết quả là dữ liệu của tất cả các cột trong bảng KHACH

- Có thể dùng các phép toán số học +,-,*,/, ^,%, các hàm tính toán đối với các cột kiểu số.

```
Select    MaH, Slton*10
From     HANG;
```

Kết quả là số lượng hàng còn trong kho của mỗi mặt hàng được nhân với 10

- Có thể thay đổi tên của các cột trong bảng kết quả ta dùng từ khoá AS

```
Select Mak AS ma_so_khach_hang
From  KHACH;
```

Kết quả là hiển thị ra bảng gồm có một cột có tên là **ma_so_khach_hang**

Nhận xét: Sau từ khoá SELECT ta còn có thể có từ khoá *TOP n*. Điều này cho phép chúng ta chỉ hiển thị *n* hàng trong bảng kết quả. Thông thường khi dùng TOP thì thường kết hợp với sắp xếp ORDER BY.

Ví dụ 3.1: Đưa ra 3 Mã hàng đầu tiên trong danh sách.

```
Select    TOP 3 MaH
From          HANG;
```

3.2.2. Từ khóa WHERE

SELECT…

FROM…

WHERE <điều kiện chọn>

Các bản ghi thoả mãn <điều kiện chọn > mới được thể hiện trong bảng kết quả.

Điều kiện chọn có thể chứa các phép toán And, Or, Between, Not Between, like, =,<>, <, <=, >, >=.

Ta có thể sử dụng các ký tự thay thế: % thay thế cho một chuỗi ký tự

_ thay thế cho một ký tự bắt buộc

‘HaNoi%’ thay thế cho tất cả các chuỗi bắt đầu bằng từ ‘HaNoi’

‘_ _ _’ thay thế cho tất cả các chuỗi có ba ký tự

Chú ý: Trong SQL hằng ký tự được bao bởi cặp dấu nháy đơn. Trong Access dấu * thay thế cho một nhóm ký tự, dấu ? thay thế cho một ký tự, hằng ký tự là cặp dấu nháy kép “”, hằng ngày tháng là cặp dấu # #.

- Hiển thị các mặt hàng có số lượng tồn lớn hơn 100

```
Select  *
From  HANG
Where  Slton>100;
```

- Cho hiển thị các khách hàng ở địa chỉ bắt đầu bằng chữ H

```
Select  *
From  KHACH
Where  diachi  like ‘H%’;
```

3.2.3. Từ khóa FROM

SELECT…

FROM <bảng1, bảng2,…>

Dùng xác định các bảng cần có trong câu lệnh.

- Cho biết các thông tin về khách hàng của các hoá đơn trong tháng 5/2010

```
Select  KHACH.*
From   KHACH, HOADON
Where  (month(NgayHD)=5)  and  (year(NgayHD)=2010)
and  (HOADON.MaK=KHACH.MaK;
```

- Nếu có 2 cột giống nhau trên nhiều bảng, ta cần dùng tên bảng và dấu chấm (.) để phân biệt.

- Ta có thể gán bí danh cho các bảng để cho câu lệnh đơn giản hơn. Sau khi gán ta sử dụng bí danh thay cho tên của bảng.

```
Select  KH.*
From   KHACH KH,HOADON HD
Where  (month(NgayHD)=5)  and  (year(NgayHD)=2010)
and  (HD.MaK=KH.MaK);
```

3.2.4. Từ khóa ORDER BY

SELECT…

FROM…

ORDER BY <tên cột> | <biểu thức> [ASC | DESC], <tên cột> | <biểu thức> [ASC | DESC],…

Biểu thức phải có giá trị số: nó thể hiện số thứ tự của cột trong bảng kết quả được chỉ định sắp xếp thứ tự thay vì phải chỉ rõ tên cột, hơn nữa nếu cột kết quả là cột tính toán thì nó chưa có tên nên các sử dụng biểu thức là một biện pháp thay thế hữu dụng. Có thể sắp xếp theo thứ tự tăng dần (với từ khoá ASC - viết tắt của ASCending - mặc định là ASC) hoặc giảm dần (Descending) theo giá trị cột. Trước hết các bản ghi được xếp theo thứ tự của cột thứ nhất; các bản ghi có cùng giá trị ở cả 2 cột 1 và 2 sẽ được xếp theo cột thứ 3.

- Cho biết các mặt hàng sắp xếp tăng theo số lượng tồn kho

```
Select   MaH, TenH, Slton
From  HANG
Order By  Slton;
```

- Hoặc có thể viết cách khác dùng số thứ tự của cột cần sắp xếp.

```
Select   MaH, TenH, Slton
From  HANG
Order By  3;
```

- Cho biết các khách hàng được sắp xếp theo địa chỉ, nếu cùng địa chỉ thì giảm theo tên.

```
Select  *
From  KHACH
Order By Diachi  Asc, Tenk  Desc;
```

3.2.5. Từ khóa GROUP BY – Phân nhóm dữ liệu

SELECT…

FROM…

GROUP BY <tên cột 1>,<tên cột 2>,…

[HAVING <điều kiện>]

- Từ khóa GROUP BY dùng để nhóm các bản ghi có giá trị giống nhau trên các cột được liệt kê sau từ khóa GROUP BY.

- HAVING theo sau GROUP BY dùng để kiểm tra điều kiện nhóm. Nhóm nào thoả mãn điều kiện sau HAVING thì mới được hiển thị. Lưu ý, dữ liệu được nhóm trước rồi mới kiểm tra điều kiện sau HAVING.

- Đưa ra số lương khách của mỗi địa chỉ

```
Select diachi as Tinh, count(*) as SL_Khach
From KHACH
```

```
Group by diachi;
```

- Cho biết các khách hàng có nhiều hơn 15 lần mua hàng

```
Select Mak, count(Mak) AS So_Lan_mua_hang
From  HOADON
Group By Mak
Having count(Mak)>15;
```

Chú ý: - Nếu trong một câu lệnh vừa có điều kiện Where và Having thì điều kiện sau Where được xử lý trước. Chỉ có những bộ nào thoả mãn điều kiện Where mới được nhóm và sau khi nhóm xong, mỗi nhóm lại kiểm tra điều kiện sau Having.

- Sau từ khóa Having thì có thể có các hàm thống kê, còn sau Where thì không được phép để các hàm thống kê.

- Chỉ có các cột phân nhóm mới được thể hiện trên mệnh đề Select nên chúng ta phải cẩn thận khi ghi những cột trên mệnh đề SELECT.

- Cho biết các khách hàng có nhiều hơn 10 lần mua hàng trong tháng 2 năm 2010

```
Select Mak, count(Mak) as So_Lan_mua_hang
From  HOADON
Where (month(NgayHd)=2) and (year(NgayHd)=2010)
Group By Mak
Having count(Mak)>10;
```

- Tính tổng số lượng hàng của mỗi hoá đơn bán.

```
Select SoHD, sum(Slban) as  So_luong_hang_ban
From chitietHD
Group By Sohd;
```

- Khi phân nhóm dữ liệu ta có thể sử dụng các hàm phân nhóm để tính toán trên mỗi nhóm như count, sum, avg, max, min,…

- Tính số lượng bán trung bình của mỗi hóa đơn.

```
Select SoHD, avg(Slban) as  So_luong_hang_ban
From chitietHD
Group By Sohd;
```

3.3. CÁC HÀM THAO TÁC DỮ LIỆU

3.3.1. Các hàm tính toán trên nhóm các bảng ghi

Qua ví dụ trên, chúng ta đã nhận thấy sự cần thiết của những tính toán trong câu lệnh SELECT. SQL cung cấp một số hàm xây dụng sẵn (Built- in) làm việc trên nhóm theo kỹ thuật tính toán nhanh tiên tiến RushMore.

COUNT (*| <tên cột> - đếm số bản ghi có giá trị xác định tại cột được cho bởi <tên cột>

SUM (<biểu thức>) – tìm tổng giá trị các biểu thức

MIN (<biểu thức>) – tìm giá trị nhỏ nhất

MAX (<biểu thức>) – tìm giá trị lớn nhất

AVG (<biểu thức>) – tính giá trị trung bình của biêu thức dựa trên các bản ghi của các nhóm.

Các hàm này thường phải được đi kèm với từ khóa GROUP BY để thực hiện phân nhóm các bản ghi theo giá trị các cột nào đó trước khi tính toán. Nếu không có từ khóa GROUP BY thì câu lệnh sẽ coi toàn bộ các bản ghi là một nhóm.

- Cho biết số lượng tồn lớn nhất và nhỏ nhất của mỗi mặt hàng

```
Select MaH, TenH, Max(Slton), Min(Slton)
From  HANG;
```

- Đưa ra số lượng lớn nhất của một mặt hàng trong mỗi đơn hàng.

```
Select SoHd, Max(Slban) as [So luong Max]
From ChitietHD
Group By Sohd;
```

Chú ý: các hàm SUM, MAX, MIN, AVG chỉ áp dụng với biểu thức kiểu số

3.3.2. Các hàm tính toán trên bản ghi

Các hàm toán học.

ASB (x)	Trị tuyệt đối của x.
SQRT (x)	Căn bậc hai của x (Access và SQL – Server là:SQR (x))
LOG (x)	Logarit tự nhiên của x.
EXP (x)	Hàm mũ cơ số e của x:
SING (x)	Lấy dấu của số x (-1 : x < 0, 0: x = 0, +1: x > 0)
ROUND (x,n)	Làm tròn tới n số lẻ (Access và SQL – Server, RND (x))

và các hàm lượng giác: SIN, COS, TAN, ASIN, ACOS, ATAN…

Các hàm xử lý chuỗi ký tự.

LEN (str)	Cho chiều dài dãy ký tự
LEFT (str, n)	Lấy n ký tự về phía trái của dãy str

RIGHT (str, n) Lấy n ký tự về phía phải của dãy str

MID (str, p, n) Lấy n ký tự của dãy str kể từ vị trí p trong dãy

Các hàm xử lý ngày tháng và thời gian.

DATE ( ) Cho ngày tháng năm hiện tại (oracle: SYSDATE)

DAY (dd) Cho số thứ tự ngày trong tháng của biểu thức ngày dd

MONTH (dd) Cho số thứ tự tháng trong năm của biểu thức ngày dd

YEAR (dd) Cho năm của biểu thức ngày dd

HOUR (tt) Cho giờ trong ngày (0- 23)

MINUTE (tt) Cho số phút của thời gian tt

SECONDS (tt) Cho số giây của biểu thức giờ tt.

Các hàm chuyển đổi kiểu giá trị.

FORMAT (biểu thức, mẫu): Đổi biểu thức có kiểu bất kỳ thành chuỗi theo mẫu đã cho trong tham số thứ 2. Có thể sử dụng hàm STR để thay thế.

Họ các hàm chuyển đổi biểu thức có kiểu bất kỳ thành một giá trị thuộc kiểu xác định: CSTR, CINT,CLNG, CSIN, CDBL,…

Cú pháp và ngữ nghĩa cụ thể của các hãng cung cấp phần mềm. Tài liệu này không có tham vọng trình bày chi tiết các hàm của ngôn ngữ hệ quản trị CSDL cụ thể.

3.4. TRUY VẤN THÔNG TIN TỪ NHIỀU BẢNG

Việc thực hiện các câu truy vấn trên nhiều bảng, về bản chất là giống như trên một bảng, tức là chỉ cần chỉ ra thông tin gì cần tìm và lấy từ các nguồn dữ liệu nào. Các bảng dữ liệu nguồn này cần chỉ ra trong FROM của câu lệnh SELECT.

Nếu các bảng dữ liệu nguồn có các tên thuộc tính giống nhau thì tên thuộc tính này phải được viết tường minh trong biểu thức tìm kiếm với tên bảng đi kèm phía trước. Nói chung trong một CSDL quan hệ, các bảng thường có các mối quan hệ với nhau. Các bảng được liên kết với nhau qua phép kết nối của từ khóa FROM hoặc thông qua điều kiện của từ khóa WHERE của câu lệnh SELECT. Nếu không thể hiện mối quan hệ này, kết quả sẽ là bảng tích Đề các của bảng 2.

3.4.1. Kết nối tự nhiên

SELECT …

FROM …

WHERE <điều kiện kết nối>…

- Cho biết tên các khách hàng mua hàng trong năm 2011

```
Select KH.Tenk
```

```
From KHACH  KH, HOADON  HD
Where  (KH.Mak=HD.Mak)  and (year(NgayHD)=2011);
```

- Ta có thể sử dụng phép kết nối nội **Inner join** để viết lại câu lệnh trên

```
Select KHACH.Tenk
From  KHACH  Inner  Join  HOADON  on KHACH.Mak=HOADON.Mak
Where  year(NgayHD)=2011;
```

3.4.2. Kết nối ngoại (Outer join)

Kết nối ngoại gồm 2 loại, kết nối trái (Left Outer Join) và kết nối phải (Right Outer Join).

- Cho biết các thông tin về khách hàng và các đơn mua hàng của họ nếu có.

```
Select KHACH.*, HOADON.*
From  KHACH  Left  Outer  Join  HOADON  On KHACH.Mak=HOADON.Mak
```

3.4.3. Truy vấn lồng nhau (Query with SubQuery)

Một truy vấn lồng vào một truy vấn khác gọi là **Subquery**, Subquery cũng bao gồm các từ khóa cơ bản như Query và có thể lồng nhau nhiều mức. *Subquery được bao bởi hai dấu ngoặc và lồng vào truy vấn tại Where hoặc Having*.

Có hai loại truy vấn lồng nhau:

- **Truy vấn lồng nhau phân cấp**: Mức cao hơn chỉ nhận kết quả của mức thấp. Khi thực hiện, các truy vấn cấp thấp hơn sẽ định trị trước một lần rồi cung cấp kết quả cho truy vấn cấp cao hơn.

- **Truy vấn lồng nhau tương quan**: Mỗi một tính toán của truy vấn mức cao hơn có tham chiếu đến các truy vấn mức thấp hơn, mỗi lần tham chiếu như vậy các truy vấn mức thấp hơn phải định trị lại.

- Cho biết đầy đủ thông tin về những mặt hàng có tồn kho lớn nhất.

```
Select *
From HANG
Where Slton=(Select  Max(Slton) From HANG);
```

☞ Truy vấn con thực hiện trước và tìm ra số lượng hàng tồn lớn nhất, sau đó làm điều kiện cho truy vấn ngoài để liệt kê những mặt hàng có số lượng tồn bằng với số lượng tồn lớn nhất.

- Cho biết 5 mặt hàng có tồn kho lớn nhất

```
Select *
From HANG  H
```

```
Where  (Select  count(*)   From HANG
                Where  Slton>H.Slton) < 5;
```

☞ Với mỗi mặt hàng của truy vấn ngoài, truy vấn con bên trong sẽ đếm các mặt hàng có số lượng tồn lớn hơn mặt hàng đó, nếu có ít hơn 5 mặt hàng có số lượng tồn lớn hơn chúng thì có nghĩa là nó nằm trong 5 mặt hàng lớn nhất.

- Tương tự cho biết 5 mặt hàng có tồn kho bé nhất

```
Select *
From HANG  H
Where  (Select  count(*)   From HANG
           Where  Slton < H.Slton) < 5;
```

Các phép toán có thể dùng đối với truy vấn lồng nhau

Phép toán tập hợp In, Not in: Để xem một bản ghi có thuộc một bảng hay không ta dùng Subquery với toán tử **In** hoặc **Not In**.

- Cho biết các khách hàng ở Ha Noi mua hàng trong tháng 1/2011

```
Select *
From KHACH
Where (Diachi like 'Ha Noi') and
      (Mak in (Select Mak
               From HOADON
   Where month(NgayHD)=1 and year(NgayHD)=2011));
```

- Cho biết các mặt hàng chưa từng được bán

```
Select *
From HANG
Where Mah Not in (Select Mah From ChitietHD);
```

Phép so sánh tập hợp:

<some, <=some, >some, >=some, =some, <>some Tương đương với:

<any, <=any, >any, >=any, =any, <>any

<all, <=all, >all, >=all, =all, <>all

Chú ý: **=some** tương đương với **In** nhưng **<>some** không tương đương với **Not In**, **<>all** tương đương với **Not In**

- Liệt kê các mặt hàng không phải là mặt hàng có tồn kho lớn nhất

```
Select *
From HANG
```

```
Where Slton < some (Select Slton From HANG);
```

- Cho biết số lượng trung bình một lần đặt hàng của một mặt hàng.

```
Select Mah, Avg(Slban)
From ChitietHD
Group By Mah;
```

- Bây giờ ta muốn biết mặt hàng có số lượng đặt hàng trung bình lớn nhất. Dễ dàng có thể nghĩ đến cách dùng Max(Avg(SOLUONGBAN)) nhưng trong SQL không cho phép các hàm thống kê lồng nhau. Cách giải quyết là:

```
Select Mah,Avg(Slban)
From ChitietHD
Group By Mah
Having Avg(Slban)>=All (Select Avg(Slban)
                        From ChitietHD
                        Group By Mah);
```

Phép toán kiểm tra bảng rỗng

Exists(Q) = True nếu có ít nhất một bản ghi trong Q

= False nếu ngược lại

Not Exists(Q) = True Q không có bộ nào

= False nếu ngược lại

- Cho biết thông tin về các mặt hàng được bán trong tháng 7/2012

```
Select H.*
From HANG H
Where Exists (Select *
              From HOADON D, ChitietHD C
              Where (year(NgayHD)=2012) and
(month(NgayHD)=7) and (D.SoHD=C.SoHD) and (C.Mah=H.Mah));
```

Kiểm tra các bản ghi trùng nhau

Unique(Q) = True nếu Q không có các bộ trùng nhau

= False nếu ngược lại

Not Unique(Q) = True nếu Q có các bộ trùng nhau

= False nếu ngược lại

- Tìm các khách hàng chỉ mua hàng một lần

```
Select *
From KHACH  K
```

```
Where Unique (Select Mak
                From HOADON H Where K.Mak=H.Mak);
```

- Tìm các khách hàng có ít nhất hai lần mua hàng

```
Select *
From KHACH  K
Where Not Unique (Select Mak
                From HOADON H Where K.Mak=H.Mak);
```

3.5. CÁC LỆNH CẬP NHẬT DỮ LIỆU

3.5.1. Bổ sung giá trị mới

Có thể thêm vào bảng mỗi lần một bản ghi hoặc nhiều bàn ghi lấy kết quả từ một truy vấn nào đó.

Bổ sung trực tiếp một bộ giá trị

INSERT INTO <tên bảng> [(tên cột 1>, <tên cột 2>,…)]

VALUES (<biểu thức 1>, <biểu thức 2>,…);

☞ Thêm một bản ghi mới vào bảng có tên được chỉ ra sau từ khoá INTO với giá trị của <biểu thức 1> được gán cho <tên cột 1>, <biểu thức 2> được gán cho <tên cột 2>, …

Lưu ý: Số lượng biểu thức và kiểu giá trị của các biểu thức phải tương ứng với số lượng và kiểu giá trị của các tên cột trong danh sách tên cột của bảng. Ngoài ra, các giá trị còn phải phù hợp với các ràng buộc toàn vẹn định nghĩa trên quan hệ, trong đó có ràng buộc toàn vẹn về khoá chính (Primary key), khoá ngoại (Foreign key) và miền giá trị. Tên thuộc các tính khoá chính và khoá ngoại phải có mặt trong danh sách tên cột của lệnh. Nếu các giá trị của các biểu thức sau từ khoá VALUES vi phạm ràng buộc toàn vẹn thì hệ quản trị CSDL sẽ thông báo lỗi và bộ giá trị mới sẽ không được bổ sung vào bảng.

Ví dụ 3.2: Thêm một khách hàng mới có nội dung

Mak=K2000, Tenkh=Dinh Gia Linh, Diachi=Hanoi, Dienthoai=048570581 vào bảng KHACH

Insert Into KHACH

Values (’K2000’ ,’Dinh Gia Linh’, ’Hanoi’, ’0438570581’);

Thêm một hay nhiều bộ giá trị từ truy vấn.

INSERT INTO <tên bảng> [(<tên cột 1>, <tên cột 2>,…)]

SELECT <biểu thức 1>, <biểu thức 2>,…

FROM <danh sách các bảng nguồn>

[WHERE <điều kiện>]…

☞ Cũng như trên, số lượng biểu thức và kiểu giá trị của các biểu thức sau SELECT phải phù hợp với số lượng và kiểu của các cột có tên trong danh sách đi sau tên bảng, đồng thời phải phù hợp với các ràng buộc toàn vẹn được định nghĩa trên bảng đó.

Nếu giá trị của các biểu thức sau từ khoá SELECT hoàn toàn phù hợp về số lượng, miền giá trị và thứ tự của các cột trong bảng thì danh sách tên các cột của bảng sau khi từ khoá INTO có thể được bỏ qua.

3.5.2. Tạo mới một bảng với các bộ giá trị lấy từ CSDL

Các câu truy vấn dữ liệu để tìm kiếm thông tin tạo ra một bảng trung gian với những mối quan hệ sao cho có thể xem và nếu được phép có thể sửa chữa dữ liệu hoặc xoá bỏ chúng. Các QUERY đó đã tạo ra những khung nhìn (VIEW). Trong nhiều trường hợp chúng ta muốn các bảng này có thể được thực hiện nhờ một lệnh truy vấn hành động (Action Query) gọi là truy vấn tạo bảng mới (Make Table Query)

SELECT <biểu thức 1>, <biểu thức 2>,…

FROM <danh sách các bảng nguồn>

INTO TABLE <tên bảng>

[WHERE <điều kiện>]

GROUP BY <danh sách cột phân nhóm>]

[HAVING <điều kiện>]

[ORDER BY <cột 1>[ASC | DESC], <cột 2> [ASC | DESC],…]:

Ví dụ 3.3: Tạo bảng mới có tên là KHHANOI gồm các khách hàng ở Hanoi

```
Select Mak, Tenk, diachi, dienthoai
From KHACH
Into Table KHHANOI
Where Diachi like 'Hanoi';
```

3.5.3. Sửa nội dung của bộ

UPDATE <tên bảng>

SET <tên cột 1> = <biểu thức 1>,

<tên cột 2> = <biểu thức 2,…

<tên cột n> = <biểu thức n>

[WHERE <điều kiện>];

☞ Giá trị của các cột có tên trong danh sách <tên cột 1>, <tên cột 2>,… của những bản ghi thoả mãn điều kiện sau WHERE sẽ được sửa đổi thành giá trị của các <biểu thức 1>, <biểu thức 2>,… tương ứng. Nếu không có điều kiện WHERE, thì tất cả các bản ghi của bảng sẽ được sửa đổi.

Ví dụ 3.4: Sửa số lượng hàng tồn kho của tất cả các mặt hàng còn lại một nửa.

```
Update HANG
Set Slton=Slton/2;
```

3.5.4. Xoá bộ

DELETE FROM <tên bảng có bộ cần xóa>
[FROM <tên các bảng>]
[WHERE <điều kiện>];

☞ Các bản ghi thoả mãn điều kiện sau WHERE sẽ bị xoá khỏi bảng, nếu không có WHERE thì tất cả các bản ghi của bảng sẽ bị xoá khỏi bảng.

Ví dụ 3.5: Xoá các khách hàng tại HaiPhong

```
Delete from KHACH
Where diachi like 'HaiPhong';
```

3.6. CÁC LỆNH LIÊN QUAN ĐẾN CẤU TRÚC

3.6.1. Cách đặt tên đối tượng và các kiểu dữ liệu.

Cách đặt tên

SQL chuẩn hoá (86, 89, 92, 96) đều quy định cách đặt tên các đối tượng như tên bảng, tên cột của bảng tên View, tên ràng buộc toàn vẹn… (gọi chung là định danh – Identifier) như sau:

Tên gọi gồm tối đa 32 ký tự chữ cái Latinh, chữ số Arập và dấu gạch chân (Underscore) và phải bắt đầu bằng một chữ cái Latinh hoặc dấu gạch chân. Tuyệt đối không chứa khoảng trắng hay ký tự chữ cái không phải là Latinh như tiếng Việt chẳng hạn. Chữ in hoa hay chữ thường đều được xem là như nhau. Tên bảng phải là duy nhất trong CSDL và tên bảng trung gian, và không trùng với bất cứ từ khoá nào trong ngôn ngữ quản trị CSDL.

Tên cột của một bảng là khác nhau, nhưng chúng có thể giống nhau nếu chúng nằm trong các bảng khác nhau.

Tuy nhiên phiên bản mới của một số hệ quản trị CSDL cho phép đặt tên có dấu cách, nhưng khi thao tác phải bao bởi cặp dấu ngoặc vuông []

Các kiểu dữ liệu

Char (w)	Kiểu ký tự với kích thước cố định. Chiều dài của giá trị dữ liệu luôn luôn là w ký tự. Kích thước tối thiểu là 1 và tối đa là 255 ký tự.
Varchar (w)	Kiểu ký tự với kích thước thay đổi từ 0 đến 2000 ký tự.
Int/integer	Số nguyên
Smallint/byte	Số nguyên nhỏ
Numberic(w,s)	Số thực gồm w chữ số kể cả dấu chấm và s chữ số thập phân.
Real,Double	Số thực dấu phảy động
Float(n)	Số thực dấu phảy động ít nhất n chữ số

Date	Kiểu dữ liệu ngày tháng năm
Time	Kiểu giờ, phút giây
Logical	Kiểu dữ liệu lôgic 1 byte có giá trị hoặc đúng (True), hoặc sai (False).

3.6.2. Tạo bảng CSDL

CREATE TABLE <tên bảng>
(<tên cột 1> <kiểu dữ liệu 1> (<kích thước 1>),
<tên cột 2> <kiểu dữ liệu 2> (<kích thước 2>), …
<tên cột n> <kiểu dữ liệu n> (<kích thước n>)
[<Ràng buộc toàn vẹn 1>,
…,
<Ràng buộc toàn vẹn 2>]);

Các ràng buộc toàn vẹn bao gồm :

Khoá chính : primary Key
Khoá thành viên : Unique
Khoá ngoại : Foreign Key (A,B,…,C) References
Biểu thức logic kiểm tra P : Check(P)

Ví dụ 3.6: Tạo bảng HANG

```
Create table HANG(
Mah char(5) not Null,
Tenh char(30),
Slton  integer,
Primary Key (Mah),
Check (Slton>=0) ;
```

☞ Từ khoá **Not Null** không cho phép giá trị của cột rỗng

Ví dụ 3.7: Tạo bảng ChitietHD

```
Create table ChitietHD(
Sohd char(5) not null,
Mah char(5) not null,
Slban real,
primary key (Sohd, Mah),
foreign Key (Mah) References HOADON,
check (Slban>=0)) ;
```

3.6.3. Xoá một bảng

DROP TABLE <tên bảng>

Ví dụ 3.8: xoá bảng khách hàng

```
Drop table KHACH;
```

3.6.4. Sửa đổi cấu trúc của bảng

Thêm một cột

ALTER TABLE <tên bảng>

ADD <tên cột><kiểu dữ liệu>;

Xoá một cột

ALTER TABLE <tên bảng>

DROP COLUMN <tên cột>;

Thay đổi kiểu dữ liệu của cột

ALTER TABLE <tên bảng>

ALTER COLUMN <tên cột> <kiểu dữ liệu mới>;

Ví dụ 3.9: Thêm cột Giới tính vào bảng KHACH

```
Alter table KHACH
Add GT char(3);
```

Thay đổi độ rộng của cột địa chỉ trong bảng khachhang

```
Alter table KHACH
  Alter column Diachi char(40);
```

Xoá bỏ cột GT trong bảng KHACH

```
Alter table KHACH
   Drop column GT;
```

3.7. CÁC LỆNH GIAO QUYỀN TRUY NHẬP CSDL

GRANT là hình thức giao quyền truy nhập những bảng cho người sử dụng, và khi cần thiết có thể thu hồi lại những quyền ấy. Trong đó các quyền, các bảng và người sử dụng là những đối tác cụ thể của GRANT.

Lệnh giao quyền

GRANT <các quyền> ON <tên các bảng>

TO <tên các user> [WITH GRANT OPTION];

Lựa chọn [WITH GRANT OPTION] cho phép các user này được phép giao quyền tiếp.

Lệnh thu hồi quyền

REVOKE <các quyền> ON

<tên các bảng> FROM <tên các user>;

Các quyền có thể được giao và thu hồi là:

- SELECT: gọi, đọc dữ liệu, tạo truy vấn từ bảng
- UPDATE: Thay đổi dữ liệu của bảng
- DELETE: xoá các bản ghi
- INSERT: Thêm các bản ghi
- INDEX: Tạo chỉ mục từ bảng
- ALTER: Hiệu chỉnh cấu trúc của bảng

Ví dụ 3.10: giao quyền SELECT, INSERT, DELETE cho GiaLinh với các bảng KHACH, HANG

```
GRANT SELECT, INSERT, DELETE
ON KHACH, HANG
TO GiaLinh WITH GRANT OPTION;
```

Thu hồi lại quyền DELETE của GiaLinh đối với bảng HANG

```
REVOKE DELETE ON HANG FROM GiaLinh;
```

BÀI TẬP CHƯƠNG 3

3.1: Cho cơ sở dữ liệu dùng để quản lý các chuyến đi của một công ty du lịch

1. DIADIEM(MADD, TENDD)

Mỗi một địa điểm có một mã số(MADD) dùng để phân biệt với các địa điểm khác và có một tên (TENDD)

2. XE(BIENSO, KHTD)

Mỗi một xe có một biển số duy nhất(BIENSO) để phân biệt với các xe khác và có số lượng khách tối đa mà xe đó có thể chở(KHTD)

3. HUONGDV(MAHDV, HTHDV, DCHDV)

Mỗi một hướng dẫn viên của công ty có một mã số duy nhất để phân biệt(MAHDV), có họ tên(HTHDV) và địa chỉ của hướng dẫn viên(DCHDV)

4. CHUYENDI(MACD, TENCD, NGKH, NGKT, KHDK)

Mỗi một chuyến đi có một mã số để phân biệt(MACD), thông tin về chuyến đi bao gồm: tên chuyến đi(TENCD), ngày khởi hành(NGKH), ngày kết thúc(NGKT) và số khách dự kiến(KHDK).

5. CTIETCD(MACD, MADD, SNLUU)

Chi tiết của chuyến đi (MACD) là các địa điểm mà chuyến đi đó đi qua (MADD), (SNLUU) là số ngày lưu lại tại điểm du lịch đó.

6.HUONGDAN(MACD, MAHDV)

Ghi nhận các hướng dẫn viên(MAHDV) tham gia hướng dẫn cho chuyến đi (MACD)

7. KHACH(MACD, HTKH, TUOI, DCKH, DTKH)

Ghi nhận thông tin về khách hàng đăng ký vào chuyến đi(MACD), bao gồm: họ tên(HTKH), tuổi (TUOIKH), địa chỉ(DCKH) và điện thoại liên lạc của khách(DTKH)

8. XEPV(MACD, BIENSO)

Ghi nhận các xe (BIENSO) phục vụ cho chuyến đi (MACD)

Dùng câu lệnh SQL để thực hiện các yêu cầu sau:

1. Tạo tất cả các bảng trên.
2. Cho biết danh sách các hướng dẫn viên của công ty.
3. Liệt kê đầy đủ thông tin về các điểm du lịch liên kết với công ty.
4. Cho biết đầy đủ thông tin về các địa điểm mà chuyến đi mã số CD2010 đi qua.
5. Liệt kê các lữ khách của chuyến đi CD2010.
6. Cho biết số lượng khách của chuyến đi CD2009.
7. Chuyến đi nào có số lượng khách lớn hơn số lượng dự kiến.
8. Cho biết tổng số lượng khách của tất cả các chuyến đi có ngày khởi hành trong tháng 12/2011.
9. Cho biết số ngày lưu lại trung bình, số ngày lưu lại lớn nhất, nhỏ nhất qua các điểm du lịch của chuyến đi CD2010.

10. Cho biết số lượng xe phụ vụ cho chuyến đi CD2010.

11. Điểm du lịch nào(Mã số, tên) có số ngày lưu lại lớn nhất của chuyến đi CD2010.

12. Điểm du lịch nào(Mã số, tên) có số ngày lưu lại lớn hơn số ngày lưu lại trung bình qua các điểm của chuyến đi CD2010.

13. Điểm du lịch SaPa(mã số SP) có bao nhiêu chuyến đi ghé qua và khai thác được bao nhiêu ngày(tổng số ngày phục vụ).

14. Liệt kê 3 điểm du lịch đầu tiên của chuyến đi CD2010 có số ngày lưu lại lớn nhất.

15. Liệt kê 3 điểm du lịch đầu tiên của chuyến đi CD2010 có số ngày lưu lại ít nhất.

16. Liệt kê các điểm du lịch của chuyến đi CD2010 ngoại trừ điểm có số ngày lưu ít nhất

17.Cho biết số lượng các điểm du lịch, tổng số ngày lưu lại tại các địa điểm, số lượng các hướng dẫn viên, số lượng xe phụ vụ cho từng chuyến đi có ngày khởi hành trong tháng 12/2010.

18. Chuyến đi nào (đầy đủ thông tin) có số lượng khách nhiều nhất.

19. liệt kê các chuyến đi, ngoại trừ chuyến đi điều động xe ít nhất.

20. Hướng dẫn viên nào chưa từng tham gia hướng dẫn.

3.2: Xét CSDL quản lý công chức viên chức CCVC, gồm các bảng ĐONVI, LOAIDVI, NGACHCBVC, NGACHBACLUONG, CBVC như sau:

1. DONVI(Madv, Tendv, loai) là quan hệ đơn vị gồm mã đơn vị, tên đơn vị, loại đơn vị.

2. LOAIDVI(Loai, Tenloaihinh), là quan hệ về loại hình tổ chức của đơn vị gồm loại hình và tên loại hình.

3. NGACHCBVC(Ngach, Tenngach): quan hệ ngạch cán bộ viên chức gồm có ngạch và tên ngạch.

4. NGACHBACLUONG(Ngach, Bac, Hesoluong): quan hệ ngạch bậc và hệ số lương của cán bọ viên chức gồm có ngạch, bậc lương, hệ số lương.

5. CBVC(MaDV, MaCC, HT, GT, NS, Ngach, Bac, Ngayxep) là quan hệ về cán bộ viên chức gồm có Mã đơn vị, mã công chức, họ tên, giới tính, ngày tháng năm sinh, ngạch lương, bậc lương, ngày xếp lương.

Hãy viết các câu lệnh truy vấn thông tin cho các câu hỏi sau đây:

1. Cho danh sách CBVC theo thứ tự Alphabet của họ tên của các CBVC.

2. Cho danh sách CBVC có hệ số lương từ 3.0 trở lên.

3. Cho biết tổng hệ số lương của từng đơn vị.

4. Cho danh sách CBVC thuộc các đơn vị mà tên có chữ "phòng".

5. Cho danh sách CBVC thuộc các đơn vị có tên loại hình tổ chức là "hành chính"

6. Cho danh sách CBVC thuộc ngạch "cán sự" có bậc 7 trở lên, hoặc những người có hệ số lương lớn hơn 3.06

7. Cho danh sách CBVC (mà) có thời hạn xếp lương tính đến cuối năm 2010 là 3 năm trở lên đối với các ngạch chuyên viên và chuyên viên chính; hoặc 2 năm trở lên đối với các ngạch còn lại.

8. Cho danh sách các CBVC có hệ số lương cao hơn hệ số lương của những người thuộc ngạch "cán sự".

3.3: Cho lược đồ CSDL sau dùng để quản lý việc học tập của sinh viên

KHOA(Makh, Vpkh)

Mỗi khoa có 1 mã số phân biệt (Makh), ta biết được vị trí của văn phòng khoa.

LOP(Malop, Makh)

Mỗi lớp có 1 mã số để phân biệt (Malop) thuộc duy nhất một khoa nào đó (Makh).

SINHVIEN(Masv, Hssv, Tensv, Nssv, Dcsv, Loptr, Malop)

Mỗi sinh viên có một mã số để phân biệt với các sinh viên khác (Masv), thông tin của từng sinh viên là họ và đệm (Hosv), tên (Tensv), năm sinh(Nssv), địa chỉ (Dcsv), có phải là lớp trưởng không (Loptr) và thuộc một lớp duy nhất nào đó (Malop)

MONHOC(Mamh, Tenmh, LT, TH)

Mỗi môn học có một mã số duy nhất (Mamh), có một tên (Tenmh), số tiết lý thuyết (LT), số tiết thực hành (TH)

CTHOC(Malop, HK, Mamh)

Mỗi lớp học (Malop) trong từng học kỳ (HK) sẽ có một số môn học (Mamh) được giảng dạy cho lớp đó.

DIEMSV(Masv, Mamh, Lan, Diem)

Ghi nhận điểm của các môn học (Mamh) ở lần thi nào (Lan), của sinh viên(Masv).

Yêu cầu: Viết câu lệnh SQL để thực hiện yêu cầu sau:

Cho biết danh sách lớp

Cho biết danh sách sinh viên lớp TH1.

Cho biết danh sách SV khoa CNTT

Cho biết chương trình học của lớp TH1

Điểm lần 1 môn CSDL của SV lớp TH1.

Điểm trung bình lần 1 môn CTDL của lớp TH1.

Số lượng SV của lớp TH2.

Lớp TH1 phải học bao nhiêu môn trong HK1 và HK2.

Cho biết 3 SV đầu tiên có điểm thi lần 1 cao nhất môn CSDL.

Cho biết sĩ số từng lớp.

Khoa nào đông SV nhất.

Lớp nào đông nhất khoa CNTT.

Môn học nào mà ở lần thi 1 có số SV không đạt nhiều nhất.

Tìm điểm thi lớn nhất của mỗi SV cho mỗi môn học (vì SV được thi nhiều lần).

Điểm trung bình của từng lớp khoa CNTT ở lần thi thứ nhất môn CSDL.

Sinh viên nào của lớp TH1 đã thi đạt tất cả các môn học ở lần 1 của HK2.

Danh sách SV nhận học bổng học kỳ 2 của lớp TH2, nghĩa là đạt tất cả các môn học của học kỳ này ở lần thi thứ nhất.

Biết rằng lớp TH1 đã học đủ 6 học kỳ, cho biết SV nào đủ điều kiện thi tốt nghiệp, nghĩa là đã đạt đủ tất cả các môn.

Chương 4
RÀNG BUỘC TOÀN VẸN, PHỤ THUỘC HÀM VÀ KHÓA

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 trình bày về các RBTV, phụ thuộc hàm và khóa. Đây là những vấn đề nền tảng rất quan trọng của CSDL quan hệ.

4.1. CÁC VẤN ĐỀ LIÊN QUAN ĐẾN RÀNG BUỘC TOÀN VẸN

4.1.1. Định nghĩa

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

Trong một CSDL, luôn luôn tồn tại rất nhiều mối liên kết 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:

Trong CSDL về quản lý học viên của một trường học đã cho trong các ví dụ của chương trước, chúng ta có một số ràng buộc toàn vẹn như sau:

R1 : Mỗi lớp học phải có một mã số duy nhất để phân biệt với mọi lớp học khác trong trường.

R2 : Mỗi lớp học phải thuộc một KHOA của trường.

R3 : Mỗi học viên có một mã số riêng biệt, không trùng với bất cứ học viên nào khác.

R4 : Mỗi học viên phải đăng ký vào một lớp của trường.

R5 : Mỗi học viên được thi tối đa 2 lần cho mỗi môn học.

R6 : 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.

Khi xác định một RBTV cần chỉ rõ:

Điều kiện (tức là nội dung) của RBTV, từ đó xác định cách biểu diễn.

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

Tầm ảnh hưởng của RBTV. Khả năng tính toàn vẹn dữ liệu bị vi phạm, và

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

4.1.2. Điều kiện của ràng buộc toàn vẹn

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 dưới đây).

Ví dụ 4.2:

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

HOADON (Sohoadon, Soloaihang, Tongtrigia).

DMHANG (Mahang, Tenhang, Donvitinh).

CHITIETHD (Sohoadon, Mahang, Soluongdat, Dongia, Trigia).

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”:

$\forall$hđ1, hđ2 $\in$ HOADON, hđ1 $\neq$ hđ2 $\Rightarrow$ hđ1.Sohoadon $\neq$ hđ2. Sohoadon.

R2:“Soloaihang = số bộ của CHITIETHD có cùng Sohoadon”:

$\forall$hđ $\in$ HOADON thì:

hđ.Soloaihang = COUNT (cthđ $\in$ CHITIETHD, cthđ.Sohoadon = hđ.Sohoadon)

R3 : “Tổng các trị giá của các mặt hàng trong CHITIETHD có cùng Sohoadon phải bằng Tongtrigia ghi trong HOADON”:

$\forall$hđ $\in$ HOADON thì:

hđ.Tongtrigia = SUM (cthđ.Trigia) đối với các cthđ $\in$ CHITIETHD sao cho : cthđ. Sohoadon= hđ. Sohoadon.

R4 : “Mỗi bộ của CHITIETHD phải có mã hàng thuộc về danh mục hàng”:

CHITIETHD [Mahang] $\subseteq$ DMHANG[Mahang]

hoặc biểu diễn bằng cách khác:

$\forall$ cthđ $\in$ CHITIETHD, $\exists$ hh $\in$ DMHANG

sao cho: cthđ.Mahang=hh.Mahang.

4.1.3. Bối cảnh của Ràng buộc toàn vẹn

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

Như trong ví dụ trên mục 4.1.2, bối cảnh của ràng buộc toàn vẹn R1 chỉ là một quan hệ HOADON; bối cảnh của ràng buộc toàn vẹn R2 và R3 là hai quan hệ HOADON và CHITIETHD; bối cảnh của ràng buộc toàn vẹn R4 là hai quan hệ CHITIETHD và DMHANG.

4.1.4. Tầm ảnh hưởng của ràng buộc toàn vẹn

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, đo đó cần xác định rõ thao tác nào dẫn đến việc cần phải kiểm tra RBTV.

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 bộ 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 4.3:

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_1

Quan hệ	Thêm	Sửa	Xóa
HOADON	+ (Sohoadon)	- (*)	+

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_2

Quan hệ	Thêm	Sửa	Xóa
HOADON	-	+ (Soloaihang)	+
CHITIETHD	+	-	+

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_3

Quan hệ	Thêm	Sửa	Xóa
HOADON	-	+ (Tongtrigia)	+
CHITIETHD	+	+ (Trigia)	+

Bảng tầm ảnh hưởng của ràng buộc toàn vẹn R_4

Quan hệ	Thêm	Sửa	Xóa
CHITIETHD	+ (Mahang)	- (*)	+
DMHANG	-	- (*)	+

Trong thực tế, ràng buộc toàn vẹn R1 là không cần thiết, bởi thuộc tính Sohoadon là khóa của quan hệ HOADON, do vậy nó luôn luôn phải là duy nhất và không được phép chứa giá trị rỗng; đồng thời không được phép sửa đổi.

Sau khi xây dựng các bảng tầm ảnh hưởng của từng RBTV trên các quan hệ liên quan, cần phải tổng hợp lại bằng cách xây dựng một bảng tầm ảnh hưởng tổng hợp các RBTV nhằm xác định tất cả các RBTV cần phải kiểm tra trên từng quan hệ. Bảng này gồm cột chủ từ là các RBTV, 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 RBTV trong CSDL.

Ví dụ 4.4:

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

Q.Hệ	HOADON			CHITIETHD			DMHANG		
RBTV	T	S	X	T	S	X	T	S	X
R1	+ (Sohd)	- (*)	+						
R2	-	+ (Soloaihang)	+	+	- (*)	+			
R3	-	+ (Tongtrigia)	+	+	+ (Trigia)	+			
R4				+	-	+	-	- (*)	+

Nhìn vào bảng tổng hợp trên chúng ta có thể thấy quan hệ HOADON khi thêm và xóa một bộ giá trị, phải kiểm tra ràng buộc toàn vẹn R1, R2 và R3; khi sửa giá trị thuộc tính Soloaihang thì phải kiểm tra ràng buộc toàn vẹn R2 và khi sửa giá trị thuộc tính tổng trị giá thì phải kiểm tra ràng buộc toàn vẹn R3. Quan hệ CHITIETHD khi được cập nhật cần kiểm tra 2 RBTV: R2 và R3; Quan hệ DMHANG cần kiểm tra ràng buộc toàn vẹn R4 khi xóa một bộ giá trị.

4.1.5. 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. CÁC LOẠI RÀNG BUỘC TOÀN VẸN

4.2.1. 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 4.5:

Trong quan hệ KQUATHI mô tả trong chương trước, do quy định mỗi học viên chỉ được thi một môn học tối đa là 2 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:

R1: $\forall kq \in$KQUATHI thì $0 \leq kq.Lanthi \leq 2$

R2: $\forall kq \in$KQUATHI thì $0 \leq kq.Diemthi \leq 10$

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:

R3: $\forall kq \in$KQUATHI | $kq.Diemthi$ (lần trước) $\leq kq.Diemthi \leq 10.0$

4.2.2. 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 4.6:

Quan hệ CHITIETHD trong CSDL quản lý hóa đơn bán hàng cho trong ví dụ trên có RBTV liên thuộc tính là:

∀cthđ ∈ CHITIETHD | cthđ.Trigia = cthđ.Soluongdat * cthđ.Dongia.

Ví dụ 4.7:

Quan hệ danh sách cán bộ - công chức Nhà nước CBCC với tập các thuộc tính:

{Madonvi, MaCBCC, Hoten, Gioitinh, Ngaysinh, Ngaytuyendung, NgachCBCC, Bậc, Hesoluong, Ngayxepluong}.

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: ∀cc ∈ CBCC | nếu cc.Gioitinh = Nam thì (Now() - cc.Ngay_sinh) / 365 trong khoảng 18 và 60. Nếu cc.Gioitinh = Nữ thì (Now() - cc.Ngay-sinh) / 365 trong khoảng 18 và 55.

R2: ∀cc ∈ CBCC | (cc.Ngaytuyendung - cc.Ngaysinh) / 365 ≥ 18 và cc.Ngaytuyendung ≤ Now().

Chú ý:

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.

4.2.3. 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 4.8:

Trong ví dụ trên vừa nêu, chúng ta thấy điểm thi không chỉ liên quan đến thuộc tính Lanthi 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: ∀kq ∈KQUATHI| Nếu kq.Lanthi = 1 thì 0 ≤ kq.Diemthi ≤ 10.0

hoặc:

Nếu kq.Lần thi > 1 thì ∃kq'∈KQUATHI

sao cho kq'.Lanthi = kq.Lanthi - 1 và kq.Diemthi ≥ kq'.Diemthi.

Ví dụ 4.9:

Giả thiết trong quan hệ ngạch bậc hệ số lương công chức.

NGACHBACLUONG (Mangach, Bac, Hesoluong).

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 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 sau của chương này.

4.2.4. 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 4.10:

Trong thể hiện của quan hệ CHITIETHD, sự tồn tại của mỗi bộ giá trị cthđ đều phụ thuộc vào sự tồn tại của một bộ giá trị hđ trong thể hiện của quan hệ HOADON sao cho hđ.Sohoadon = cthđ.Sohoadon, và phụ thuộc cả vào sự tồn tại của một bộ giá trị mh trong thể hiện của quan hệ DMHANG sao cho mh.Mahang = cthđ.Mahang.

Biểu diễn các RBTV này như sau:

RBTV1: "Mỗi bộ của CHITIETHD phải có một hóa đơn với Sohoadon tương ứng":

$\forall$cthđ $\in$ CHITIETHD, $\exists$hđ $\in$ HOADON sao cho cthđ.Sohoadon = hđ.Sohoadon.

hoặc biểu diễn bằng cách khác:

CHITIETHD[Sohoadon] $\subseteq$ HOADON[Sohoadon]

RBTV2: "Mỗi bộ của CHITIETHD phải có mã hàng thuộc về danh mục hàng":

$\forall$cthđ $\in$ CHITIETHD, $\exists$hh $\in$ DMHANG sao cho cthđ.Mahang=hh.Mahang

hoặc biểu diễn bằng cách khác:

CHITIETHD[Mahang] $\subseteq$ DMHANG[Mahang]

Ví dụ 4.11:

Trong CSDL về quản lý CBCC nêu trong các ví dụ trên, RBTV về phụ thuộc tồn tại giữa 2 quan hệ CBCC và NGACHBACLUONG được xác định:

$\forall$cbcc $\in$ CBCC, $\exists$ng $\in$ NGACHBACLUONG

sao cho (cbcc.Mangach = ng.Mangach) ^ (cbcc.Bac = ng.Bac)

hoặc biểu diễn bằng cách khác:

CBCC[Mangach, Bac] $\subseteq$ NGACHBACLUONG[Mangach, Bac]]

Ví dụ 4.12:

Trong CSDL về quản lý học viên đã nêu trong chương trước, các RBTV về phụ thuộc tồn tại gồm:

RBTV1 : "Mỗi LOPHOC đều phải thuộc một KHOA nhất định":

$\forall lh \in LOPHOC, \exists kh \in KHOA$ sao cho lh.Makhoa = kh.Makhoa.

hoặc biểu diễn qua phép chiếu quan hệ:

$LOPHOC[Makhoa] \subseteq KHOA[Makhoa]$.

RBTV2 : "Mỗi HOCVIEN đều phải thuộc một LOPHOC nhất định":

$\forall hv \in HOCVIEN, \exists lh \in LOPHOC$ sao cho hv.Malop = lh.Malop.

hoặc biểu diễn qua phép chiếu quan hệ:

$HOCVIEN[Malop] \subseteq LOPHOC[Malop]$.

RBTV3 : "Mỗi KQUATHI đều phải là của một HOCVIEN nhất định":

$\forall kq \in KQUATHI, \exists hv \in HOCVIEN$

sao cho kq.Mahocvien = hv.Mahocvien.Hoặc biểu diễn qua phép chiếu quan hệ:

$KQUATHI[Mahocvien] \subseteq HOCVIEN[Mahocvien]$.

RBTV4 : "Mỗi môn thi trong KQUATHI đều phải có tên trong danh sách các môn học" :

$\forall kq \in KQUATHI, \exists mh \in MONHOC$ sao cho kq.Mamon = mh.Mamon

hoặc biểu diễn qua phép chiếu quan hệ:

$KQUATHI[Mamon] \subseteq MONHOC[Mamon]$

Chúng ta có thể thấy vế phải của phép toán tập con ($\subseteq$) là phép chiếu trên thuộc tính khóa nội của một quan hệ, còn vế 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 RBTV loại này là RBTV về khóa ngoại. Phát biểu tổng quát về loại RBTV này là như sau:

R(U) và S(V) là hai quan hệ. $K_R \subseteq U$ là tập các thuộc tính khóa nội của quan hệ R; $K_S \subseteq V$ là tập các thuộc tính khóa nội của quan hệ S; và $W \subseteq V$ 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 tầm ảnh hưởng của loại RBTV này, các thao tác Thêm và Sửa một bộ giá trị của quan hệ R (vế phải 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 tầm ảnh hưởng có 2 dạng ứng với 2 trường hợp trên như sau:

a. Ứng với trường hợp khóa ngoại:

Quan hệ	Thêm	Sửa	Xóa
R	-	- (*)	+
S	+	+ (W)	-

b. Ứng với trường hợp lồng khóa:

Quan hệ	Thêm	Sửa	Xóa
R	-	- (*)	+
S	+	- (*)	-

4.2.5. Ràng buộc toàn vẹn tổng hợp (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 4.13:

Xét CSDL về quản lý học viên nêu trong ví dụ trên, RBTV liên quan hệ - liên bộ có thể được xác định: "Với mọi bộ giá trị của LOPHOC, 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 R6 đã nêu trong ví dụ tại mục 1.1.

Biểu diễn hình thức của RBTV này như sau:

$\forall$ lh $\in$ LOPHOC thì: nếu lh.Sohocvien > 0 thì:

h.Sohocvien = COUNT (hv $\in$ HOCVIEN, hv.Malop = lh.Malop).

Ví dụ 4.14:

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

1) HOADON (Sohoadon, Soloaihang, Tongtrigia).

2) CHITIETHD (Sohoadon, Mahang, Soluongdat, Dongia, Trigia).

3) DMHANG (Mahang, Tenhang, Donvitinh).

RBTV1 : "Soloaihang = số bộ của CHITIETHD có cùng Sohoadon" :

∀hđ ∈ HOADON thì:

hđ.Soloaihang = COUNT (cthđ ∈ CHITIETHD, cthđ.Sohoadon = hđ.Sohoadon)

RBTV2 : "Tổng tất cả các Trigia của các mặt hàng trong CHITIETHD có cùng Sohoadon phải bằng Tongtrigia của hóa đơn đó trong HOADON":

∀hđ ∈ HOADON thì hđ.Tongtrigia = SUM (cthđ.Trigia)

đối với các cthđ ∈ CHITIETHD sao cho : cthđ. Sohoadon= hđ. Sohoadon.

Chúng ta có thể nhận thấy trong CSDL này có sự dư thừa thông tin một cách cố ý. Đó là thuộc tính tính toán Trigia 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 lôgíc biểu thức sẽ không còn phù hợp nữa.

Bây giờ để đạt được dạng chuẩn (Normal Form) tốt hơn cho quan hệ CHITIETHD, chúng ta có thể loại bỏ thuộc tính Trigia. Khi đó RBTV2 được viết lại là:

∀hđ ∈ HOADON thì hđ.Tongtrigia = SUM (cthđ.Soluongdat * cthđ.Dongia)

đối với các cthđ ∈ CHITIETHD sao cho : cthđ. Sohoadon= hđ. Sohoadon.

4.3. PHỤ THUỘC HÀM (Functional Dependency)

4.3.1. Định nghĩa và biểu diễn phụ thuộc hàm

Trong Chương trước 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=A1A2...An. X,Y⊂ U là 2 tập con của tập thuộc tính U. Nếu tồn tại một ánh xạ f: X → Y thì ta nói rằng X xác định hàm Y, hay Y phụ thuộc hàm vào X, và ký hiệu là X → Y.

Định nghĩa hình thức của phụ thuộc hàm như sau:

Quan hệ Q (ABC) có phụ thuộc hàm A xác định B (ký hiệu là A → B) nếu:

∀q, q' ∈ 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)

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

Phụ thuộc hàm hiển nhiên: A → B được gọi là *phụ thuộc hàm hiển nhiên* nếu B ⊆ A.

Phụ thuộc hàm nguyên tố: A → 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) vào A nếu ∀A' ⊂ A đều không có A' → B.

Ví dụ 4.15: Trong CSDL quản lý hàng hóa, quan hệ HANG(MaH, TenH, SLTon) có các phụ thuộc hàm sau:

f1:MaH $\rightarrow$ tenH; f2: MaH $\rightarrow$ SLTon;

Các phụ thuộc hàm trên đều là nguyên tố.

Ví dụ 4.16:

Trong lược đồ CSDL quản lý hóa đơn bán hàng đã cho trong ví dụ trên, quan hệ HOADON(Sohoadon, Soloaihang, Tongtrigia) có các phụ thuộc hàm sau:

f1: Sohoadon $\rightarrow$ Soloaihang;

f2: Sohoadon $\rightarrow$ Tongtrigia;

Bởi vì Sohoadon là khóa của lược đồ quan hệ HOADON. 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 Soloaihang và Tongtrigia tất cả các mặt hàng của hóa đơn. Các phụ thuộc hàm trên đều là nguyên tố.

Quan hệ CHITIETHD (Sohoadon, Mahang, Soluongdat, Dongia, Trigia) có các phụ thuộc hàm sau:

f1: Sohoadon, Mahang $\rightarrow$ Soluongdat.

f2: Sohoadon, Mahang $\rightarrow$ Dongia.

f3: Sohoadon, Mahang $\rightarrow$ Trigia.

f4: Soluongdat, Dongia $\rightarrow$ Trigia.

f5: Mahang $\rightarrow$ Dongia

Thuộc tính Dongia phụ thuộc hàm không đầy đủ vào khóa (Sohoadon, Mahang), bởi vì nó chỉ phụ thuộc vào mặt hàng (thông qua Mahang).

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 F.

4.3.2. Bao đóng của tập phụ thuộc hàm và hệ luật dẫn Armstrong

4.3.2.1. Bao đóng của tập phụ thuộc hàm

Gọi 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 F nếu R thỏa các phụ thuộc hàm của F thì cũng thỏa X $\rightarrow$ Y và ký hiệu là: F |= X $\rightarrow$ Y.

Gọi F^+ là bao đóng (Closure) của F, tức là tập các phụ thuộc hàm được suy diễn lôgic từ F. Nếu F = F^+ thì ta nói 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 F hay không (tức là X $\rightarrow$ Y $\in$ 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.

4.3.2.2. Hệ luật dẫn Armstrong

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) 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 gia tăng: Nếu X $\rightarrow$ Y thì X Z $\rightarrow$ 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 bổ đề (ở đây không chứng minh):

Bổ 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.

Bổ 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.

Ví dụ 4.17:

Cho lược đồ quan hệ R(U), U=ABCDEGH và tập các phụ thuộc hàm 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 f1)
2. AB$\rightarrow$AB (tính chất phản xạ)
3. AB$\rightarrow$B (luật tách)
4. B$\rightarrow$D (cho trước - phụ thuộc hàm f2)
5. AB$\rightarrow$D (bắc cầu 3 & 4)
6. AB$\rightarrow$CD (hợp 1 & 5)
7. CD$\rightarrow$E (cho trước - phụ thuộc hàm f3)
8. AB$\rightarrow$E (bắc cầu 6 & 7). Kết thúc.

Ví dụ 4.18:

Cho lược đồ quan hệ R(U), U= ABCDEGHIJ và tập các phụ thuộc hàm F = {AB→E, AG→J, BE→I, E→G, GI→H }. Tìm chuỗi suy diễn AB→GH

Giải:

1. AB→E (cho trước - phụ thuộc hàm f1)
2. AB→AB (phản xạ)
3. AB→B (luật tách)
4. AB→BE (hợp của 1 & 3)
5. BE→I (cho trước - phụ thuộc hàm f3)
6. AB→I (bắc cầu 4 & 5)
7. E→G (cho trước - phụ thuộc hàm f4)
8. AB→G (bắc cầu 1 & 7)
9. AB→GI (hợp 6 & 8)
10. GI→H (cho trước - phụ thuộc hàm f5)
11. AB→H (bắc cầu 9 & 10)
12. AB→GH (hợp 8 & 11)

Bổ đề 3: X → Y được suy diễn lôgic từ F nhờ hệ tiên đề Amstrong khi và chỉ khi $Y \subseteq X_f^+$.

(X_f^+ là bao đóng của tập thuộc tính X đối với tập phụ thuộc hàm F chúng ta sẽ nghiên cứu trong mục sau đây)

4.3.3. Bao đóng của tập thuộc tính

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

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

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 C như sau

```
Void  Closure (X, F)
{
   ketqua=X;
   While  (có sự thay đổi trên tập ketqua)
   For  (mỗi pth W→Z  trong F)
```

If $W \subseteq$ **ketqua**

ketqua = ketqua $\cup$ Z

Return **ketqua**;

};

Tập **ketqua** là bao đóng của tập phụ thuộc hàm X

Ví dụ 4.19: cho tập phụ thuộc hàm F={A→BC, I→K, GB→H, CG→I, B→H} của quan hệ R(ABCDEFGHIK). Hãy tính bao đóng của tập thuộc tính AG, $(AG)^+$

Áp dụng thuật toán trên ta tính như sau:

Ban đầu ketqua=AG

Ta lần lượt xét tất cả các phụ thuộc hàm trong F:

A→BC có A$\subseteq$ ketqua nên ketqua=ketqua $\cup$ BC = AGBC

I→K có I$\not\subset$ ketqua nên ketqua vẫn giữ nguyên

GB→H có GB $\subseteq$ ketqua nên ketqua=ketqua $\cup$ H = AGBCH

CG→I có CG $\subseteq$ ketqua nên ketqua=ketqua $\cup$ I = AGBCHI

B→H có B$\subseteq$ketqua nhưng đã có H trong ketqua nên ketqua giữ nguyên

Quay lại từ đầu tập F lần 2:

A→BC có A$\subseteq$ ketqua nhưng đã có BC trong ketqua nên ketqua giữ nguyên

I→K có I$\subseteq$ ketqua nên ketqua=ketqua $\cup$ K = AGBCHIK

Tiếp tục các phụ thuộc hàm sau không làm thay đổi kết quả.

Lần này tập ketqua có thay đổi nên lại quay lại từ đầu tập F lần 3:

Lần này ketqua không thay đổi nên dừng.

Cuối cùng ta được $(AG)^+$ = ketqua= AGBCHIK.

Để xác định một phụ thuộc hàm có thuộc F^+

Để xác định phụ thuộc hàm X→Y có thuộc F^+ ta tính X^+ ta áp dụng bổ đề 3.

Nếu Y$\subseteq X^+$ thì X→Y thuộc F^+, trái lại thì không thuộc.

Ví dụ 4.20: F={CD→A, E→B, DB→C, C→D}

Phụ thuộc hàm nào sau đây thuộc F^+: DE→BC, AC→BE

Vì $(DE)^+$= DEBCA chứa BC nên BC nên DE→DC thuộc F^+

Vì $(AC)^+$=ACD không chứa BE nên AC→BE không thuộc F^+

4.3.4. Phủ và tương đương (Equivalence)

4.3.4.1. Định nghĩa

Hai tập phụ thuộc hàm F và G dựa trên Q được gọi tương đương. Ký hiệu là $F \equiv G$ nếu $F^+ = G^+$

$F \equiv G$ thì F được gọi là 1 phủ của G, hay G là một phủ của F.

Để chứng minh $F \equiv G$ ta đi chứng minh:

(i) $\forall X \rightarrow Y \in F \Rightarrow X \rightarrow Y \in G^+$

Để chứng minh điều này ta tính X_G^+, nếu $Y \subseteq X_G^+$ thì $X \rightarrow Y \in G^+$

(ii) $\forall W \rightarrow Z \in G \Rightarrow W \rightarrow Z \in F^+$

Tương tự ta tính W_F^+, nếu $Z \subseteq W_F^+$ thì $W \rightarrow Z \in F^+$

4.3.4.2. Phủ tối tiểu (Minimum Cover)

- Cho F là tập các phụ thuộc hàm dựa trên Q và một tập các RBTV dạng phụ thuộc hàm.

- Điều quan trọng ở đây là khi ta cập nhật CSDL thì hệ thống phải bảo đảm tất cả các phụ thuộc hàm trên không bị vi phạm.

- Ta biết rằng từ F ban đầu ta có tìm ra nhiều tập F_i tương đương với F bằng cách suy từ các phụ thuộc hàm của F. Quan hệ thoả các F_i thì cũng thoả F và ngược lại.

Giả sử ta có $F=\{A \rightarrow B, B \rightarrow C\}$

Thì ta cũng có : $F_1=\{A \rightarrow B, B \rightarrow C, A \rightarrow C\}$, $F_2= \{A \rightarrow B, B \rightarrow C, A \rightarrow C, AB \rightarrow BC\}$, …

và $F_1 \equiv F$, $F_2 \equiv F$

Vấn đề được đặt ngược lại là nếu cho F thì ta có thể tìm ra được tập phụ thuộc hàm đơn giản hơn F và tương đương với F.

Tập phụ thuộc hàm có ít phụ thuộc hàm hơn F và các phụ thuộc hàm đó cũng đơn giản hơn. Và như vậy hệ thống dễ dàng kiểm tra hơn.

Định nghĩa thuộc tính dư thừa (Extraneous):

Một thuộc tính được gọi là dư thừa trong tập phụ thuộc hàm F nếu như ta bỏ nó ra khỏi các phụ thuộc hàm mà bao đóng của F vẫn không đổi.

Cho $X \rightarrow Y$ là một phụ thuộc hàm của F

- A là thuộc tính dư thừa trong X nếu $A \in X$ và $F \equiv (F - \{X \rightarrow Y\}) \cup \{(X - A) \rightarrow Y\}$

- B là thuộc tính dư thừa trong Y nếu $B \in Y$ và $F \equiv (F - \{X \rightarrow Y\}) \cup \{X \rightarrow (Y - B)\}$

Phủ tối tiểu của F ký hiệu là Fc là tập phụ thuộc hàm tương đương với F và thoả các tính chất sau:

Không có phụ thuộc hàm nào trong Fc chứa các thuộc tính dư thừa.

Không có 2 phụ thuộc nào cùng vế trái.

Thuật toán tìm phủ tối tiểu từ F.

do

Sử dụng công thức hợp để thay thế các phụ thuộc hàm có cùng vế trái:

$X_1 \rightarrow Y_1$ và $X_1 \rightarrow Y_2 \Rightarrow X_1 \rightarrow Y_1Y_2$

Nếu có một phụ thuộc hàm$X \rightarrow Y$ có các thuộc tính dư thừa bên vế trái hay vế phải thì xoá nó khỏi $X \rightarrow Y$

while (F không thay đổi)

Ví dụ 4.21: Cho F là tập phụ thuộc hàm trên lược đồ quan hệ (ABC) như sau:

$F=\{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$

Tìm phủ tối tiểu của F

Có hai phụ thuộc hàm cùng vế trái: $A \rightarrow BC$, $A \rightarrow B$

Ta thay thế hai phụ thuộc hàm trên bằng phụ thuộc hàm $A \rightarrow BC$

Tập phụ thuộc hàm mới: $F=\{A \rightarrow BC, B \rightarrow C, AB \rightarrow C\}$

A là thuộc tính dư thừa trong $AB \rightarrow C$ vì $F \equiv (F- \{AB \rightarrow C\}) \cup \{B \rightarrow C\}$

Do đó $AB \rightarrow C$ được thay bằng $B \rightarrow C$

Tập phụ thuộc hàm mới là: $F=\{A \rightarrow BC, B \rightarrow C\}$

C là thuộc tính dư thừa trong $A \rightarrow BC$ vì $F \equiv (F -\{A \rightarrow BC\}) \cup \{A \rightarrow B\}$

Do đó $A \rightarrow BC$ được thay thế bằng $A \rightarrow B$

Tập phụ thuộc hàm mới là: $F=\{A \rightarrow B, B \rightarrow C\}$

Vậy phủ tối tiểu của F là: $Fc=\{A \rightarrow B, B \rightarrow C\}$

4.4. KHÓA

4.4.1. Khái niệm khóa

Trong chương 2 chúng ta đã có rất nhiều định nghĩa khóa cho quan hệ.

Bây giờ chúng ta nghiên cứu kỹ hơn về khóa dựa trên lược đồ quan hệ và tập phụ thuộc hàm.

Định nghĩa: Khóa của quan hệ theo phụ thuộc hàm

Cho lược đồ quan hệ R xác định trên tập thuộc tính U, và tập phụ thuộc hàm F. Tập con bất kỳ K⊆U là khóa của lược đồ quan hệ R khi và chỉ khi thỏa mãn các điều kiện sau:

K->U (K xác định hàm các thuộc tính của U)

Không tồn tại K'⊂ K mà K'->U (Không tồn tại con của K mà cũng xác định được U như K).

Giá trị các thành phần của khóa không thể nhận giá trị NULL hay các giá trị không xác định.

Các thuộc tính là các phần tử của khóa gọi là các thuộc tính khóa, ngược lại, các thuộc tính không chứa trong khóa gọi là các thuộc tính không khóa.

4.4.2. Thuật toán tìm một khóa

Thuật toán 1:

Ý tưởng: Xuất phát từ một siêu khóa K (có thể là U), lần lượt xem xét và loại bỏ thuộc tính A nếu $(K - Ai)^+ = U$

Input: R(U,F), U = A1A2... An

Output: Tập thuộc tính khóa K

Bước 1: K=U;

Bước 2: While (Ai ∈K)

If $((K - Ai)^+ = U)$ K= K – Ai

K còn lại chính là một khóa cần tìm.

Nếu muốn tìm các khóa khác nhau (nếu có) của lược đồ quan hệ, ta có thể thay đổi thứ tự loại bỏ các phần tử của K.

Ví dụ 4.22: Cho lược đồ quan hệ R(U), U=ABC, F={A->B, A->C, B-A}. Hãy tìm một khóa của R.

Giải:

K= ABC

Loại thuộc tính A do $(K-A)^+$= ABC = U nên K = BC

Thuộc tính B không loại được do $(K-B)^+ = C \neq U$ nên K = BC

Loại thuộc tính C do $(K-C)^+$= BCA = U nên K = B

Vậy một khóa của R là B.

Ví dụ 4.23: Cho R(U), U=ABCDEH với F = { AB→C, CD→E, EC→A, CD→H, H→B }. Hãy tìm khóa của R

Giải:

K= ABCDEH

Loại thuộc tính A do $(K\text{-}A)^+ = ABCDEH = U$ nên K = BCDEH

Loại thuộc tính B do $(K\text{-}B)^+ = CDEHAB = U$ nên K = CDEH

Không loại thuộc tính C do $(K\text{-}C)^+ = DEHB \neq U$ nên K = CDEH

Không loại thuộc tính D do $(K\text{-}D)^+ = CEHBA \neq U$ nên K = CDEH

Loại thuộc tính E do $(K\text{-}E)^+ = CDEHAB = U$ nên K = CDH

Loại thuộc tính H do $(K\text{-}H)^+ = CDEHAB = U$ nên K = CD

Vậy khóa của R là K = CD.

Thuật toán 2: Ta có thể sử dụng thuật toán khác để tìm một khóa của quan hệ.

Biểu diễn lược đồ quan hệ R (U,F) 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 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.

Thuật toán

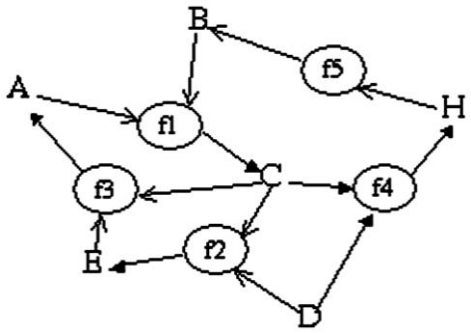
Bước 1: Xuất phát từ tập các nút gốc (X).

Bước 2: Tính bao đóng của tập thuộc tính X, X^+

Bước 3: Nếu $X^+ = U$ thì X là khoá, ngược lại thì bổ sung một thuộc tính không thuộc nút lá vào X rồi tìm bao đóng. Quay lại **bước 2.**

Ví dụ 4.24:

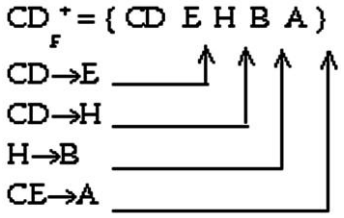
Cho R(U), U=ABCDEH với F = {AB→C, CD→E, EC→A, CD→H, H→B}. Hãy tìm khóa của R.



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.

$D^+ = \varnothing$, 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 $(CD)^+$ = CDEHBA do đó CD là khóa của R.

Ví dụ 4.25: Cho quan hệ GIANGDAY(MS_CBGD, MS_MH, T_CBGD, HH_CBGD, ML, TSSV) với tập phụ thuộc hàm:

F = { MS_CBGD → T_CBGD;

MS_MH → HH_CBGD, ML;

HH_CBGD → ML;

MS_CBGD → HH_CBGD;

MS_CBGD, MS_MH → TSSV}

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.

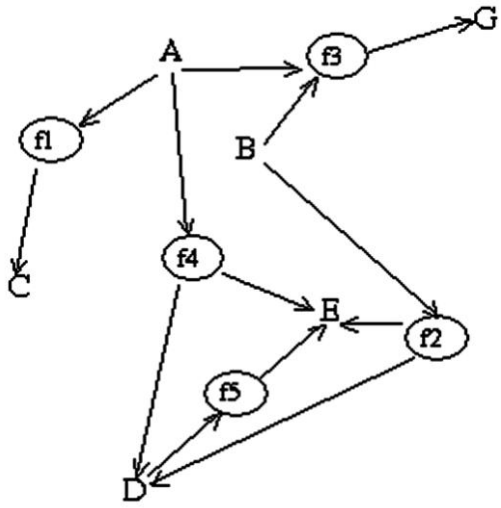
Hãy xác định khóa của quan hệ GIANGDAY.

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 GIANGDAY và tập phụ thuộc hàm F được viết ngắn gọn lại là:

R(U), U=ABCDEG

F = { A→C, B→DE, D→E, A→ED, AB→G }

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 = AB

X^+ = ABCDEG

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

$A \rightarrow C$

$B \rightarrow D$

$D \rightarrow E$

$AB \rightarrow G$

Vậy AB là khóa của R. Tức là, khóa của quan hệ GIANGDAY là {MS_CBGD, MS_MH }.

4.4.3. Một số tính chất của khóa

Cho lược đồ quan hệ R(U,F) với U là tập thuộc tính, F là tập phụ thuộc hàm

- Giao của các khóa ký hiệu là X được xác định như sau:

$$X = U - \bigcup_{L \rightarrow R \in F} (R - L)$$

X được tính bằng tập thuộc tính U trừ đi hợp của các vế phải trừ vế trái của phụ thuộc hàm.

- Lược đồ quan hệ có một khóa duy nhất khi $X^+ = U$.

4.4.4. Thuật toán tìm nhiều khóa

Cho lược đồ quan hệ R(U,F) với tập thuộc tính U=A1A2…Ai,…, F là tập phụ thuộc hàm.

Thuật toán tìm tất cả các khóa của lược đồ quan hệ:

Input: R(U,F)

Output: Tất cả các tập thuộc tính khóa K của R.

Bước 1: Tìm tập X là giao của các khóa theo công thức trong mục 4.4.3

Bước 2: Tính X^+

Nếu $X^+ = U$ thì quan hệ R chỉ có một khóa duy nhất K=X

Ngược lại thì R có nhiều hơn một khóa và chuyển sang bước 3.

Bước 3: Tính $(X \cup Ai)^+ = U$ thì K = X∪Ai là các khóa của R.

Ví dụ 4.25: Cho lược đồ quan hệ R(U), U=ABC và tập phụ thuộc hàm F={A->B, A->C, B->A}. Hãy tìm tất cả các khóa của R.

Giải:

Ta có giao của các khóa là X = {ABC} - {BCA} = ∅

Ta thấy $X^+ = \emptyset \neq U$ nên quan hệ có nhiều hơn một khóa.

Ta có A^+=ABC=U

Ta có B^+=ABC=U

Vậy quan hệ R có 2 khóa A hoặc B.

Ví dụ 4.26: Cho lược đồ quan hệ R(U), U=ABCDG và tập phụ thuộc hàm F={B->C, C->B, A->GD}. Hãy tìm tất cả các khóa của R.

Ta có giao của các khóa là X = {ABCDG} - {BCGD} = {A}

Vì A^+ = AGD ≠ U nên quan hệ có nhiều hơn một khóa.

Bố xung thêm các thuộc tính khác vào cùng với giao của các khóa ta sẽ có các khóa khác nhau của lược đồ quan hệ.

Bổ xung thêm B, ta có $(AB)^+$ =AGDBC = U

Bổ xung thêm C, ta có $(AC)^+$ =AGDCB = U

Vậy quan hệ R có 2 khóa AB hoặc AC.

BÀI TẬP CHƯƠNG 4

4.1: 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)

Quy tắc: 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-ỨNG (Mã-TK, TK-Đối-ứng);

Quy tắc: 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).

Quy tắc: 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).

Quy tắc: Từ sổ chứng từ (SỔ-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 ảnh 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.

1. 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.

2. Xác định khóa của các quan hệ trong CSDL nêu trên.

4.2: Vận dụng hệ tiên đề Armstrong để tìm chuỗi suy diễn:

Cho R(U), U=ABCDEGH với F = { AB-> C; B-> D; CD-> E; CE-> GH; G-> A }

(a) Tìm chuỗi suy diễn cho AB-> E.

(b) Tìm chuỗi suy diễn cho BG-> C.

(c) Tìm chuỗi suy diễn cho AB->G.

4.3: Cho R(U), U=ABCDEG với tập phụ thuộc hàm F={A-> C, AC->D, D->EG, G->B, A-> D, CG-> A}

1. Chứng minh rằng R thỏa F thì R thỏa các phụ thuộc hàm AB->E, AD->BC. Hay nói cách khác các phụ thuộc hàm AB->E, AD->BC được suy diễn logic từ F.

2. Tính bao đóng của các tập thuộc tính: A^+, $(AC)^+$

3. Tìm khóa của quan hệ R

4.4: Xác định khóa của các lược đồ quan hệ sau:

Q1 (ABCDEH)

với F = { AB-> C; CD-> E; AH-> B; B-> D; A-> D }

Q2 (ABCDMNPQ)

với F = { AM-> NB; BN-> CM; A-> P; D-> M; PC-> A; DQ-> A }

Q3 (MNPQRSTUW)

với F = { M-> W; MR-> T; T-> R; QR-> T; M-> U; MT-> P; NP-> Q; UW-> R }

Chương 5
DẠNG CHUẨN VÀ CHUẨN HOÁ

Chương này trình bày về các dạng chuẩn, chính là công cụ dùng để đánh giá một CSDL tốt xấu như thế nào?. Ở đây cũng trình bày về cách thức chuẩn hóa lược đồ CSDL để có một dạng chuẩn tốt hơn.

5.1. DẠNG CHUẨN

5.1.1. Thiết kế kém gây nguy hiểm cho CSDL

Một trong hai nguyên nhân sau đây do thiết kế kém sẽ gây nguy hiểm cho CSDL.

Trùng lắp thông tin không có khả năng trình bày thông tin một cách chắc chắn

Ví dụ 5.1: Cho một lược đồ quan hệ dùng để ghi nhận giảng viên và lớp giảng dạy của giảng viên

GIANGDAY(MONHOC,SOTIET,LOP,GV,HV,DC)

Các phụ thuộc hàm: MONHOC → SOTIET; MONHOC, LOP → GV; GV→ HOCVI,DC

Xét một tình trạng dữ liệu như sau:

MONHOC	SOTIET	LOP	GV	HOCVI	DC
CSDL	60	CNTT1	D.T.Hien	TS	HN
CSDL	60	CNTT2	D.T.Hien	TS	HN
CTDL	45	CNTT1	H.V.A	ThS	HP
CTDL	45	CNTT2	H.V.A	ThS	HP

Do có phụ thuộc hàm MONHOC → SOTIET nên số tiết của dòng thứ 2 và dòng thứ 4 gây nên trùng lắp thông tin. Tương tự phụ thuộc hàm GV → HOCVI, DC nên học vị và địa chỉ của dòng thứ 2 và dòng thứ 4 gây nên trùng lắp thông tin. Các dữ liệu gây trùng lắp thông tin là các dữ liệu có thể suy đoán được một cách chắc chắn và duy nhất từ phụ thuộc hàm.

Ở đây để lưu học vị và địa chỉ của một giảng viên thì giảng viên đó phải tham gia giảng dạy một lớp nào đó. Để giải quyết vấn đề lưu thông tin các giảng viên không tham gia giảng dạy người ta dùng giá trị NULL cho các thuộc tính MONHOC, SOTIET, LOP. Như vậy, lược đồ quan hệ này lưu trữ hai thông tin của hai đối tượng khác nhau một là giảng dạy của các giảng viên tham gia giảng dạy, hai là thông tin của các giảng viên không tham gia giảng dạy. Vấn đề nảy sinh ở đây là khi ta chỉ cần

cập nhật việc giảng dạy ta phải đảm bảo không gây ảnh hưởng tới các giảng viên không tham gia giảng dạy và ngược lại. Như vậy thông tin lưu trữ ở lược đồ quan hệ này không chắc chắn.

5.1.2. Phân rã

Từ một lược đồ quan hệ kém chất lượng ban đầu cùng với tập phụ thuộc hàm của nó ta tuân theo một nguyên tắc nào đó phân rã thành những lược đồ quan hệ chất lượng hơn.

Ví dụ 5.2: Phân rã lược đồ quan hệ GIANGDAY thành hai lược đồ TKB và GV

TKB(MONHOC, SOTIET, LOP)

GV(LOP,GV,HOCVI,DC)

Tình trạng dữ liệu của hai lược đồ trên như sau:

TKB = $\Pi_{MONHOC, SOTIET, LOP}$(GIANGDAY) GV = $\Pi_{LOP,GV, HOCVI, DC}$(GIANGDAY)

MONHOC	SOTIET	LOP
CSDL	60	CNTT1
CSDL	60	CNTT2
CTDL	45	CNTT1
CTDL	45	CNTT2

LOP	GV	HOCVI	DC
CNTT1	D.T.Hien	TS	HN
CNTT2	D.T.Hien	TS	HN
CNTT1	H.V.A	ThS	HP
CNTT2	H.V.A	ThS	HP

Sau đây là 2 rắc rối xảy ra

Để trả lời câu hỏi "Cho biết thông tin của giảng viên dạy CSDL của lớp CNTT1 ta phải kết nối tự nhiên hai quan hệ TKB và GV. Kết quả như sau:

MONHOC	SOTIET	LOP	GV	HOCVI	DC
CSDL	60	CNTT1	D.T.Hien	TS	HN
CSDL	60	CNTT1	H.V.A	ThS	HP
CSDL	60	CNTT2	D.T.Hien	TS	HN
CSDL	60	CNTT2	H.V.A	ThS	HP
CTDL	45	CNTT1	H.V.A	ThS	HP
CTDL	45	CNTT1	D.T.Hien	TS	HN
CTDL	45	CNTT2	H.V.A	ThS	HP
CTDL	45	CNTT2	D.T.Hien	TS	HN

Ta thấy rằng có tới hai giáo viên dạy môn CSDL của lớp CNTT1 trong khi thông tin ban đầu chỉ có D.T.Hien

→ Vấn đề này gọi là phân rã không bảo toàn thông tin.

Xét phụ thuộc hàm trên lược đồ phân rã:

TKB(MONHOC, SOTIET, LOP) MONHOC → SOTIET

GV(LOP, GV, HOCVI, DC) GV → HOCVI, DC

Từ hai phụ thuộc hàm trên ta không thể suy ra được phụ thuộc hàm MONHOC, LOP → GV. Như vậy, hai phụ thuộc hàm trên không đảm bảo kiểm tra các ràng buộc toàn vẹn do 3 phụ thuộc hàm ban đầu gây ra nên.

→ Vấn đề này gọi là phân rã không bảo toàn phụ thuộc hàm.

Sau đây, ta sẽ xét các quy tắc phân rã sao cho không vi phạm hai vấn đề trên.

5.1.2.1. Phân rã bảo toàn thông tin

Cho lược đồ quan hệ Q. Ta có định nghĩa sau:

Tập {Q1, Q2,…,Qn} là một phân rã của Q nếu:

$$Q = Q1 \cup Q2 \cup \ldots \cup Qn$$

Một cách tổng quát TQ là một quan hệ của Q thì:

$$TQ \subseteq \Pi_{R1}(TQ) \bowtie \Pi_{R2}(TQ) \bowtie \ldots \bowtie \Pi_{Rn}(TQ)$$

Phân rã thông tin trên bảo toàn thông tin nếu:

$$TQ = \Pi_{R1}(TQ) \bowtie \Pi_{R2}(TQ) \bowtie \ldots \bowtie \Pi_{Rn}(TQ)$$

Điều kiện để phân rã bảo toàn thông tin

Cho Q và F là tập phụ thuộc hàm định nghĩa trên Q

Q1 và Q2 là một phân rã bảo toàn thông tin trên Q nếu thoả một trong hai phụ thuộc hàm sau:

$$\mathbf{Q1 \cap Q2 \rightarrow Q1 \backslash Q2} \text{ hoặc } \mathbf{Q1 \cap Q2 \rightarrow Q2 \backslash Q1}$$

Vì vậy nếu $X \rightarrow Y \in F^+$ thì phân rã sau sẽ bảo toàn thông tin

Q1(XY)

Q2(R-Y)

Thật vậy, vì Q1 có X→Y và Q1∩Q2=X, Q2\Q1=Y do đó Q1∩Q2→Q1\Q2

Ví dụ 5.3: Lược đồ GIANGDAY nếu phân rã thành hai lược đồ sau thì bảo toàn thông tin.

Q1(MONHOC, SOTIET, LOP, GV)

Q2(GV, HOCVI, DC)

vì Q1∩Q2=GV mà GV →HOCVI,DC

Phương tiện để kiểm tra phân rã bảo toàn thông tin:

Phương tiện để kiểm tra phân rã bảo toàn thông tin là kỹ thuật Tableau: là một bảng T như sau:

	1	2	…	m
Q1				
Q2				
…				
Qn				

m cột cho m thuộc tính của Q

n dòng cho n quan hệ phân rã

(i,j) = a_j nếu Qi có chứa thuộc tính thứ j của Q

= b_k nếu ngược lại, k bắt đầu từ 1 và tăng dần.

Áp dụng luật phụ thuộc hàm để biến đổi bảng T thành T^* theo thuật toán sau:

While ($X \rightarrow A \in F$)

{

Chọn dòng W1 và W2 sao cho W1.X = W2.X

If (W1.A != W2.A)

{

Nếu W1.A = a_j và W2.A = b_k thay W2.A bằng W1.A

Nếu W1.A = b_k và W2.A = a_j thay W1.A bằng W2.A

Nếu W1.A = a_j và W2.A = b_k thay W2.A bằng W1.A

}

}

Cuối cùng xem bảng kết quả nếu trong bảng xuất hiện hàng gồm các kí hiệu a_1, a_2, a_3, …, a_m thì phân rã bảo toàn thông tin.

Ví dụ 5.4: Kiểm tra Q1(MONHOC, SOTIET, LOP, GV), Q2(GV, HOCVI, DC)

F = {MONHOC $\rightarrow$ SOTIET; MONHOC, LOP $\rightarrow$ GV; GV $\rightarrow$ HOCVI, DC}

T

	MONHOC	SOTIET	LOP	GV	HOCVI	DC
Q1	a_1	a_2	a_3	a_4	b_1	b_2
Q2	b_3	b_4	b_5	a_4	a_5	a_6

Từ GV → HOCVI, DC ta thay thế b_1 thành a_5 và b_2 thành a_6

T*

	MONHOC	SOTIET	LOP	GV	HOCVI	DC
Q1	a_1	a_2	a_3	a_4	a_5	a_6
Q2	b_3	b_4	b_5	a_4	a_5	a_6

Như vậy ta được dòng thứ nhất toàn a_j suy ra phân rã trên bảo toàn thông tin.

5.1.2.2. Phân rã bảo toàn phụ thuộc hàm

Cho lược đồ quan hệ Q và tập phụ thuộc hàm F xác định trên Q

Phân rã Q thành {Q1, Q2…Qn} thì mỗi Q sẽ xác định một tập phụ thuộc hàm F_i:

$$Fi = \{X \to Y : XY \subseteq Qi \text{ và } X \to \ \in F^+\}$$

Fi được gọi là tham chiếu của F^+ lên Qi

Phân rã trên bảo toàn phụ thuộc hàm nếu:

Đặt F' = F1 ∪ F2 ∪ … ∪ Fn

thì F' = F (nghĩa là $F'^+ = F^+$)

Để kiểm tra phân rã bảo toàn phụ thuộc hàm ta đi kiểm tra F1∪F2∪ … ∪ Fn ≡ F

Lưu ý: Khi tính các Fi thường hay thiếu sót các phụ thuộc hàm vì Fi là chiếu của F^+ lên Qi chứ không phải F lên Qi. Như vậy để tính đầy đủ Fi của Qi ta tính bao đóng của tất cả các tập con thực sự của Qi.

$X \subset Qi$. Nếu $X^+ \cap Qi \neq X$ thì $(X \to (X^+ \cap (Qi - X)) \in Fi$.

Ví dụ 5.5: Cho Q(ABCD), F = {A → B, B → C, C → D, D → A}

Phân rã Q thành {Q1(AB), Q2(BC), Q3(CD)} sẽ dễ dàng nhầm lẫn:

Q1(AB), F1 = {A → B}

Q1(BC), F2 = {B → C}

Q1(CD), F1 = {C → D}

Lúc này F' = F1 ∪ F2 ∪ F3 = {A → B, B → C, C → D}

Rõ ràng là F' không tương đương với F vì $F'^+ \neq F^+$ do $D \to A \notin F'^+$

Như vậy nếu vội vã kết luận phân rã trên không bảo toàn phụ thuộc hàm là sai

Thực ra:

Q1(AB) $A_f^+ = ABCD \Rightarrow A \to B \in F1$ $B_f^+ = BCDA \Rightarrow B \to A \in F1$

Vậy F1 = {A → B, B → A}

Q2(BC) $B_f^+ = BCDA \Rightarrow B \to C \in F2$ $C_f^+ = CDAB \Rightarrow C \to B \in F2$

Vậy F2 = {$B \to C$, $C \to B$}

Q3(CD) $C_f^+ = CDAB \Rightarrow C \to D \in F3$ $D_f^+ = DABC \Rightarrow D \to C \in F3$

Vậy F3 = {$C \to D$, $D \to C$}

Vậy F' = F1 $\cup$ F2 $\cup$ F3 = { $A \to B$,$B \to A$, $B \to C$, $C \to B$, $C \to D$, $D \to C$}

Ta tính được $F'^+ = F^+ \Rightarrow F' \equiv F$

Kết luận: Phân rã trên bảo toàn phụ thuộc hàm

5.1.3. Dạng chuẩn (Normal Form-NF)

- **Thuộc tính khoá**: Thuộc tính tham gia vào bất kỳ khoá nào đó của quan hệ chứa nó.

- Ngược lại gọi là thuộc tính không khoá.

Ví dụ 5.6: Q(ABCDEF) A, B, D, E là các thuộc tính khoá. C, F là các thuộc tính không khoá.

- **Thuộc tính đơn**: Miền giá trị của nó không phải là tích hợp của các miền giá trị khác

- $X \to A$ là **phụ thuộc hàm nguyên tố** của tập F nếu: Không $\exists$ Y là tập con thực sự của X, $Y \to A \in F^+$

Ví dụ 5.7: Cho F = {$AB \to C$, $B \to C$} thì:

$AB \to C$: không là phụ thuộc hàm nguyên tố vì có $B \to C$

$B \to C$: là phụ thuộc hàm nguyên tố

- A là **thuộc tính phụ thuộc đầy đủ** vào X nếu $X \to A$ là phụ thuộc hàm nguyên tố .

Ví dụ trên thì C là thuộc tính phụ thuộc đầy đủ vào B chứ không phụ thuộc đầy đủ vào AB.

5.1.3.1. Dạng chuẩn 1(1NF)

Lược đồ quan hệ Q ở dạng 1NF nếu tất cả các thuộc tính của Q đều là thuộc tính đơn.

Lược đồ CSDL *C* ở 1NF nếu tất cả các Qi của *C* đều ở 1NF

Đây là dạng chuẩn đơn giản nhất, nó không chú ý đến các phụ thuộc hàm do đó có rất nhiều trùng lắp thông tin do các phụ thuộc hàm trên gây ra.

5.1.3.2. Dạng chuẩn 2 (2NF)

Lược đồ quan hệ Q ở dạng 2NF nếu ở 1NF và tất cả thuộc tính không khoá đều phụ thuộc đầy đủ vào khoá.

Lược đồ CSDL ở *C* dạng 2NF nếu tất cả Qi của *C* đều ở 2NF

Ví dụ 5.8: Cho Q(ABCD), F = {A → C, B → D}, không đạt 2NF vì:

Khoá chính là AB

C,D là hai thuộc tính không khoá

AB → C không là phụ thuộc hàm nguyên tố vì có A → C

AB → D không là phụ thuộc hàm nguyên tố vì có B → D

C và D không phụ thuộc đầy đủ vào khoá.

Xét một tình trạng Q có sự trùng lắp thông tin (các giá trị trong ngoặc là trùng lắp)

Q	A	B	C	D
	(a1)	b1	(c1)	d1
	(a1)	b2	(?)	d2
	a2	(b3)	c2	(d3)
	a3	(b3)	c3	(?)

Ví dụ 5.9: Cho Q(ABCD), F = {AB → C, C → D} ở 2NF vì:

Khoá chính là AB.

C,D là hai thuộc tính không khoá. AB → C và AB → D đều là các phụ thuộc hàm nguyên tố.

⇒ C và D đều là phụ thuộc đầy đủ vào khoá.

Xét một tình trạng Q có sự trùng lắp thông tin:

Q	A	B	C	D
	a1	b1	(c1)	(d1)
	a2	b2	(c1)	(?)

Ta thấy rằng ở ví dụ 5.9, C → D gây ra trùng lắp thông tin vì thuộc tính không khoá D phụ thuộc bắc cầu vào khoá(nghĩa là phụ thuộc hàm khoá → D suy diễn nhờ qui tắc bắc cầu Armstrong).

5.1.3.3. Dạng chuẩn 3 (3NF)

Lược đồ quan hệ Q ở dạng 3NF nếu ở 2NF và tất cả các thuộc tính không khoá không phụ thuộc bắc cầu vào khoá.

Định nghĩa này có thể định nghĩa lại như sau:

Lược đồ quan hệ Q ở dạng 3NF nếu ở 2NF và tất cả phụ thuộc hàm không hiển nhiên X → Y của F^+ thoả một trong hai điều kiện sau:

X là một siêu khoá (X chứa một khoá nào đó)

(ii) Mỗi thuộc tính trong tập (Y - X) nằm trong một khoá nào đó

Lược đồ CSDL *C* ở dạng 3NF nếu tất cả các Qi của *C* đều ở dạng 3NF

Ví dụ 5.10: Cho Q(ABCD), F = {AB → CD} ở dạng 3NF

Vì AB → CD có vế trái là một siêu khoá.

Ví dụ 5.11: Cho Q(ABCDE), F = {AB → CDE, B → D, DE → ABC} ở dạng 3NF vì:

AB → CD có vế trái là một siêu khoá.

B → D có (VP) – (VT) = D chứa trong khoá DE.

DE → ABC có vế trái là một siêu khoá.

Xét một tình trạng Q có sự trùng lắp thông tin:

Q	A	B	C	D	E
	a1	(b1)	c1	(d1)	e1
	a2	(b1)	c2	(?)	e1

Ở dạng 3NF nếu có nhiều khoá sẽ có khả năng trùng lặp thông tin do các phụ thuộc hàm loại (ii) trong định nghĩa dạng chuẩn 3.

5.1.3.4. Dạng chuẩn Boyce –codd (BCNF)

Lược đồ quan hệ Q ở BCNF nếu ở dạng 1NF và tất cả phụ thuộc hàm không hiển nhiên X → Y của F^+ thì X là một siêu khoá (X chứa một khoá nào đó).

Lược đồ CSDL C ở dạng BCNF nếu tất cả các Qi của C đều ở dạng BCNF.

Ví dụ 5.12: Cho Q(ABCD), F{AB → CD, D → AB} ở dạng BCNF vì:

AB → CD có vế trái là một siêu khoá

D → AB có vế trái là một siêu khoá

D → C $\in F^+$ vế trái là một siêu khoá

Ở dạng chuẩn BCNF không có sự trùng lắp thông tin do phụ thuộc hàm gây ra. Tuy nhiên dạng BCNF có điều kiện quá khắt khe, đôi khi người ta chỉ cần chấp nhận ở dạng 3NF.

5.2. CHUẨN HÓA LƯỢC ĐỒ CƠ SỞ DỮ LIỆU

5.2.1. Phương pháp phân rã (Decomposition)

Dựa vào điều kiện phân rã bảo toàn thông tin: Q thành Q1 và Q2 thoả Q1 ∩ Q2 → Q1\Q2 hay Q1 ∩ Q2 → Q2\Q1.

Thuật toán phân rã thành các lược đồ ở dạng chuẩn BCNF như sau:

Cho Q và tập phụ thuộc hàm F xác định trên Q

Phân_rã = {Q} ;

done = false ;

Tính F^+;

while (not done)

if (có một Qi trong Phân_rã không ở dạng BCNF)

{

$X \rightarrow Y$ là phụ thuộc hàm không hiển nhiên trên Qi thoả:

$X \rightarrow Qi \notin F^+$ và $X \cap Y = \varnothing$ thì

Phân_rã = (Phân_rã – Qi) $\cup$ (XY) $\cup$ (Qi – Y)

}

else done = true;

Kết quả ta được tập Phân_rã gồm các lược đồ ở dạng **BCNF**

Ví dụ 5.13: Cho Q(ABCD), F = {AB $\rightarrow$ C, C$\rightarrow$ A, B $\rightarrow$ D}

Q(ABCD) không ở dạng **BCNF**, chọn C $\rightarrow$ A. Phân rã thành Q1 và Q2

Q1(AC), F1 = {C $\rightarrow$ A}, **BCNF**

Q2(BCD), F2 = {B $\rightarrow$ D}, không **BCNF**

Q2(BCD) không ở **BCNF**, chọn B $\rightarrow$ D. Phân rã thành Q21 và Q22

Q1(AC), F1 = {C $\rightarrow$ A}, **BCNF**

Q21(BD), F21 = {B $\rightarrow$ D}, **BCNF**

Q22(BC), F22 = $\varnothing$, **BCNF**

Cuối cùng ta phân rã :

Q1(AC), F1 = {C $\rightarrow$ A}, **BCNF**

Q2(BD), F2 = {B $\rightarrow$ D}, **BCNF**

Q3(BC), F3 = $\varnothing$, **BCNF**

Phân rã trên bảo toàn thông tin thành BCNF nhưng không đảm bảo lúc nào cũng bảo toàn phụ thuộc hàm (Ví dụ trên 3 tập phụ thuộc hàm cuối cùng F1, F2 và F3 không tương đương với tập phụ thuộc hàm F ban đầu).

5.2.2. Phương pháp tổng hợp (Synthesis)

Thuật toán sau cho phân rã đạt tối thiểu dạng 3NF.

Cho Q và tập F xác định trên Q

Tính Fc là một phủ tối tiểu của F;

Xác định các khoá của Q ;

i = 0 ;

for (Mỗi phụ thuộc hàm X → Y trong Fc)

if (không có Qj, j = 1,2,…i chứa XY)

{ i++ ;

Qi = XY ;

}

if (Không có Qj, j=1,2,…i chứa khoá của Q)

{ i++;

Qi = bất kỳ khoá nào của Q ;

}

return (Q1,Q2,…,Qi) ;

Ví dụ 5.14: Phân rã Q(ABCDE), F = {A → CD, C → D, B → E}

Fc = {A → C, C → D, B → E}

Suy ra:

Q1 = AC

Q2 = CD

Q3 = BE

Không có Qi chứa khoá nên ta có:

Q4 = AB

Vậy phân rã trên thành:

Q1(AC), F1 = {A → C}, đạt 3NF (đạt luôn BCNF)

Q2(CD), F2 = {C → D}, đạt 3NF (đạt luôn BCNF)

Q3(BE), F3 = {B → E}, đạt 3NF (đạt luôn BCNF)

Q4(AB), F2 = ∅, đạt 3NF (đạt luôn BCNF)

Thuật toán bảo toàn phụ thuộc hàm vì mỗi phụ thuộc hàm trong Fc cho một quan hệ và quan hệ này xác định luôn phụ thuộc hàm đó. Vậy F' = ∪Fi, F' ≡ Fc ≡ F

Thuật toán chắc chắn bảo toàn thông tin vì có ít nhất một lược đồ trong phân rã chứa khoá.

5.2.3. Cách thức chuẩn hoá trong thực tế

Trong thực tế khi chuẩn hoá lược đồ cơ sở dữ liệu người ta thường thực hiện theo các bước sau:

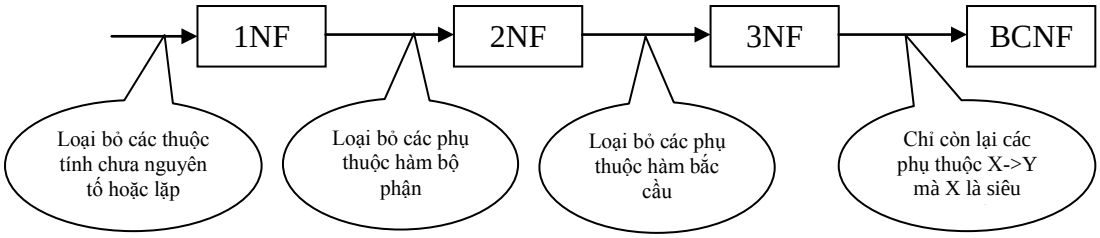
Bước 1: Kiểm tra xem quan hệ đã đạt đạt dạng chuẩn 1NF chưa?. Nếu chưa ở 1NF có nghĩa là có các thuộc tính chưa nguyên tố hoặc các thuộc tính lặp. Ta tiến hành tách các thuộc tính đó. Nhưng thông thường dạng chuẩn 1NF thường dễ đạt được.

Bước 2: Kiểm tra xem chúng có ở dạng 2NF không?, nghĩa là kiểm tra xem các thuộc tính không khoá có *phụ thuộc hoàn toàn* vào khoá chính không?. Nếu chưa đạt dạng chuẩn 2NF, có nghĩa là sẽ có những phụ thuộc hàm không hoàn toàn(bộ phận) vào khoá chính, thì ta tiến hành tách những phụ thuộc hàm bộ phận đó thành các bảng con để giảm sự trùng lặp thông tin.

Bước 3: Kiểm tra xem chúng đã đạt dạng chuẩn 3NF chưa?, nghĩa là các thuộc tính không khoá thì *phụ thuộc trực tiếp* vào khoá chính. Nếu chưa đạt 3NF có nghĩa là sẽ có những phụ thuộc hàm bắc cầu vào khoá chính. Ta tiến hành tách những phụ thuộc hàm bắc cầu đó thành bảng con.

Bước 4: kiểm tra xem chúng đã đạt dạng chuẩn BCNF chưa?, nghĩa là tất cả các phụ thuộc hàm đều có vế trái là siêu khoá. Nếu chưa đạt dạng chuẩn BCNF thì sẽ có những phụ thuộc hàm mà vế trái chưa phải là siêu khoá, khi đó ta tiến hành tách phụ thuộc hàm vi phạm đó.

Ta có thể tóm tắt lại quá trình chuẩn hoá như sau:



Chú ý: Trong thực tế người ta thường chỉ phân tích đạt được tới dạng chuẩn 3NF và thấp hơn, bởi càng dạng chuẩn cao thì thực hiện câu lệnh truy vấn sẽ càng lâu hơn. Nên có những ứng dụng người ta chấp nhận dạng chuẩn thấp để đổi lại là đơn giản trong cài đặt và hệ thống chạy nhanh hơn.

5.2.4. Ví dụ minh hoạ

Giả sử khi khảo sát thực tế về việc mua bán vật tư, ta có được lược đồ quan hệ sau

PHIEUNHAP(sophieu, ngaynhap, makhach,tenkh, diachikh, dienthoai, makho,

1 2 3 4 5 6 7

diachikho, hinhthucthanhtoan, loaitien, mavattu*, tenvattu*, soluong*,

8 9 10 11 12 13

donvitinh*, dongia*, tyleVAT*)

14 15 16

Có các phụ thuộc hàm sau:

F={1-> (2,3,4,5,6,7,8,9,10); 3->(4,5,6); 7->8; 11-> (12,14,15,16); (1,11)->13}

Bước1: Ta thấy khoá chính của quan hệ PHIEUNHAP là K=(1,11)=(sophieu, mavattu).

Ta thấy quan hệ trên chưa ở dạng 1NF vì có các thuộc tính (có dấu *) là thuộc tính lặp. Ta tiến hành phân rã thành hai quan hệ:

Quan hệ 1: Các thuộc tính lặp và phần khoá chính xác định chúng.

Quan hệ 2: các thuộc tính còn lại và phần khoá chính xác định phần này(các thuộc tính không lặp)

Quan hệ 1 gồm các thuộc tính lặp (11,12,13,14,15,16) và khoá chính là(1)

Tức là VATTU(Sophieu, mavattu, tenvattu, soluong, donvitinh,
1 11 12 13 14

dongia, tyleVAT)
15 16

Phụ thuộc hàm của quan hệ này là F1={ 11->(12,14,15,16); (1,11)->13}

Khoá chính của quan hệ này là {1,11}

Quan hệ 2 gồm các thuộc tính (2,3,4,5,6,7,8,9,10) và khoá (1)

PHIEUNHAP(sophieu, ngaynhap, makhach, tenkh, diachikh, dienthoai, makho,
1 2 3 4 5 6 7

diachikho, hinhthucthanhtoan, loaitien)
8 9 10

Phụ thuộc hàm của quan hệ này là F2={1->(2,3,4,5,6,7,8,9,10);3->(3,4,5); 7->8}

Khoá chính là {1}

Bước 2: Xét xem các quan hệ 1 và 2 ở trên đã đạt dạng chuẩn 2NF chưa?. Nếu chưa ta tiến hành tách đôi thành quan hệ 1: gồm các thuộc tính phụ thuộc vào một phần khóa chính và phần khoá chính; Quan hệ 2 : các thuộc tính còn lại và khoá chính.

- Xét quan hệ 1: Ta thấy chưa đạt dạng 2NF vì khoá là {1,11}, mà lại có phụ thuộc hàm 11->12,14,15,16, nghĩa là các thuộc tính 12,14,15,16 không phụ thuộc hoàn toàn vào khoá. Để đạt dạng chuẩn 2 ta tách thành 2 quan hệ:

Quan hệ 1_1: VATTU(mavattu, tenvattu, donvitinh, dongia, tyleVAT)
11 12 14 15 16

F1_1={11->12,14,15,16}, khoá là {11}

Quan hệ 1_2: DONGVATTU(sophieu, mavattu, soluong)
1 11 13

F1_2={(1,11)->13}

- Xét quan hệ 2: ta thấy quan hệ này đã đạt dạng chuẩn 2NF vì thuộc tính khoá của nó là sophieu và các thuộc tính đều phụ thuộc hoàn toàn vào khoá.

Bước 3: Xem xét chúng đã đạt dạng 3NF chưa?. Nếu chưa ta tiến hành tách đôi thành quan hệ 1: gồm các thuộc tính phụ thuộc bắc cầu và thuộc tính cầu; quan hệ 2: gồm các thuộc tính còn lại và thuộc tính cầu.

- Xét quan hệ 1_1:

Quan hệ 1_1: VATTU(mavattu, tenvattu, donvitinh, dongia, tyleVAT)
11 12 14 15 16

Đã là dạng chuẩn 3NF vì không có phụ thuộc hàm bắc cầu.

- Xét quan hệ 1_2:

Quan hệ 1_2: DONGVATTU(sophieu, mavattu, soluong)
1 11 13

Đã là dạng chuẩn 3NF vì không có phụ thuộc hàm bắc cầu.

- Xét quan hệ 2:

PHIEUNHAP(sophieu, ngaynhap, makhach,tenkh, diachikh, dienthoai, makho,
1 2 3 4 5 6 7

diachikho, hinhthucthanhtoan, loaitien
8 9 10

F2={1->(2,3,4,5,6,7,8,9,10);3->(4,5,6); 7->8}

Khoá chính là {1}

Ta thấy chưa đạt dạng 3NF vì có phụ thuộc hàm bắc cầu: Các thuộc tính (4,5,6) phụ thuộc hàm bắc cầu vào khoá chính qua cầu (3), còn thuộc tính (8) phụ thuộc bắc cầu vào khoá chính qua cầu (7). Vì vậy để có dạng chuẩn 3NF ta tách thành những quan hệ sau:

Quan hệ 2_1: KHACH(makhach,tenkh, diachikh, dienthoai)
3 4 5 6

F2_1={3->4,5,6}, khoá chính là (3)

Quan hệ 2_2: KHO(makho, diachikho)
7 8

F2_2={7->8}; khoá chính là (7)

Quan hệ 2_3: PHIEUNHAP(sophieu, ngaynhap, makhach, makho,
1 2 3 7

hinhthucthanhtoan, loaitien)
9 10

F2_3={1-(2,3,7,9,10)} ; khoá chính là (1)

Bước 4: Kiểm tra xem chúng đạt dạng BCNF chưa?.

Ta thấy tất cả các quan hệ trên đều đã đạt ở dạng BCNF vì tất cả các vế trái của các phụ thuộc hàm đều là siêu khoá.

Tóm lại: Ta có các quan hệ sau khi chuẩn hoá đạt dạng chuẩn BCNF như sau:

1 VATTU(mavattu, tenvattu, donvitinh, dongia, tyleVAT)

2 DONGVATTU(sophieu, mavattu, soluong)

3 KHACH(makhach,tenkh, diachikh, dienthoai)

4 KHO(makho, diachikho)

5 PHIEUNHAP(sophieu, ngaynhap, makhach, makho, hinhthucthanhtoan, loaitien).

BÀI TẬP CHƯƠNG 5

5.1: Xác định khoá và xét các phân rã sau đây theo hai tiêu chuẩn bảo toàn thông tin và bảo toàn phụ thuộc hàm.

a. Q(ABCD), F = {A → BC, C → D}

   Q1(AB), F1 = {A → B}

   Q2(CD), F2={C→D}

b. Q(ABCD), F={A→B, AC→D}

   Q1(AB), F1={A→B}

   Q2(ADC),F2={AC→D}

c. Q(ABDCE),F={A→C, B→C, C→D, DE→C, CE→A}

   Q1(AB),F1={A→D}

   Q2(CD),F2=∅

   Q3(AB),F3=∅

   Q4(CD),F4={C→D, DE→C, CE→D}

   Q5(AB), F5=∅

d. Q(ABCD), F={A→B, C→D}

   Q1(AB), F1={A→B}

   Q2(CD), F2={C→D}

5.2: Xét dạng chuẩn của các lược đồ quan hệ sau:

a. Q(ABCD), F={A→B, C→D}

b. Q(ABCD), F={AB→C, C→D}

c. Q(ABCD), F={AB→CD, CD→AB, C→B}

d. Q(ABCD), F={AB→CD, D→E, DE→ABC}

e. Q(ABCDEF), F={AB→E, AC→F, AD→B, B→C, C→D}

5.3: Cho lược đồ quan hệ Q(ABCDEF), F={C→F, E→A, CE→D, A→B}

a. Xác định khoá của Q

b. Phân rã thành dạng chuẩn Boyce-Codd bảo toàn thông tin

c. Phân rã thành dạng chuẩn 3 bảo toàn thông tin và bảo toàn phụ thuộc hàm.

5.4: Giả sử bài 3 được phân rã thành Q1(CDEF) và Q2(ABE), hãy xác định F1, F2 và đánh giá phân rã trên.

5.5: Nếu bài 3 được phân rã thành Q1(CF), Q2(AE), Q3(CDE), Q4(AB). Hãy xác định F1, F2, F3, F4 và đánh giá chúng.

5.6: Cho lược đồ quan hệ

VẬNCHUYỂN(TÀU, LOẠITÀU, CHUYẾN, HÀNG, CẢNG, NGÀY)

Mỗi tàu (TÀU) thuộc duy nhất một loại tàu nào đó (LOẠITÀU), mỗi chuyến có một mã số riêng biệt (CHUYẾN) dùng để xác định một chuyến tàu (TÀU) chở một khối lượng hàng hoá nào đó (HÀNG), mỗi chiếc tàu trong một ngày(NGÀY) chỉ cập vào một cảng duy nhất (CẢNG) của một chuyến vận chuyển nào đó (CHUYẾN)

a. Xác định tập các phụ thuộc hàm trên.

b. Xác định dạng chuẩn của VẬNCHUYỂN

c. Nếu VẬNCHUYỂN chưa tốt hãy tìm một phân rã tốt cho nó.

5.7: Cho quan hệ PHIEUNHAP(số phiếu, ngày, mã NCC, tên NCC, địa chỉ, Mã vật tư, tên vật tư, số lượng, đơn vị, đơn giá)

Có các phụ thuộc hàm

F={ Số phiếu->ngày, mã NCC; mã NCC->tên NCC, địa chỉ; mã vật tư->tên vật tư, đơn vị, đơn giá; số phiếu, mã vật tư -> số lượng}

Giả sử tách thành hai quan hệ:

PHIEUNHAP(số phiếu, ngày, mã NCC, Mã vật tư, tên vật tư, số lượng, đơn vị, đơn giá)

NHACUNGCAP(mã NCC, tên NCC, địa chỉ)

Kiểm tra xem chúng đạt dạng chuẩn nào? vì sao?

Giả sử tách thành các quan hệ:

PHIEUNHAP(số phiếu, ngày, mã NCC)

DONGPHIEU(số phiếu, Mã vật tư, số lượng)

NHACUNGCAP(mã NCC, tên NCC, địa chỉ)

VATTU(Mã vật tư, tên vật tư, đơn vị, đơn giá)

Kiểm tra xem chúng đạt dạng chuẩn nào? vì sao?

5.8. Giả sử có quan hệ BANHANG(Ngày tháng, mã hàng, tên hàng, đơn giá, số lượng, tổng tiền theo ngày, thanh toán)

Có các phụ thuộc hàm sau:

{mã hàng}->{tên hàng, đơn giá}; {ngày tháng}->{ tổng tiền theo ngày, thanh toán}

{ngày tháng, mã hàng}->{số lượng}

Hãy chuẩn hoá quan hệ BANHANG thành dạng chuẩn BCNF.

Chương 6
TỐI ƯU HOÁ CÂU HỎI

Các ngôn ngữ bậc cao nói chung và ngôn ngữ con dữ liệu nói riêng khi thực hiện đều mất rất nhiều thời gian. Do đó, trước khi thực hiện các câu lệnh thuộc các ngôn ngữ đó cần thiết phải biến đổi hợp lý về dạng tương đương, tức là dạng cho cùng một kết quả, để giảm thời gian tính toán. Việc làm đó được gọi là "tối ưu hóa" (*Optimiztation*). Chương này trình bày một vài phương pháp tối ưu hóa các biểu thức quan hệ. Sau đó sẽ trình bày chi tiết một phương pháp tối ưu hóa cho một lớp phổ cập các biểu thức quan hệ.

6.1. CÁC NGUYÊN TẮC TỔNG QUÁT ĐỂ TỐI ƯU HÓA CÂU HỎI

6.1.1. Các nguyên tắc tổng quan

Chúng ta hãy xét một ví dụ đơn giản sau đây:

Cho hai quan hệ R(AB) với *n* bản ghi và S(CD) với *m* bản ghi. Tích Đề-các của R và S là một quan hệ Q(ABCD) có *n * m* bản ghi. Chúng ta có câu hỏi "*Lấy giá trị của thuộc tính A sao cho B=C và D=50*". Câu hỏi được viết lại dưới dạng ngôn ngữ đại số quan hệ như sau:

$$\mathbf{(\,(}\ R(AB) \times S(CD)\ \mathbf{)}\ :\ (B=C \wedge D=50)\ \mathbf{)[}A\mathbf{]}$$

Nếu đưa phép chọn D =50 vào bên trong phép tích Đề-các sẽ được:

$$\mathbf{((}\ R(AB) \times \mathbf{(}\ S(CD)\ :\ (D = 50)\ \mathbf{)\ )}\ :\ (B=C)\ \mathbf{)[}A\mathbf{]}$$

và sau đó chuyển phép chọn B=C của tích Đề-các thành phép "kết nối bằng" chúng ta thu được: $\mathbf{(}R(AB) \overset{B=C}{\rhd\!\lhd} S(CD)\ :\ (D=50)\ \mathbf{)[}A\mathbf{]}$

Rõ ràng, phép tính cuối cùng sẽ đỡ tốn kém thời gian hơn rất nhiều.

Việc biến đổi câu hỏi thành câu hỏi tương đương như ví dụ nêu trên là một minh họa cho việc giảm bớt thời gian trả lời câu hỏi bằng cách giảm bớt số lần cần truy nhập tới bộ nhớ thứ cấp dựa trên nguyên tắc thực hiện phép chọn càng sớm càng tốt. Trình tự thực hiện các phép tính sẽ đóng một vai trò quan trọng quá trình tổ chức câu hỏi.

J. D. Ullman trong các kết quả nghiên cứu công bố lần đầu tiên của mình đã trình bày 6 chiến lược tổng quan cho việc tối ưu hóa câu hỏi như sau:

1. Thực hiện phép chọn càng sớm càng tốt.

Biến đổi câu hỏi để đưa phép chọn vào thực hiện trước nhằm làm giảm bớt kích cỡ của kết quả trung gian và do vậy chi phí phải trả cho việc truy nhập bộ nhớ thứ cấp cũng như lưu trữ của bộ nhớ chính sẻ nhỏ đi.

2. Tổ hợp những phép chọn xác định với phép tích Đề-các thành phép kết nối.

Như đã biết, phép kết nối, đặc biệt là phép kết nối bằng (*Equi Join*) có thể được thực hiện ít tốn kém hơn nhiều so với phép tích Đề-các trên cùng các quan hệ. Nếu kết quả của tích Đề-các RxS là đối số của phép chọn và phép chọn liên quan tới các phép so sánh giữa các thuộc tính của R và S thì rõ ràng phép tích Đề-các là phép kết nối.

3. Tổ hợp dãy các phép toán quan hệ một ngôi như các phép chọn và phép chiếu.

Một dãy các phép một ngôi như phép chọn hoặc phép chiếu mà kết quả của chúng phụ thuộc vào các bộ của một quan hệ độc lập thì có thể nhóm các phép đó lại.

4. Tìm các biểu thức con chung trong một biểu thức.

Nếu kết quả của một biểu thức con chung (tức là biểu thức xuất hiện nhiều hơn một lần) là một quan hệ không lớn và nó có thể được đọc từ bộ nhớ thứ cấp với ít thời gian thì nên tính toán trước biểu thức đó chỉ một lần. Nếu biểu thức con chung có liên quan tới một phép kết nối thì trong trường hợp tổng quát không thể thay đổi được nó bằng cách "đẩy" phép chọn vào trong.

Điều đáng quan tâm là, các biểu thức con chung có tần số xuất hiện lớn thường được biểu diễn trong các VIEW (khung nhìn) của người sử dụng, bởi vì, để thực hiện các câu hỏi đó cần phải thay thế nó bằng một biểu thức cố định cho VIEW.

5. Tiền xử lý các quan hệ / bảng (Table Preprocessing).

Có hai vấn đề quan trọng cần xử lý trước cho các quan hệ là sắp xếp trước các bộ giá trị theo thứ tự vật lý và sắp xếp lôgíc - tức là thiết lập các bảng chỉ mục (*Index*) cho các bản ghi. Khi đó việc thực hiện các phép toán có liên quan tới hai quan hệ (các phép toán hai ngôi) sẽ nhanh hơn rất nhiều.

6. Đánh giá trước khi thực hiện tính toán.

Mỗi khi cần chọn trình tự thực hiện các phép toán trong biểu thức, hoặc chọn một trong hai đối số của một phép hai ngôi, thì cần tính toán xem chí phí (*Cost*) thực hiện các phép tính đó (thường tính theo số phép toán, thời gian, hoặc/và dung lượng bộ nhớ cần thiết so với kích thước của các quan hệ, từ đó xác định được chi phí tổng thể phải trả cho các cách khác nhau khi thực hiện các câu hỏi.

Dựa vào các nguyên tắc nêu trên, chúng ta sẽ biến đổi câu truy vấn thành câu hỏi tương đương tối ưu hơn, để việc thực hiện có chi phí xử lý ít hơn. Nhưng trước khi có thể "tối ưu hóa" các biểu thức, cần làm rõ khái niệm khi nào thì hai biểu thức được gọi là tương đương.

6.1.2. Biểu thức tương đương và các quy tắc

- Biểu thức trong ngôn ngữ đại số quan hệ có các hạng thức là *biến quan hệ (relation variables)* R_1,..., R_n, các *quan hệ hằng (constant relation),* được xác định như là một ánh xạ từ các k-bộ của các quan hệ (r_1, ..., r_k) trong đó r_i là quan hệ trên lược đồ r_i và thay thế r_i vào R_i khi đánh giá biểu thức.

- Hai biểu thức E_1 và E_2 được gọi là *tương đương (Equivalent),* viết tắt là $E_1 \equiv E_2$, nếu chúng biểu diễn cùng một ánh xạ, nghĩa là, nếu chúng ta thay thế cùng các quan hệ cho tên các lược đồ tương ứng ở hai biểu thức E_1 và E_2, thì chúng sẽ cho ra cùng một kết quả.

6.1.2.1. Các quy tắc liên quan tới phép kết nối và tích Đề-các.

1) Quy tắc giao hoán của phép kết nối và tích Đề-các

Nếu E_1 và E_2 là hai biểu thức quan hệ và F là điều kiện trên các thuộc tính của E_1 và E_2 thì:

$E_1 \bowtie E_2 \equiv E_2 \bowtie E_1$ // Tính giao hoán của kết nối

$E_1 * E_2 \equiv E_1 * E_2$ // Tính giao hoán của kết bằng

$E_1 \times E_2 \equiv E_1 \times E_2$ // Tính giao hoán của tích Đề-các.

Chú ý: Nếu quan niệm quan hệ là tập các bộ (có thứ tự thuộc tính cố định) thì phép θ -kết, kết tự nhiên và tích Đề-các không thể giao hoán được vì thứ tự các thuộc tính trong quan hệ kết quả bị thay đổi.

2) Quy tắc kết hợp của phép kết nối và tích Đề-các.

Nếu E_1, E_2 và E_3 là các biểu thức quan hệ: F_1, F_2 là điều kiện thì:

$(E_1 \bowtie E_2) \bowtie E_3 \equiv E_1 \bowtie (E_2 \bowtie E_3)$

$(E_1 * E_2) * E_3 \equiv E_1 * (E_2 * E_3)$

$(E_1 * E_2) \times E_3 \equiv E_1 \times (E_2 \times E_3)$

Việc kiểm tra tính tương đương của các quy tắc trên khá dễ dàng.

6.1.2.2. Các quy tắc liên quan tới phép chọn và phép chiếu

Dãy các phép chiếu có thể tổ hợp lại thành một phép chiếu, biểu diễn theo các trường hợp sau:

3) Dãy các phép chiếu

(E[B1B2 ... Bm]**)[**A1A2 ... An**]** ≡ E**[**A1... An**]**

Ở đây, các thuộc tính A1...An phải nằm trong tập các thuộc tính B1…Bm. Ngữ nghĩa của việc biến đổi tương đương này là: Nếu thực hiện một phép chiếu biểu thức quan hệ E trên tập các thuộc tính B, rồi sau đó thực hiện tiếp phép chiếu trên tập con các thuộc tính $A \subset B$ của quan hệ vừa tìm được, thì kết quả của dãy phép chiếu này hoàn toàn tương đương với một phép chiếu biểu thức quan hệ E trên tập thuộc tính A.

Tương tự, dãy các phép chọn có thể tổ hợp thành một phép chọn để kiểm tra tất cả các điều kiện cùng một lúc và được biểu diễn như sau:

4) Dãy các phép chọn:

(((E : (f_1)) : f_2) : ... **)** : fn ≡ E : ($f_1 \wedge f_2 \ldots \wedge$ fn)

Ngữ nghĩa: Việc lần lượt thực hiện các phép chọn trên quan hệ kết quả của một phép chọn trước đó đối với biểu thức quan hệ E là tương đương với việc chọn trên E các bộ giá trị thỏa mãn đồng thời tất cả các điều kiện chọn f_1, f_2 ... fn.

5) Giao hoán phép chọn và phép chiếu:

(E[A1... An] : (f)**)** ≡ **(**E : (f)**)[**A1... An**]**

Một cách tổng quát hơn, nếu điều kiện chọn *f* liên quan tới các thuộc tính B1 ... Bm mà không nằm trong tập thuộc tính A1...An thì:

(E[A1 ... An]**)** : (f) ≡ **((**E[A1 ... An B1... Bm]**)** : (f)**)[**A1 ... An**]**

6) Giao hoán phép chọn và tích Đề-các:

Nếu tất cả các thuộc tính của F là thuộc tính của E_1 thì:

(E_1 x E_2**)** : (f) ≡ **(**E_1 : (f)**)** x E_2

Từ đó dễ dàng suy ra rằng, nếu F có dạng f = $f_1 \wedge f_2$ trong đó f_1 chỉ liên quan tới các thuộc tính của E_1; f_2 chỉ liên quan tới các thuộc tính của E_2, thì có thể sử dụng các luật 1, 4 và 6 để có:

(E_1 x E_2**)** : (f) ≡ **(**E_1 : (f_1)**)** x **(**E_2 : (f_2)**)**

Hơn nữa nếu f_1 chỉ liên quan tới các thuộc tính của E_1, nhưng f_2 liên quan tới các thuộc tính của cả E_1 và E_2 thì:

(E_1 x E_2**)** : (f) ≡ (**(**E_1 : (f_1)**)** x E_2) : (f_2)

7) Giao hoán phép chọn và một phép hợp:

Nếu có biểu thức E = $E_1 \cup E_2$ có thể giả thiết thêm rằng, các thuộc tính của E_1 và E_2 có cùng tên như của E hoặc ít nhất mỗi thuộc tính của E là phù hợp với một thuộc tính duy nhất của E_1 và một thuộc tính duy nhất của E_2. Khi đó:

($E_1 \cup E_2$) : (f) ≡ **(**E_1 : (f)**)** $\cup$ **(**E_2 : (f))

Nếu tên các thuộc tính của E_1 và hoặc E_2 khác với tên thuộc tính của E thì trong *f* ở vế phải của công thức trên cần thay đổi để sử dụng tên cho phù hợp.

8) Giao hoán phép chọn và một phép hiệu tập hợp

(E_1 - E_2**)** : (f) ≡ **(**E_1 : (f)**)** - **(**E_2 : (f)**)**

Như đã nêu trong luật 7, nếu tên các thuộc tính của E_1 và E_2 là khác nhau thì cần thay thế các thuộc tính trong *f* ở vế phải biểu thức tương đương tương ứng với E_1. Chú ý rằng, phép chọn **(**E_2 : (f)**)** có thể là không cần thiết. Trong nhiều trường hợp, việc thực hiện phép chọn (E_2 : (f)) trước sẽ có hiệu quả hơn là tính toán trực tiếp với E_2 vì kích cỡ quan hệ lúc đó sẽ bé đi rất nhiều.

Các quy tắc nêu trên nói chung là đẩy phép chọn xuống trước phép kết nối vì phép kết nối thường thực hiện lâu như phép tích Đề Các. Quy tắc đẩy phép chọn xuống trước phép kết nối suy ra từ quy tắc 4, 5 và 6. Quy tắc đẩy phép chiếu xuống trước phép tích Đề-các hoặc phép hợp cũng tương tự như quy tắc 6 và 7. Chú ý là không có phương pháp tổng quát cho việc đẩy phép chiếu xuống trước phép hiệu các tập hợp.

9) Giao hoán một phép chiếu với một phép tích Đề-các:

Gọi E_1, E_2 là hai biểu thức quan hệ, A1...An là tập các thuộc tính trong đó B1...Bm là các thuộc tính của E_1, các thuộc tính còn lại C1...Ck thuộc E_2. Khi đó:

$$(E_1 \times E_2)[A1 \ldots An] \equiv E_1[B1 \ldots Bm] \times E2[C1 \ldots Ck]$$

10) Giao hoán một phép chiếu với một phép hợp:

$$(E_1 \cup E_2)\ [A1 \ldots An] \equiv E_1[A1 \ldots An] \cup E_2[A1 \ldots An]$$

Như đã nêu trong luật 7, nếu tên các thuộc tính của E_1 và/hoặc E_2 là khác với các thuộc tính trong $E_1 \cup E_2$ thì cần thay A1 ... An bên vế phải bằng các tên phù hợp.

6.2. VÍ DỤ VỀ THUẬT TOÁN TỐI ƯU HÓA BIỂU THỨC QUAN HỆ

- Tới đây có thể áp dụng các quy tắc nêu trong mục trên để có thể tối ưu hóa các biểu thức quan hệ. Biểu thức "tối ưu" kết quả phải tuân theo các nguyên tắc đã nêu ở phần trên mặc dù rằng các nguyên tắc đó không có nghĩa là bảo đảm để tối ưu cho mọi trường hợp tương đương.

- Lưu ý rằng, luôn luôn đẩy phép chọn và phép chiếu xuống mức càng sâu càng tốt trong cây biểu diễn biểu thức quan hệ nhằm tạo nên một dãy các phép chọn cũng như phép chiếu để từ đó có thể tổ chức thành một phép chọn theo sau một phép chiếu. Nhóm các phép chọn và phép chiếu lại trong một nhóm để thực hiện trước các phép tính hai ngôi như phép hợp, tích Đề-các, hiệu tập hợp v.v...

- Có một số trường hợp đặc biệt xảy ra khi một phép tính hai ngôi có các hạng thức chứa phép chọn và/hoặc phép chiếu được áp dụng đối với lá của cây biểu diễn biểu thức. Khi đó cần xem xét cẩn thận tác động của phép tính hai ngôi vì một số trường hợp cần phải liên kết phép chọn hoặc phép chiếu với phép hai ngôi đó.

- Kết quả đầu ra (*Output*) của thuật toán là một chương trình bao gồm các bước như sau:

a. Áp dụng của một phép chọn hoặc một phép chiếu đơn giản.

b. Áp dụng của một phép chọn và một phép chiếu hoặc

c. Áp dụng của một tích Đề-các, phép hợp hoặc phép hiệu tập hợp cho hai hạng thức mà trước đó các phép chọn hoặc các phép chiếu đã được áp dụng cho một hoặc cả hai hạng thức.

Hãy xét một CSDL quản lý thư viện bao gồm các quan hệ sau đây:

1. **SACH** (Tensach, Tacgia, NhaXB, Masach): là quan hệ về các loại sách trong thư viện.

2. **NHAXUATBAN** (NhaXB, Diachi, Thanhpho): quan hệ về nhà xuất bản.

3. **DOCGIA** (TenDG, DchiDG, TphoDG, Sothe) : quan hệ về độc giả

4. **MUONSACH** (Sothe, Masach, Ngaymuon): quan hệ sổ theo dõi mượn.

Để lưu trữ thông tin về sách có thể giả thiết thêm rằng có một khung nhìn (VIEW) theo dõi các sách được mượn, TDMUON, bao gồm một số thông tin bổ sung về sách được mượn, là kết quả của kết nối tự nhiên của quan hệ SACH, DOCGIA và MUONSACH, chẳng hạn được xác định qua biểu thức quan hệ:

((SACH x DOCGIA x MUONSACH) : (f))[{S}]

Ở đây:

f = (DOCGIA.Sothe = MUONSACH.Sothe) ∧ (SACH.Masach = MUONSACH.Masach).

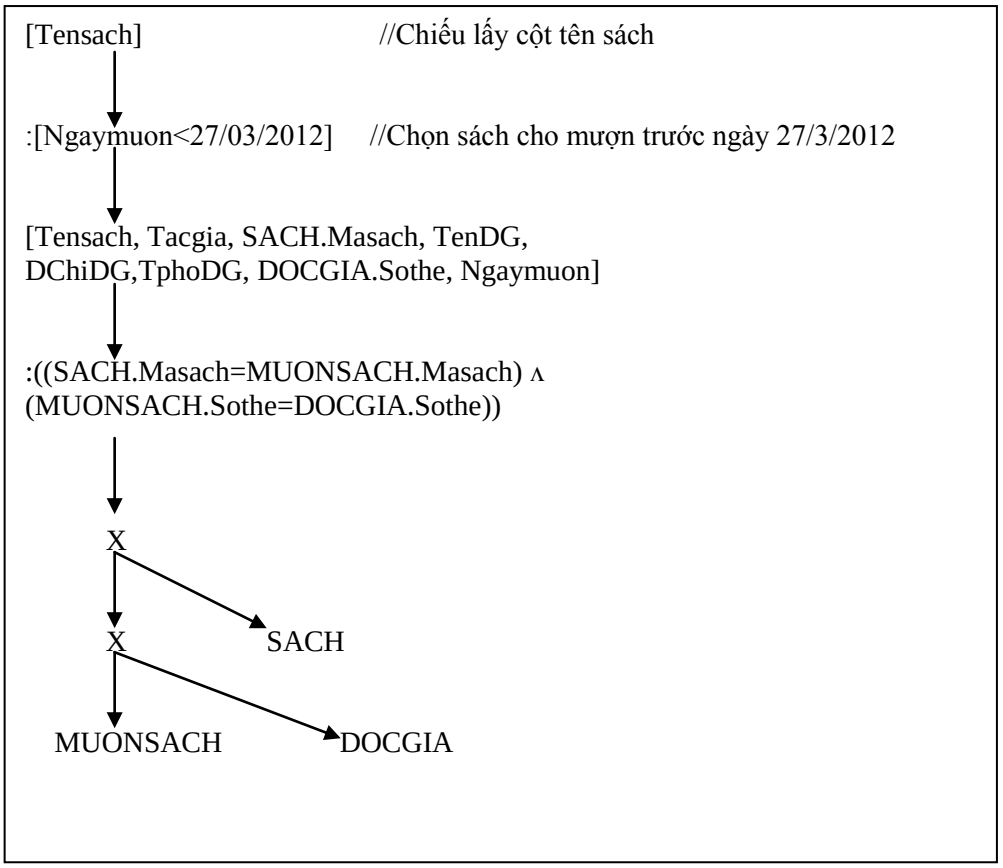
và S là tập các thuộc tính:

S ={Tensach, Tacgia, NhaXB, SACH.Masach, Tên_ĐG, DchiDG, TphoDG, DOCGIA.Sothe, Ngaymuon}.

Câu hỏi: Cho danh sách những cuốn sách đã cho mượn trước ngày 27/03/2012. Biểu thức quan hệ được viết như sau:

(TDMUON : (Ngaymuon < *27/03/2012*))[Tensach]

Hình cây của biểu thức trên được biểu diễn bằng hình 6.1. Xin lưu ý là, các phép toán nằm ở phía dưới là các phép toán được thực hiện trước các phép toán ở phía trên của cây.



Hình 6.1. Biểu diễn cây của biểu thức hỏi

Thay thế các giá trị *f* và S vào biểu thức hỏi có được cây biểu diễn của biểu thức quan hệ như trong hình 6.1

Bước thứ nhất của tối ưu là tách phép chọn *f* thành hai phép chọn với điều kiện:

SACH.Masach = MUONSACH.Masach

Và MUONSACH.Sothe = DOCGIA.Sothe

Bây giờ chúng ta có 3 phép chọn. Cần "đẩy" chúng xuống mức thấp hơn chừng nào còn có thể được.

Phép chọn với điều kiện Ngaymuon < *27/03/2012* được đẩy xuống dưới phép chiếu và hai phép chọn kia bằng cách áp dụng các quy tắc (hoặc luật) 4 và 5. Phép chọn đầu được áp dụng cho tích Đề-các ((MUONSACH x DOCGIA) x SACH). Vì thuộc tính **Ngaymuon** trong phép chọn chỉ có ở quan hệ MUONSACH nên có thể thay thế:

((MUONSACH x DOCGIA) x SACH) : (Ngaymuon < 27/0*3/2012)*

bằng biểu thức:

((MUONSACH x DOCGIA) : (Ngaymuon < *27/03/2012*)) x SACH)

và tiếp tục đẩy xuống nữa, cuối cùng ta được biểu thức:

(((MUONSACH : (Ngaymuon < *27/03/2012*)) x DOCGIA) x SACH)

Như vậy đã đẩy được phép chọn theo ngày mượn sách này xuống sâu như có thể.

Bây giờ tiếp tục đẩy phép chọn với điều kiện SACH.Masach = MUONSACH.Masach. xuống mức thấp nhất nếu có thể. Không thể đẩy phép chọn này xuống dưới tích Đề-các vì nó liên quan tới một thuộc tính của quan hệ SACH và một thuộc tính thuộc quan hệ MUONSACH. Do vậy phép chọn:

: MUONSACH.Sothe = DOCGIA.Sothe

có thể đẩy xuống để áp dụng cho tích Đề-các:

(MUONSACH x DOCGIA) : (: (Ngaymuon < *27/03/2012*)

Chú ý rằng MUONSACH.Sothe là tên một thuộc tính của phép chọn:

MUONSACH : (Ngaymuon < *27/03/2012*).

Bước tiếp theo: Tổ hợp hai phép chiếu thành một phép chiếu là [Tensach] nhờ luật 3. Kết quả được cho như trong hình 6.2. Sau đó áp dụng quy tắc mở rộng 5 thay thế:

:(MUONSACH.Sothe = DOCGIA.Sothe) và chiếu [Tensach]

nhờ dãy phép toán:

[Tensach, SACH.Masach, MUONSACH.Masach] (1)

:(SACH.Masach = MUONSACH.Masach) (2)

rồi chiếu để lấy tên sách: [Tensach] (3)

Áp dụng quy tắc 9 để thay thế biểu thức đầu tiên, biểu thức số (1), của phép chiếu nhờ phép chiếu:

[Tensach, SACH.Masach]

để áp dụng cho quan hệ SACH và [MUONSACH.Masach] áp dụng cho hạng thức phía trái của tích Đề-các trong hình 6.2.

Phép chiếu cuối và phép chọn có thể áp dụng quy tắc mở rộng 6 để có dãy:

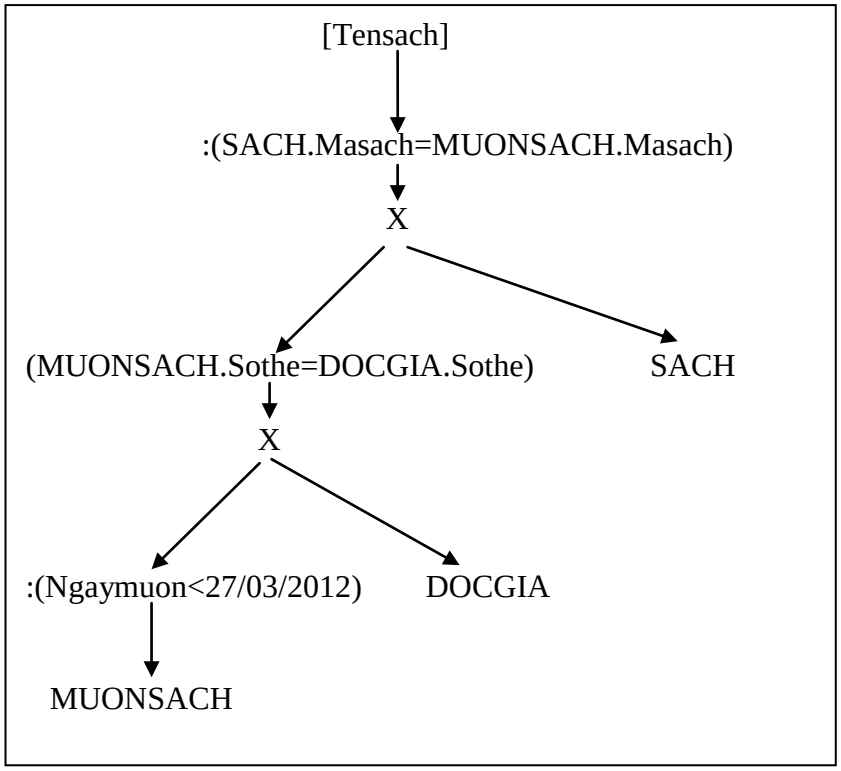
[MUONSACH.Masach, MUONSACH.Sothe, DOCGIA.Sothe] (5)

: (MUONSACH.Sothe = DOCGIA.Sothe) (6)

[MUONSACH.Masach]Masach] (7)

Phép chiếu đầu, số (5), được phân tách chuyển xuống tích Đề-các nhờ quy tắc 9.

Một phần phép chiếu DOCGIA.Sothe xuống hạng thức DOCGIA vì là thuộc tính của quan hệ này.



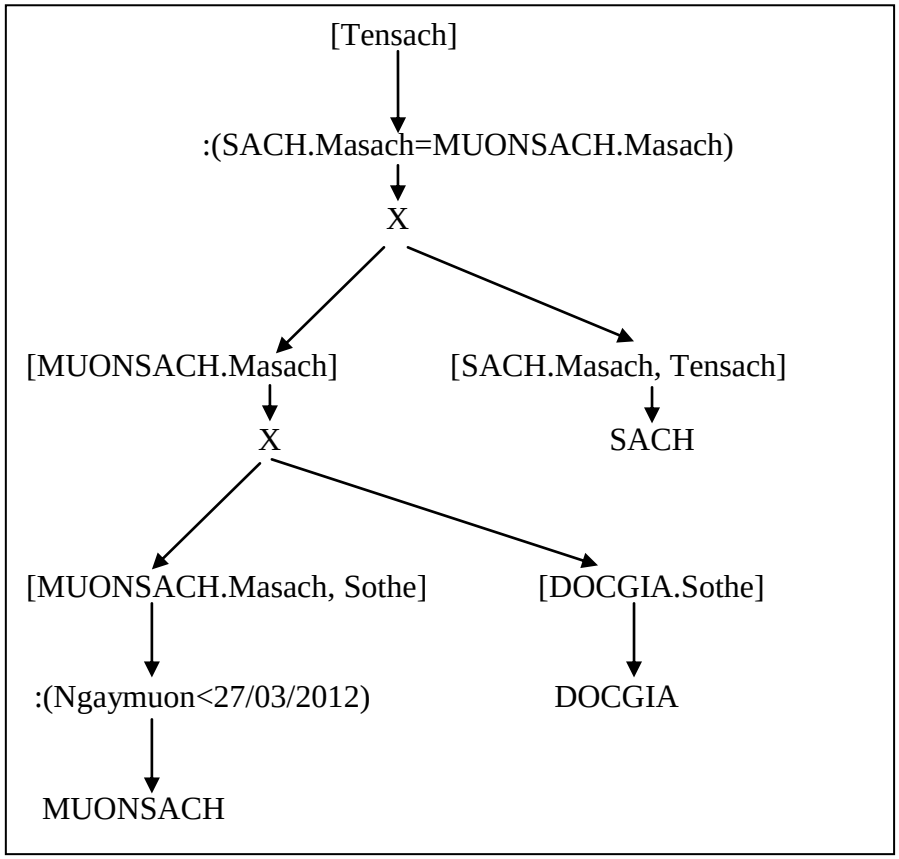
Hình 6.2. Cây với tổ hợp phép chọn và phép chiếu

Phần còn lại là phép chiếu để lấy 3 thuộc tính:

[MUONSACH.Masach, MUONSACH.Sothe, Ngaymuon]

được đẩy xuống hạng thức thứ MUONSACH. Các thuộc tính không phù hợp sẽ bị loại bỏ. Biểu diễn cây cuối cùng của biểu thức như trong hình 6.3.

Nhóm các phép tính bằng đường mũi tên gián đoạn. Mỗi tích Đề-các được tổ hợp với phép chọn để tạo thành một phép kết nối bằng nhau (*Equi Join*) rất có hiệu quả. Đặc biệt phép chọn trên quan hệ MUONSACH và phép chiếu của DOCGIA để lấy thuộc tính **Sothe** ở phía dưới là đủ để tổ hợp với tích Đề-các. Thứ tự thực hiện của cây biểu thức trong các hình 6.1, 6.2 và 6.3 là từ dưới lên: Nhóm các phép toán nằm ở phía dưới được thực hiện trước các phép toán ở phía trên.



Hình 6.3. Cây kết quả biểu diễn việc phân nhóm các biểu thức

6.3. THUẬT TOÁN TỐI ƯU HÓA CÂU HỎI

Ví dụ trên cho ta một minh họa về việc chuyển đổi một câu hỏi bằng ngôn ngữ Đại số quan hệ về dạng tương đương tốt hơn (hay tối ưu hơn). Phương pháp trên tập trung chủ yếu vào các phép chiếu, phép chọn và tích Đề-các, với mục đích làm sao "*đẩy*" được phép chọn và phép chiếu xuống mức thấp nhất, tức là thi hành các phép toán này càng sớm càng tốt, nếu có thể. Tiếp theo, kết hợp các phép chọn với tích Đề-các thành phép kết tự nhiên để làm giảm các kết quả trung gian. Cốt lõi của vấn đề tối ưu hóa chính là việc làm giảm thiểu lưu trữ trung gian và từ đó làm tăng nhanh tốc độ xử lý câu hỏi.

Tuy nhiên, để thực hiện được các quá trình tối ưu hóa như trên, chúng ta cần lưu ý tới thứ tự thực hiện các phép toán để có thể "*đẩy*" các phép toán xuống các mức hợp lý cần thiết. Bảng dưới đây cho phép chúng ta cách thực hiện các phép biến đổi tương đương đối với các phép Hội (*Union*), Trừ (*Minus*), Giao (*Intersect*), Tích Đề-các (*Cartesian*), Chia (*Division*), Chiếu (*Projection*) và Chọn (*Selection*).

(B1). Kết tự nhiên tương đương với dãy phép tích Đề-các, phép chọn và phép chiếu:

$$Q_1(AB) * Q_2(BC) \equiv (Q_1 \times Q_2 : (Q_1[B]=Q_2[B]))[ABC]$$

(*B2*). Phép *theta* kết tương đương với dãy phép tích Đề-các và phép chọn với điều kiện *theta*:

$$Q_1(AB)\ Q_2(CD) \equiv (Q_1 \times Q_2) : (B\ \theta\ D)$$

(*B3*). Phép giao (*Intersect*) tương đương với phần bù (*Complement*) của hội hai phần bù của 2 quan hệ:

$$Q_1 \cap Q_2 \equiv \neg((\neg Q_1) \cup (\neg Q_2)) \text{ và } (B4)$$

(*B4*). Phép bù của một quan hệ tương đương với tích Đề-các của các phép chiếu trên từng thuộc tính của quan hệ trừ đi các bộ giá trị đã có trong thể hiện của quan hệ:

$$\neg Q(X1\ldots Xn) \equiv (Q[X1] \times Q[X2] \times \ldots \times Q[Xn]) - Q(X1\ldots Xn)$$

(*B5*). Thương của 2 quan hệ tương đương với hiệu của các quan hệ trung gian sau:

$$Q_1(AB) \div Q_2(A) = Q_1[B] - ((Q_1[B] \times Q_2[A] - Q_1(AB))[B]$$

Áp dụng các cách biến đổi tương đương trên, kết hợp với các quy tắc "*đẩy*" và kết hợp như đã trình bày trong mục trên, chúng ta tìm hiểu thuật toán tổng quát để tối ưu hóa các câu hỏi trong ngôn ngữ đại số quan hệ.

Thuật toán:

Đầu vào (*Input*): Sơ đồ cú pháp câu hỏi bằng ngôn ngữ đại số quan hệ.

Đầu ra (*Output*): Sơ đồ cú pháp tối ưu.

Bước 1. Áp dụng các phép biến đổi tương đương nêu trong bảng (*B1*) đến (*B5*) trên.

Bước 2. Áp dụng luật 4 biến đổi dãy các phép chọn tương đương: tách phép chọn thành các phép chọn con.

Bước 3. Đối với mỗi phép chọn, áp dụng các luật 5 ,6 ,7 và 8 nhằm đẩy các phép toán chọn đó xuống càng sâu càng tốt.

Bước 4. Đối với mỗi phép chiếu, áp dụng các quy tắc 3 ,9 và 10 nhằm đẩy các phép toán chiếu đó xuống càng sâu càng tốt.

Bước 5. Tập trung các phép chọn nhằm áp dụng luật 4

Áp dụng luật 3 để loại bỏ bớt các phép chiếu vô ích.

Tập trung các phép chọn với tích Đề-các, nếu được, để chuyển thành phép kết tự nhiên hay *theta* kết bằng cách áp dụng các luật 3 và 6.

Nhận xét:

1. Thuật giải nêu trên chủ yếu nhằm giảm khối lượng dữ liệu trung gian chứ không chỉ ra thứ tự thực hiện các phép kết. Ví dụ:

$$\mathbf{(}Q_1(AB) * Q_2(BC)\mathbf{)} * Q_3(AC) \equiv$$

$$\mathbf{(}Q_1(AB) * Q_3(AC)\mathbf{)} * Q_2(BC)$$

2. Thuật giải này không cho chúng ta một kết quả tối ưu mà nó chỉ đưa ra một giải pháp tốt.

3. Các phép biến đổi chỉ dựa trên các phép toán cơ bản là Hội (*Union*), Trừ (*Minus*), Giao (*Intersect*), Tích Đề-các (*Cartesian*), Chia (*Division*), Chiếu (*Projection*) và Chọn (*Selection*) mà chúng ta còn có thể thực hiện các phép biến đổi dựa trên các phép toán khác nữa.

BÀI TẬP CHƯƠNG 6

6.1. Cho các quan hệ Q(ABD), R(BDF), S(FG) tối ưu biểu thức đại số sau:

a) $\sigma_{D=d}(\Pi_{BDF}(Q*R) - \Pi_{BDF}(R*S))$

b) $\Pi_{B}(\sigma_{A=a}(\sigma_{D=d}(Q)*(R - \Pi_{BDF}(R*S))))$

6.2. Cho cơ sở dữ liệu gồm các quan hệ Q(AB), R(BC), S(AC) tối ưu biểu thức đại số sau:

a) $\Pi_{B}(Q*R*S)$

b) $\Pi_{AC}(Q*R)$

c) $\Pi_{AC}(Q*R*S)$

d) $\Pi_{AC}(\sigma_{A=c1}(Q)* \sigma_{A=c2}(R)*S)$

6.3. Cho các quan hệ Q(AB), R(BC), S(AC) tối ưu biểu thức đại số sau:

a) $E1 = \sigma_{B<=C \wedge C=4 \wedge D<A}(Q*R*S)$

b) $E2 = \Pi_{ABC}(Q*\sigma_{D=d}(R*S))$

TÀI LIỆU THAM KHẢO

1. Nguyễn Kim Anh, *Nguyên lý của các hệ cơ sở dữ liệu*, NXB ĐHQGHN, 2009.
2. TS. Lê Văn Phùng, *Bài giảng cơ sở dữ liệu*, NXB lao động- xã hội, 2004.
3. Vũ Đức Thi, *Cơ sở dữ liệu kiến trúc và thực hành*, NXB thống kê, Hà nội 1997.
4. Đỗ Trung Tuấn, *Cơ sở dữ liệu (DataBase)*, NXB Giáo dục Hà nội, 1998.
5. TS. Trần Văn Tư, *Microsoft SQL Server 7.0*, NXB Thống kê 2000.
6. Trần Thành Trai, *Nhập môn Cơ sở dữ liệu,* NXB Giáo dục, TP.Hồ Chí Minh 1996.
7. Lê Tiến Vương. *Nhập môn cơ sở dữ liệu*, NXB Thống kê Hà nội, 2000. Tái bản lần 5.
8. Nhóm tác giả trường ĐHQG TPHCM, *Bài giảng cơ sở dữ liệu*
9. *Introdution to Oracle SQL and PL/SQL Using Procedure Builder*. Vol 1,2,3,4. ORACLE 7.3. 1996
10. Raghu Ramakrishnan, *Database Management Systems*, McGraw – Hill internationa Editions.
11. Elmasri , Navathe, *Fundamentals of Database Systems*, Pearson Education
12. Peter Rob, Carlos Coronel, *Database Systems*, Wadworth Publishing Company.

MỤC LỤC