HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CƠ SỞ DỮ LIỆU PHÂN TÁN

(Dùng cho sinh viên hệ đào tạo đại học từ xa)

Lưu hành nội bộ

HÀ NỘI - 2009

CƠ SỞ DỮ LIỆU PHÂN TÁN

Biên soạn : TS. PHẠM THẾ QUẾ

LỜI NÓI ĐẦU

Tài liệu "Cơ sở dữ liệu phân tán" là sách hướng dẫn học tập dùng cho sinh viên hệ đào tạo từ xa ngành công nghệ thông tin và ngành kỹ thuật điện tử, viễn thông. Nội dung của tài liệu bao gồm:

- Chương I giới thiệu khái niệm cơ bản về cơ sở dữ liệu phân tán, xử lý phân tán vàhệ thống xử lý phân tán. Sự cần thiết của hệ cơ sở dữ liệu phân tán và các đặc điểm của cơ sở dữ liệu phân tán. Cấu trúc logic của cơ sở dữ liệu phân tán và các lợi íchphân tán dữ liệu trên mạng.

- Chương II giới thiệu tổng quát về hệ quản trị cơ sở dữ liệu phân tán.Ưu điểm cách tiếp cận mô hình cơ sở dữ liệu quan hệ và hệquản trị cơ sở dữ liệu quan hệ. Vấn đề quy tắc toàn vẹn dữ liệu. Mô hình kiến trúc hệ quản trị cơ sở dữ liệu phân tán và kiến trúc tổng quan của một hệ quản trị phức hệ CSDL phân tán

- Chương III trình bày những vấn đề thiết kế cơ sở dữ liệu phân tán,là các vấn đề phân mảnh dữ liệu. Sự cần thiết phải phân mảnh, các kiểu phân mảnh, mức độ phân mảnh, các quy tắc phân mảnh và bài toán cấp phát dữ liệu. Nội dung của chương trình bày tổng quát kỹ thuật phân mảnh ngang cơ sở và phân mảnh ngang dẫn xuất. Thông tin cần thiết của phân mảnh ngang. Phương pháp phân mảnh dọc, thông tin cần thiết của phân mảnh dọc và các thuật toán tụ nhóm và phân mảnh. Có nhiều bài toán cần thiết phải sử dụng lai ghép phân mảnh ngang và phân mảnh dọc. Bài toán cấp phát dữ liệu, thông tin cần thiết cho bài toán cấp phát và mô hình cấp phát.

- Chương IV giới thiệu kiểm soát dữ liệu ngữ nghĩa, là quá trình kiểm soát khung nhìn trong các hệ quản trịcơ sử dữ liệu tập trung và khung nhìn trong các hệ quản trị cơ sở dữ liệu phân tán. Nội dung kiểm soát dữ liệu ngữ nghĩa cũng bao hàm vấn đề an toàn dữ liệu. Kiểm soát cấp quyền tập trung và kiểm soát cấp quyền phân tán.Kiểm soát toàn vẹn ngữ nghĩa tập trung và kiểm soát toàn vẹn ngữ nghĩa phân tán.

- Chương V đề cập đến các vấn đề xử lý truy vấn trong các hệ cơ sở dữ liệu phân tán. Khái niệm xử lý truy vấn,mục đích của việc xử lý truy vấn và giới thiệu các tầng của quá trình xử lý truy vấn.

Tài liệu"Cơ sở dữ liệu phân tán" không chỉ đề cập đến những vấn đề cơ sở lý thuyết mà còn trình bày một số kỹ năng cần thiết để thiết kế và cài đặt các hệ cơ sở dữ liệu cụ thể. Hy vọng sẽ có ích cho sinh viên và những người muốn xây dựng các hệ thống tin học ứng dụng. Tài liệu có thể còn nhiều thiếu sót trong biên soạn, tôi vẫn mạnh dạn giới thiệu tài liệu này và mong nhận được sự góp ý của bạn đọc.

Tác giả

CHƯƠNG 1: KHÁI NIỆM CƠ BẢN VỀ CƠ SỞ DỮ LIỆU PHÂN TÁN

Trong chương này trình bày những khái niệm cơ bản về nguyên lý các hệ cơ sở dữ liệu phân tán, bao gồm các nội dung sau

- Xử lý dữ liệu phân tán.
- Hệ cơ sở dữ liệu phân tán là gì.
- Khả năng của các hệ cơ sở dữ liệu phân tán.
- Các mô hình xử lý dữ liệu phân tán
- Tổng quan về hệ quản trị cơ sở dữ liệu quan hệ.

1.1 MỞ ĐẦU

Nguyên lý các hệ cơ sở dữ liệu phân tán được xây dựng dựa trên sự hợp nhất của hai hướng tiếp cận đối với quá trình xử lý dữ liệu, đó là lý thuyết các hệ cơ sở dữ liệu và công nghệ mạng máy tính.

Một trong những động lực thúc đẩy sự phát triển nhanh việc sử dụng các hệ CSDL là nhu cầu tích hợp các loại dữ liệu, cung cấp đa dạng các loại hình dịch vụ và các dịch vụ đa phương tiện cho người sử dụng. Mặt khác, kết nối máy tính thành mạng với mục tiêu chia sẻ tài nguyên, khai thác có hiệu quả các tài nguyên thông tin, nâng cao khả năng tích hợp và trao đổi các loại dữ liệu giữa các thành phần trên mạng.

Nhu cầu thu thập, lưu trữ. xử lý và trao đổi thông tin bgày càng tăng, các hệ thống xử lý tập trung đã bộc lộ những nhược điểm sau :

- Tăng khả năng lưu trữ thông tin là khó khăn, bởi bị giới hạn tối đa của thiết bị nhớ
- Độ sẵn sàng phục vụ của CSDL không cao khi số người sử dụng tăng
- Khả năng tính toán của các máy tính đơn lẻ đang dần tới giới hạn vật lý.
- Mô hình tổ chức lưu trữ, xử lý dữ liệu tập trung không phù hợp cho những tổ chức kinh tế, xã hội có hoạt động rộng lớn, đa quốc gia

Những nhược điểm này đã được khắc phục khá nhiều trong hệ thống phân tán. Những sản phẩm của các hệ thống phân tán đã xuất hiện nhiều trên thị trường và từng bước chứng minh tính ưu việt của nó hơn hẳn các hệ thống tập trung truyền thống. Các hệ thống phân tán sẽ thay thế dần các hệ thống tập trung.

1.2 XỬ LÝ PHÂN TÁN VÀ HỆ THỐNG XỬ LÝ PHÂN TÁN

1.2.1 Khái niệm xử lý phân tán

Thuật ngữ xử lý phân tán có thể là thuật ngữ được lạm dụng nhiều nhất trong khoa học máy tính trong những năm vừa qua. Nó thường được dùng để chỉ những hệ thống gồm nhiều

loại thiết bị khác nhau chẳng hạn như: hệ đa bộ xử lý, xử lý dữ liệu phân tán, mạng máy tính ....

Có hai khái niệm xử lý phân tán liên quan với nhau.

- Khái niệm liên quan đến việc tính toán trên Client/Server. Trong đó ứng dụng được chia ra thành hai phần, phần của Server và phần của Client và được vận hành ở hai nơi. Trong tính toán phân tán này cho phép truy nhập trực tiếp dữ liệu và xử lý dữ liệu trên Server và Client.
- Khái niệm thứ hai là việc thực hiện các tác vụ xử lý phức tạp trên nhiều hệ thống. Không gian nhớ và bộ xử lý của nhiều máy cùng hoạt động chia nhau tác vụ xử lý. Máy trung tâm sẽ giám sát và quản lý các tiến trình này. Có trường hợp thông qua Internet, hàng nghìn máy cùng xử lý một tác vụ.

Có thể định nghĩa hệ xử lý phân tán như sau: Hệ xử lý phân tán là một tập hợp các phần tử xử lý tự trị (không nhất thiết đồng nhất) được kết nối với nhau bởi một mạng máy tính và cùng phối hợp thực hiện những công việc gán cho chúng. Phần tử xử lý ở đây để chỉ một thiết bị tính toán có khả năng thực hiện chương trình trên nó.

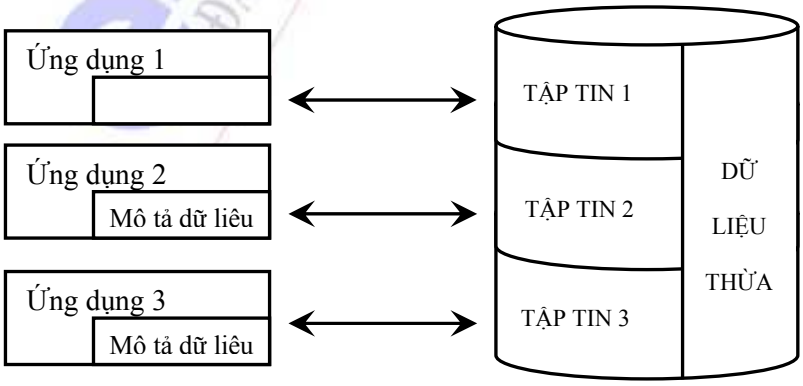
1.2.2 Hệ thống phân tán

Hệ thống phân tán là tập hợp các máy tính độc lập kết nối với nhau thành một mạng máy tính được cài đặt các hệ cơ sở dữ liệu và các phần mềm hệ thống phân tán tạo khả năng cho nhiều người sử dụng truy nhập chia sẻ nguồn thông tin chung. Các máy tính trong hệ thống phân tán có kết nối phần cứng lỏng lẻo, có nghĩa là không chia sẻ bộ nhớ, chỉ có một hệ điều hành trong toàn bộ hệ thống phân tán

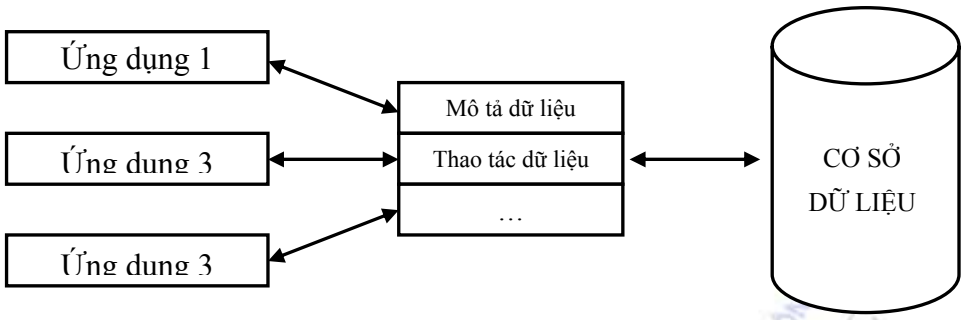
Các mạng máy tính được xây dựng dựa trên kỹ thuật Web, ví dụ như mạng Internet, mạng Intranet… là các mạng phân tán.

1.3 HỆ CƠ SỞ DỮ LIỆU PHÂN TÁN LÀ GÌ.

Công nghệ các hệ cơ sở dữ liệu phát triển từ mô hình xử lý dữ liệu, trong đó mỗi ứng dụng định nghĩa một hay nhiều tệp dữ liệu riêng của nó (hình 1.1), sang mô hình định nghĩa và quản lý dữ liệu tập trung. Dẫn đến khái niệm độc lập dữ liệu, nghĩa là tính bất biến của các hệ ứng dụng đối với sự thay đổi cấu trúc lưu trữ và các chiến lược truy nhập dữ liệu.



Hình 1.1: Xử lý dữ liệu truyền thống



Hình 1.2: Xử lý cơ sở dữ liệu

Trong ngữ cảnh hệ xử lý phân tán thì hệ cơ sở dữ liệu phân tán có thể được xem như những công cụ làm cho quá trình xử lý dữ liệu phân tán dễ dàng hơn và hiệu quả hơn. Khái niệm hệ cơ sở dữ liệu phân tán ở đây bao gồm cả khái niệm cơ sở dữ liệu phân tán và hệ quản trị cơ sở dữ liệu phân tán.

Cơ sở dữ liệu phân tán là một tập các cơ sở dữ liệu có quan hệ với nhau về mặt logic và được phân bố trên một mạng máy tính. Hệ quản trị cơ sở dữ liệu phân tán là hệ thống phần mềm cho phép quản trị cơ sở dữ liệu phân tán và làm cho sự phân tán đó là trong suốt đối với người sử dụng.

Trong mô hình cơ sở dữ liệu phân tán bản thân cơ sở dữ liệu có ở trên nhiều máy tính khác nhau. Như vậy, đặc trưng của cơ sở dữ liệu phân tán là các CSDL được phân bố trên mạng máy tính và có quan hệ với nhau về mặt logic.

Hệ CSDL phân tán không đơn thuần bao gồm nhiều file dữ liệu được tổ chức lưu trữ riêng lẻ trên các thiết bị nhớ của mạng máy tính. Để tạo một hệ CSDL phân tán, các file không chỉ có quan hệ với nhau về mặt logic mà còn cần có một cấu trúc giao diện chung giữa chúng để các file có thể truy nhập lẫn nhau.

Có rất nhiều ứng dụng yêu cầu các hệ quản trị CSDL thao tác trên dữ liệu bán cấu trúc hoặc không cấu trúc, như các file Web trên mạng Internet.

1.4 SỰ CẦN THIẾT CỦA HỆ CƠ SỞ DỮ LIỆU PHÂN TÁN

Trong những năm gần đây, công nghệ cơ sở dữ liệu phân tán đã trở thành một lĩnh vực quan trọng của công nghệ thông tin, tính cần thiết của nó ngày càng được nâng cao. Có nhiều nguyên nhân thúc đẩy sự phát triển của các hệ CSDLPT:

1.4.1 Sự phát triển của các cơ cấu tổ chức

Cùng với sự phát triển cuả xã hội, nhiều cơ quan, xí nghiệp có cơ cấu tổ chức không tập trung, hoạt động phân tán trên phạm vi rộng. Vì vậy thiết kế và cài đặt cơ sở dữ liệu phân tán là phù hợp, đáp ứng mọi nhu cầu truy xuất và khai thác dữ liệu Cùng với sự phát triển của công nghệ viễn thông, tin học, động cơ thúc đẩy kinh tế, việc tổ chức các trung tâm máy tính lớn và tập trung trở thành vấn đề cần nghiên cứu.

Cơ cấu tổ chức và vấn đề kinh tế là một trong những nguyên nhân quan trọng nhất của sự phát triển cơ sở dữ liệu phân tán.

1.4.2 Giảm chi phí truyền thông

Trong thực tế, sử dụng một số ứng dụng mang tính địa phương sẽ làm giảm chi phí truyền thông. Bởi vậy, việc tối ưu hoá tính địa phương của các ứng dụng là một trong những mục tiêu chính của việc thiết kế và cài đặt một CSDLPT.

1.4.3 Hiệu quả công việc

Sự tồn tại một số hệ thống xử lý địa phương đạt được thông quan việc xử lý song song. Vấn đề này có thể thích hợp với mọi hệ đa xử lý. CSDLPT có thuận lợi trong phân tích dữ liệu phản ánh điều kiện phụ thuộc của các ứng dụng, cực đại hoá tính địa phương của ứng dụng. Theo cách này tác động qua lại giữa các bộ xử lý được làm cực tiểu. Công việc được phân chia giữa các bộ xử lý khác nhau và tránh được các tắc nghẽn thông tin trên mạng truyền thông hoặc các dịch vụ chung của toàn hệ thống. Sự phân tán dữ liệu phản ánh hiệu quả làm tăng tính địa phương của các ứng dụng.

1.4.4 Độ tin cậy và tính sẵn sàng

Cách tiếp cận CSDLPT, cho phép truy nhập độ tin cậy và tính sẵn sàng cao hơn. Tuy nhiên, để đạt được mục đích đó là vấn đề không đơn giản đòi hỏi kỹ thuật phức tạp. Những lỗi xuất hiện trong một CSDLPT có thể xảy ra nhiều hơn vì số các thành phần cấu thành lớn hơn, nhưng ảnh hưởng của lỗi chỉ ảnh hưởng tới các ứng dụng sử dụng các site lỗi. Sự hỏng hóc của toàn hệ thống hiếm khi xảy ra.

CSDLPT là sự tập hợp các dữ liệu thuộc cùng một hệ thống về mặt logic nhưng phân bố trên các site của mạng máy tính. Công nghệ CSDLPT là sự kết hợp giữa hai vấn đề phân tán và hợp nhất:

- Phân tán : phân tán dữ liệu trên các site của mạng
- Hợp nhất : hợp nhất về mặt logic các dữ liệu phân tán sao cho chúng xuất hiện với người sử dụng giống như với CSDL đơn lẻ duy nhất.

Công nghệ CSDL phân tán mới thực sự phát triển trong những năm gần đây nhờ sự phát triển của kỹ thuật tính toán, kỹ thuật truyền thông và mạng máy tính. Những ứng dụng được xây dựng trên CSDL phân tán đã xuất hiện nhiều trên thị trường và từng bước chứng minh tính ưu việt của nó so với CSDL tập trung.

1.5 CÁC ĐẶC ĐIỂM CỦA CƠ SỞ DỮ LIỆU PHÂN TÁN

Cơ sở dữ liệu phn tán không đơn giản là sự phân bố của các cơ sở dữ liệu, bởi vì cơ sở dữ liệu phân tán có nhiều đặc điểm khác biệt so với cơ sở dữ liệu tập trung truyền thống. Phần này so sánh cơ sở dữ liệu phân tán với cơ sở dữ liệu tập trung ở một số đặc điểm: điều khiển tập trung, sự độc lập dữ liệu, sự giảm dư thừa dữ liệu, các cấu trúc vật lý phức tạp để truy xuất hiệu quả.

1.5.1 Điều khiển tập trung

Điều khiển tập trung (Centralized Control) là một đặc điểm của cơ sở dữ liệu tập trung, toàn bộ dữ liệu được tập trung lại nhằm để tránh sự dư thừa dữ liệu, đảm bảo được tính độc

lập của dữ liệu. Dữ liệu được quản lý tập trung bởi người quản trị cơ sở dữ liệu. Chức năng cơ bản của người quản trị cơ sở dữ liệu (DBA - Database Administrator) là bảo đảm sự an toàn của dữ liệu. Trong các cơ sở dữ liệu phân tán vấn đề điều khiển tập trung không được nhấn mạnh. Nói chung, trong các cơ sở dữ liệu phân tán , sự điều khiển được thực hiện theo một cấu trúc điều khiển phân cấp bao gồm hai loại người quản trị cơ sở dữ liệu:

- Người quản trị cơ sở dữ liệu toàn cục (Global Database Administrator) là người có trách nhiệm chính về toàn bộ cơ sở dữ liệu phân tán..
- Người quản trị cơ sở dữ liệu cục bộ (Local Database Administrator) là người có trách nhiệm về cơ sở dữ liệu cục bộ của họ.

Tuy nhiên, những người quản trị cơ sở dữ liệu cục bộ cần phải có những quyền độc lập riêng về cơ sở dữ liệu cục bộ của mình mà người quản trị cơ sở dữ liệu toàn cục hoàn toàn không có những quyền này và sự phối hợp giữa các vị trí được thực hiện bởi chính những người quản trị cục bộ. Đặc điểm này được gọi là sự độc lập vị trí. Các cơ sở dữ liệu phân tán có thể khác nhau rất nhiều về mức độ độc lập vị trí. Từ sự độc lập vị trí hoàn toàn (không có người quản trị cơ sở dữ liệu tập trung) đến sự điều khiển tập trung hoàn toàn.

1.5.2 Độc lập dữ liệu

Độc lập dữ liệu (Data Independence) là một đặc điểm của cơ sở dữ liệu. Độc lập dữ liệu có nghĩa là tổ chức lưu trữ dữ liệu là trong suốt đối với người lập trình ứng dụng. Ưu điểm của độc lập dữ liệu là các chương trình không bị ảnh hưởng bởi những thay đổi về tổ chức lưu trữ vật lý của dữ liệu.

Trong các hệ cơ sở dữ liệu phân tán, độc lập dữ liệu cũng quan trọng như trong các cơ sở dữ liệu tập trung. Tuy nhiên, một đặc điểm mới được đưa vào trong khái niệm thông thường của độc lập dữ liệu là sự trong suốt phân tán (Distribution Transparency). Nhờ sự trong suốt phân tán mà các chương trình ứng dụng có thể được viết giống như trong cơ sở dữ liệu không được phân tán. Vì vậy, tính đúng đắn của các chương trình ứng dụng không bị ảnh hưởng bởi sự di chuyển dữ liệu từ một vị trí này đến một vị trí khác. Tuy nhiên, tốc độ thực hiện của các chương trình ứng dụng thì bị ảnh hưởng.

Độc lập dữ liệu trong cơ sở dữ liệu tập trung được thể hiện thông qua một kiến trúc nhiều mức, các mức này có những mô tả khác nhau về dữ liệu và những ánh xạ biến đổi giữa các mức. Sự trong suốt phân tán trong cơ sở dữ liệu phân tán được thê hiện bằng cách bổ sung thêm các mức trong suốt vào kiến trúc nhiều mức của cơ sở dữ liệu tập trung.

1.5.3 Giảm dư thừa dữ liệu

Trong các cơ sở dữ liệu tập trung, sự dư thừa dữ liệu được giảm thiểu, vì tránh sự không nhất quán giữa nhiều bản sao bằng cách chỉ có một bản sao và tiết kiệm vùng nhớ lưu trữ. Các ứng dụng chia sẻ chung, truy xuất đến các tập tin dữ liệu.

Tuy nhiên, trong các cơ sở dữ liệu phân tán, sự dư thừa dữ liệu là một đặc điểm cần thiết, vì các lý do sau:

- Làm tăng tính cục bộ của các ứng dụng nếu dữ liệu được nhân bản tại tất cả các vị trí mà ứng dụng cần dữ liệu này. Khi đó, các ứng dụng cục bộ được thực hiện nhanh hơn vì không cần phải truy xuất dữ liệu từ xa.
- Làm tăng tính sẵn sàng của hệ thống ứng dụng, vì một vị trí có sự cố sẽ không làm ngưng sự thực hiện của các ứng dụng ở những vị trí khác nếu dữ liệu tại vị trí bị hỏng được nhân bản tại các vị trí khác.

Tuy nhiên, sự nhân bản dữ liệu cần phải xem xét kỹ lưỡng dựa vào hai loại ứng dụng cơ bản, đó là ứng dụng chỉ đọc và ứng dụng cập nhật. Sự nhân bản dữ liệu giúp cho các ứng dụng chỉ đọc được thực hiện nhanh hơn, nhưng nó làm cho các ứng dụng cập bị thực hiện lâu hơn vì phải cập nhật dữ liệu tại các vị trí được nhân bản.

Như vậy, sự nhân bản dữ liệu sẽ là một ưu điểm nếu hệ thống có rất nhiều ứng dụng chỉ đọc và có rất ít ứng dụng cập nhật. Trong trường hợp ngược lại thì sự nhân bản dữ liệu lại là một nhược điểm.

1.5.4 Độ tin cậy qua các giao dịch phân tán

Hệ quản trị CSDL phân tán cải thiện độ tin cậy qua các giao dịch phân tán, vì các thành phần được nhân bản hạn chế được các vị trí lỗi riêng lẻ. Lỗi của trạm riêng, hoặc lỗi của truyền thông làm cho một hoặc nhiều trạm mất liên lạc, không đủ để phá vỡ toàn bộ hệ thống. Trong trường hợp CSDL phân tán, điều này nghĩa là một số dữ liệu không thể truy nhập được, nhưng nếu biết cách hỗ trợ cho các giao dịch phân tán và các giao thức ứng dụng, thì người sử dụng vẫn có thể truy nhập được tới phần khác trong CSDL phân tán.

Giao dịch là một đơn vị tính toán cơ bản, nhất quán và tin cậy, bao gồm một chuỗi các thao tác CSDL được thực hiện chuyển từ trạng thái CSDL nhất quán này sang trạng thái CSDL nhất quán khác ngay cả khi có một số giao dịch được thực hiện đồng thời và thậm chí cả khi xảy ra lỗi. Vì vậy, hệ quản trị CSDL phải hỗ trợ đầy đủ cho giao dịch đảm bảo rằng việc thực thi đồng thời các giao dịch của người sử dụng sẽ không vi phạm tính nhất quán của CSDL trong khi hệ thống có lỗi, với điều kiện là giao dịch được thực hiện chính xác, nghĩa là tuân theo các qui tắc toàn vẹn của CSDL.

1.5.5 Cải tiến hiệu năng

Hiệu năng của CSDL phân tán được cải tiến dựa vào hai điểm:

a) Hệ quản trị CSDL phân tán có khả năng phân mảnh CSDL khái niệm và cho phép cục bộ hoá dữ liệu. Có hai ưu điểm nổi bật:

- Vì mỗi trạm chỉ xử lý một phần CSDL, sự tranh chấp về CPU và các dịch vụ vào/ra không nghiêm trọng như trong các hệ CSDL tập trung.
- Tính cục bộ làm giảm trễ truy nhập từ xa thường gặp trên các mạng diện rộng.

Hầu hết các hệ CSDL phân tán được cấu trúc nhằm tận dụng tối đa những ưu điểm của tính cục bộ dữ liệu. Lợi ích đầy đủ của việc giảm tranh chấp và giảm chi phí truyền chỉ có thể có được bằng cách phân mảnh và phân tán dữ liệu hợp lý.

b) Tính song song của các hệ thống phân tán có thể được khai thác để thực hiện song song liên truy vấn và truy vấn nội bộ. Liên truy vấn song song là khả năng thực hiện nhiều truy vấn tại cùng thời điểm, còn nội truy vấn song song là phương pháp tách một truy vấn đơn thành các truy vấn con và mỗi truy vấn con được thực hiện tại các trạm khác nhau, truy nhập các phần khác nhau của CSDL phân tán.

1.5.6 Dễ dàng mở rộng hệ thống

Trong môi trường phân tán, dễ dàng tăng kích thước dữ liệu. và hiếm khi cần sửa đổi trong các hệ thống lớn. Việc mở rộng thường có thể được thực hiện bằng cách tăng khả năng lưu trữ và xử lý của mạng. Rõ ràng là không thể có được sự gia tăng "khả năng" một cách tuyến tính, vì điều này phụ thuộc vào chi phí phân tán. Tuy nhiên, vẫn có thể có những cải tiến có ý nghĩa. Khả năng mở rộng hệ thống dễ dàng mang tính kinh tế, chi phí giảm.

1.6 CÁC MÔ HÌNH CƠ SỞ DỮ LIỆU CLIENT/SERVER

Nhìn chung mọi ứng dụng cơ sở dữ liệu bao gồm các phần:

- Thành phần xử lý ứng dụng (Application Processing Components)
- Thành phần phần mềm cơ sở dữ liệu (Database Software Componets)
- Bản thân cơ sở dữ liệu (The Database Ifself)

Có 5 mô hình kiến trúc vật lý về truy nhập dữ liệu

- Mô hình cơ sở dữ liệu tập trung (Centralized database model)
- Mô hình cơ sở dữ liệu theo kiểu file - server (File - server database model)
- Mô hình xử lý từng phần cơ sở dữ liệu (Database extract processing model)
- Mô hình cơ sở dữ liệu Client/Server (Client/Server database model)
- Mô hình cơ sở dữ liệu phân tán (Distributed database model)

1.6.1 Mô hình cơ sở dữ liệu tập trung:

Trong mô hình này, các ứng dụng, hệ quản trị cơ sở dữ liệu và cơ sở dữ liệu được cài đặt trên cùng một bộ xử lý. Ví dụ trên máy tính cá nhân có thể chạy các chương trình ứng dụng có sử dụng phần mềm cơ sở dữ liệu Oracle để truy nhập tới cơ sở dữ liệu trên đĩa cứng của máy tính cá nhân đó.

Mô hình xử lý tập trung phù hợp với hầu hết công việc của nhiều tổ chức, doanh nghiệp...Ví dụ một bộ xử lý mainframe chạy phần mềm cơ sở dữ liệu IMS hoặc DB2 của IBM có thể cung cấp cho các trạm làm việc ở các vị trí phân tán truy nhập nhanh chóng tới cơ sở dữ liệu trung tâm. Tuy nhiên trong rất nhiều hệ thống, cả 3 thành phần của ứng dụng cơ sở dữ liệu đều thực hiện trên cùng một máy mainframe do vậy cấu hình này cũng thích hợp với mô hình tập trung

1.6.2 Mô hình cơ sở dữ liệu theo kiểu File Server:

Trong mô hình cơ sở dữ liệu theo kiểu File Server, các thành phần ứng dụng và phần mềm cơ sở dữ liệu ở trên một hệ thống máy tính và các File dữ liệu vật lý cơ sở dữ liệu cài đặt trên hệ thống máy tính khác. Một cấu hình như vậy thường được dùng trong môi trường cục bộ,

trong đó một hoặc nhiều hệ thống máy tính đóng vai trò của Server lưu trữ các file dữ liệu. Mô hình File Server giống với mô hình tập trung, cơ sở dữ liệu và các thành phần ứng dụng, phần mềm cơ sở dữ liệu cài đặt trên các máy tính khác nhau. Tuy nhiên các thành phần ứng dụng và phần mềm cơ sở dữ liệu có thể có cùng thiết kế để vận hành một môi trường tập trung. Hệ điều hành mạng có thể thực hiện cơ chế đồng thời cho phép nhiều người sử dụng cuối có thể truy nhập vào cùng cơ sở dữ liệu.

1.6.3 Mô hình xử lý từng phần cơ sở dữ liệu

Mô hình trong đó một cơ sở dữ liệu ở xa có thể được truy nhập bởi phần mềm cơ sở dữ liệu, được gọi là xử lý dữ liệu từng phần. Với mô hình này, người sử dụng có thể tại một máy tính cá nhân kết nối truy nhập, khai thác cơ sở dữ liệu ở xa. Với cách tiếp cận này, người sử dụng phải biết chắc chắn là dữ liệu nằm ở đâu và làm như thế nào để truy nhập dữ liệu. Phần mềm ứng dụng cần phải có trên cả hai hệ thống máy tính để kiểm soát sự truy nhập dữ liệu và chuyển dữ liệu giữa hai hệ thống. Tuy nhiên, phần mềm cơ sở dữ liệu chạy trên hai hệ thống không cần biết rằng việc xử lý cơ sở dữ liệu từ xa đang diễn ra vì người sử dụng tác động tới chúng một cách độc lập.

1.6.4 Mô hình cơ sở dữ liệu Client/Server

Trong mô hình cơ sở dữ liệu Client/Server, cơ sở dữ liệu được cài đặt trên Server, các ứng dụng trên các máy Client và phần mềm cơ sở dữ liệu được cài đặt trên cả Client lẫn Server. Trong mô hình này, các thành phần xử lý ứng dụng trên hệ thống Client đưa ra yêu cầu cho phần mềm cơ sở dữ liệu trên máy client, phần mềm này sẽ kết nối với phần mềm cơ sở dữ liệu chạy trên Server. Phần mềm cơ sở dữ liệu trên Server sẽ truy nhập vào cơ sở dữ liệu xử lý theo yêu cầu và gửi trả kết quả cho máy Client.

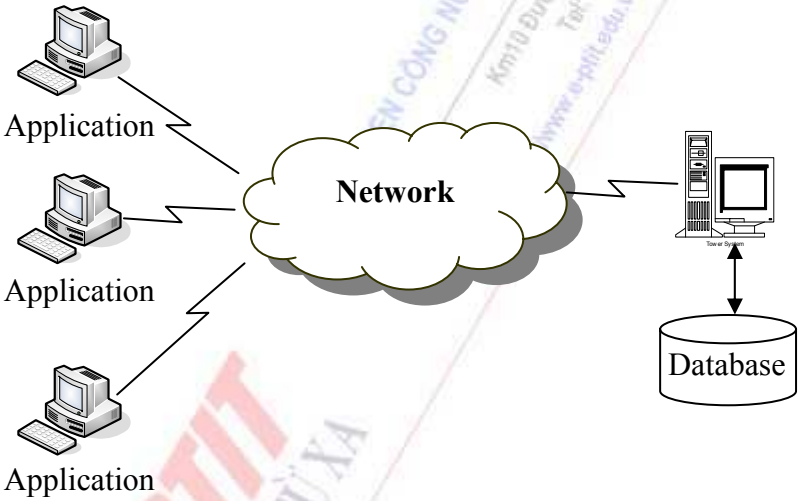
Mới nhìn, mô hình cơ sở dữ liệu Client/Server có vẻ giống như mô hình File Server, tuy nhiên mô hình Client/Server có rất nhiều thuận lợi hơn mô hình File Server. Với mô hình File Server, một giao tác cần truy nhập dữ liệu nhiều lần có thể gây ra tắc nghẽn lưu lượng truyền trên mạng. Giả sử người sử dụng tạo ra một vấn tin để lấy dữ liệu tổng số từ 1000 bản ghi, với cách tiếp cận File Server, nội dung của 1000 bản ghi phải được lưu chuyển trên mạng, vì phần mềm cơ sở dữ liệu chạy trên máy của người sử dụng phải truy nhập từng bản ghi để thoả mãn yêu cầu của người sử dụng. Với cách tiếp cận cơ sở dữ liệu Client/Server, chỉ có lời vấn tin khởi động ban đầu và kết quả cuối cùng cần đưa lên mạng, phần mềm cơ sở dữ liệu chạy trên máy lưu giữ cơ sở dữ liệu sẽ truy nhập các bản ghi cần thiết, xử lý chúng và gọi các thủ tục cần thiết để đưa ra kết quả cuối cùng.

Trong mô hình cơ sở dữ liệu Client/Server, thường nói đến các phần mềm Front End Software và Back End Software. Front End Software được chạy trên thiết bị truy nhập đầu cuối hoặc trên các Workstation, nhằm đáp ứng các yêu cầu xử lý đơn lẻ riêng biệt. Nó đóng vai trò của Client trong ứng dụng cơ sở dữ liệu Client/Server và thực hiện các chức năng hướng tới nhu cầu của người sử dụng. Front End Software chia thành các loại sau:

- End User Database Software: Phần mềm cơ sở dữ liệu này có thể được người sử dụng thực hiện trên thiết bị đầu cuối, truy nhập vào các cơ sở dữ liệu cục bộ, kết nối với các cơ sở dữ liệu trên Server.

- Simple Query and Reporting Software là phần mềm được thiết kế để cung cấp các công cụ xử lý dữ liệu từ cơ sở dữ liệu và tạo các báo cáo đơn giản từ dữ liệu đã có.
- Data Analysis Software cung cấp các hàm về tìm kiếm, khôi phục và cung cấp các phân tích phức tạp cho người sử dụng.
- Application Development Tools là phần mềm cung cấp các khả năng phát triển các ứng dụng cơ sở dữ liệu Bao gồm các công cụ về thông dịch, biên dịch đơn đến các công cụ CASE (Computer Aided Software Engineering). Chúng tự động tất cả các bước trong quá trình phát triển ứng dụng và sinh ra chương trình cho các ứng dụng.
- Database Administration Tools: Các công cụ cho phép người quản trị cơ sở dữ liệu thực hiện việc quản trị cơ sở dữ liệu như định nghĩa, lưu trữ hay phục hồi. CSDL

Back End Software được cài đặt trên Server cơ sở dữ liệu,. bao gồm phần mềm cơ sở dữ liệu Client/Server và phần mềm mạng



Hình 1.3 Mô hình Client-Server

1.6.5 Distributed database model (Mô hình cơ sở dữ liệu phân tán)

Cả hai mô hình File Server và Client/Server đều giả định là dữ liệu nằm trên một bộ xử lý và chương trình ứng dụng truy nhập dữ liệu nằm trên một bộ xử lý khác, còn mô hình cơ sở dữ liệu phân tán lại giả định bản thân cơ sở dữ liệu có ở trên nhiều máy khác nhau.

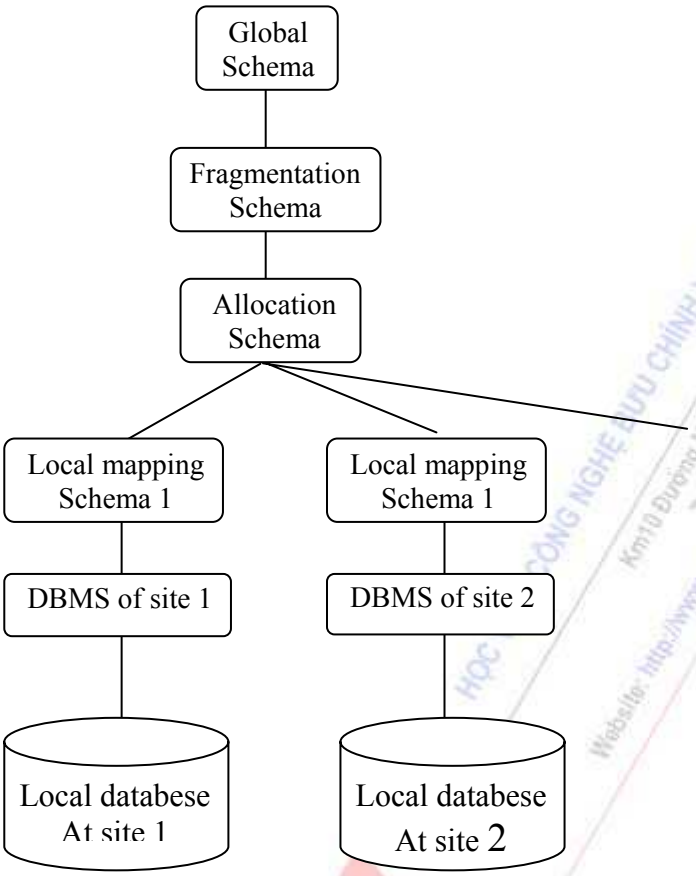
1.7 MÔ HÌNH THAM CHIẾU CƠ SỞ DỮ LIỆU PHÂN TÁN

Mô hình kiến trúc cơ sở dữ liệu phân tán tại các site gồm lược đồ tổng thể, lược đồ phân mảnh và lược đồ cấp phát.

1.7.1 Lược đồ toàn cục

Lược đồ toàn cục định nghĩa tất cả dữ liệu được chứa trong cơ sở dữ liệu phân tán như trong cở sở dữ liệu tập trung. Vì vậy, lược đồ toàn cục được định nghĩa chính xác như định nghĩa lược đồ cở sở dữ liệu tập trung. Tuy nhiên, mô hình dữ liệu lược đồ toàn cục cần phải tương thích với việc định nghĩa các ánh xạ tới các mức của cở sở dữ liệu phân tán. Vì vậy mô

hình dữ liêu quan hệ sẽ được sử dụng.trong kiến trúc mô hình tham chiếu cơ sở dữ liệu phân tán, định nghĩa một tập các quan hệ toàn cục.



Hình 1.4: Mô hình tham chiếu của cơ sở dữ liệu phân tán

1.7.2 Lược đồ phân mảnh

Mỗi quan hệ toàn cục có thể chia thành nhiều phần không chồng lặp lên nhau được gọi là phân mảnh. Ánh xạ giữa các quan hệ toàn cục và phân mảnh được định nghĩa là lược đồ phân mảnh. Ánh xạ này là mối quan hệ một-nhiều. Ví dụ, nhiều phân mảnh tương ứng với một quan hệ toàn cục, nhưng chỉ một quan hệ toàn cục tương ứng với một phân mảnh. Các phân mảnh được chỉ ra bằng tên của quan hệ toàn cục với một chỉ số (chỉ số phân mảnh), ví dụ, Ri chỉ đến phân mảnh thứ i trong quan hệ toàn cục R

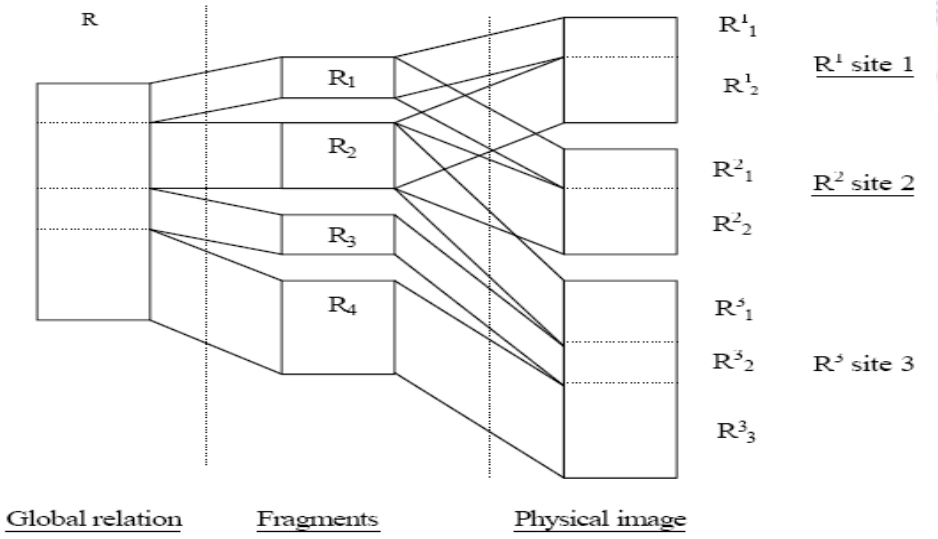
Các kiểu phân mảnh dữ liệu bao gồm phân mảnh ngang và phân mảnh dọc và một kiểu phân mảnh phức tạp hơn là sự hết hợp của 2 loại trên. Trong tất cả các kiểu phân mảnh, một phân mảnh có thể được định nghĩa bằng một biểu thức ngôn ngữ quan hệ cho các quan hệ toàn cục như là các toán hạng và kết quả đầu ra là các phân mảnh.

1.7.3 Lược đồ cấp phát

Các phân mảnh là những phần logic của các quan hệ toàn cục được chứa ở một hay nhiều site trong mạng. Lược đồ cấp phát xác định các phân mảnh được chứa ở những site nào. Tất cả các phân mảnh tương ứng với cùng một quan hệ R và được lưu ở dùng một site j tạo thành một mô hình vật lý của quan hệ toàn cục lên site j. Do đó, có một ánh xạ một-một giữa một

mô hình vật lý và một cặp là một quan hệ toàn cục được định danh và một chỉ số site tương ứng với một mô hình vật lý. Ký hiệu Rji tương ứng với mô hình vật lý mảnh thứ i của quan hệ R trên site j.

Một ví dụ của quan hệ giữa các kiểu đối tượng được định nghĩa như trên được biểu diễn trong hình sau. Một quan hệ toàn cục R chia thành 4 phân mảnh R1, R2, R3, R4. Bốn phân mảnh này được cấp phát dư tại 3 site của mạng máy tính, vì thế tạo nên ba mô hình vật lý R1 site 1, R2 site 2 và R3.site 3



Hình 1.5: Các phân mảnh và mô hình vật lý cho một quan hệ toàn cục

Có thể định nghĩa một bản sao của một phân mảnh tại một site cho trước và kí hiệu bằng tên quan hệ toàn cục R và hai chỉ số. Ví dụ R32 để chỉ bản sao của phân mảnh R2 được chứa ở site 3. Hai mô hình vật lý có thể giống nhau, ví là bản sao của nhau

Lược đồ các site phụ thuộc: gồm lược đồ ánh xạ cục bộ, DBMS của các site cục bộ, cơ sở dữ liệu ở site đó.

1.7.4 Lược đồ ánh xạ cục bộ

Do ba mức đầu các site độc lập, do đó chúng không phụ thuộc vào mô hình dữ liệu của DBMS cục bộ. Ở mức thấp hơn, nó cần phải ánh xạ mô hình vật lý thành các đối tượng được thao tác bởi các DBMS cục bộ. Ánh xạ này được gọi là lược đồ ánh xạ cục bộ và phụ thuộc vào kiểu của DBMS cục bộ. Trong hệ thống không đồng nhất có các kiểu khác nhau của ánh xạ cục bộ tại các site khác nhau. Yếu tố quan trong nhất để thiết kế kiến trúc này là:

- Phân mảnh và phân phát dữ liệu
- Quản lí dư thừa dữ liệu
- Sự độc lập của các DBMS cục bộ

1.7.5 DBMS ở các site cục bộ độc lập

Tính năng trong suốt trong ánh xạ cục bộ cho phép xây dựng một hệ thống cơ sở dữ liệu phân tán đồng nhất hoặc không đồng nhất. Trong hệ thống đồng nhất, các lược đồ độc lập của một site được định nghĩa sử dụng cùng một mô hình như DBMS cục bộ nhưng trong hệ thống không đồng nhất thì các lược đồ ánh xạ cục bộ dùng để phối hợp các kiểu khác nhau của DBMS…

1.8 CẤU TRÚC LOGIC CỦA CƠ SỞ DỮ LIỆU PHÂN TÁN

Có 3 kiểu thiết kế cơ sở dữ liệu phân tán trên mạng máy tính.

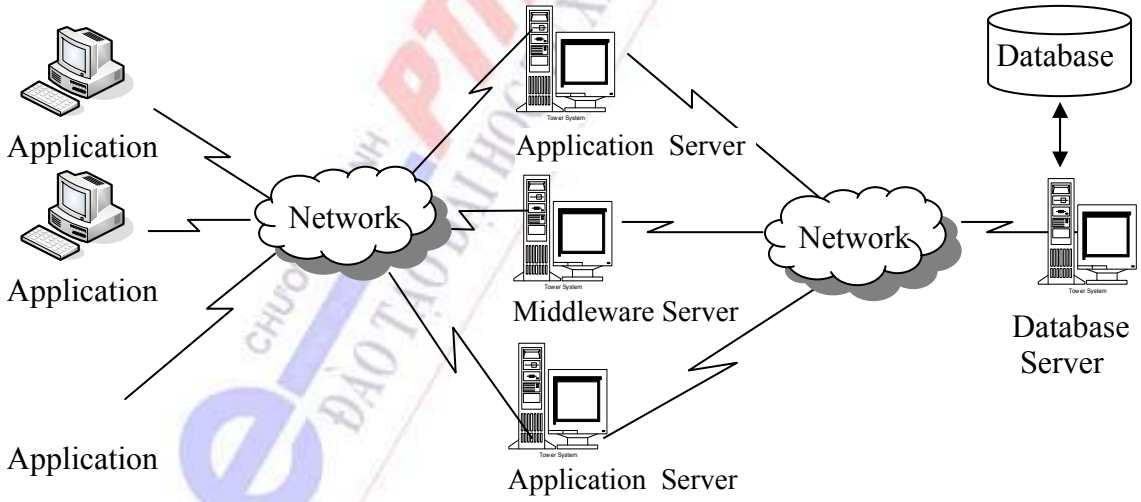
a) Các bản sao: Cơ sở dữ liệu được sao chép thành nhiều bản và được lưu trữ trên các site phân tán khác nhau của mạng máy tính.

b) Phân mảnh: Cơ sở dữ liệu được phân thành nhiều mảnh nhỏ theo kỹ thuật phân mảnh dọc hoặc phân mảnh ngang, các mảnh được lưu trữ trên các site khác nhau.

c) Mô hình kết hợp các bản sao và phân mảnh. Trên một số site chứa cấc bản sao, một số site khác chứa các mảnh

1.9 LỢI ÍCH PHÂN TÁN DỮ LIỆU TRÊN MẠNG

1. Việc phân tán dữ liệu tạo cho cơ sở dữ liệu có tính tự trị địa phương. Tại một site, dữ liệu được chia sẻ bởi một nhóm người sử dụng tại nơi họ làm việc và như vậy dữ liệu được kiểm soát cục bộ, phù hợp đối với những tổ chức phân bố tập trung. Cho phép thiết lập và bắt buộc sách lược địa phương đối với việc sử dụng cơ sở dữ liệu.



Hình 1.6 Mô hình Client-Server nhiều lớp

2. Tính song song trong các hệ cơ sở dữ liệu phân tán có thể nâng cao được hiệu quả truy nhập. Tính chất này có thể lợi dụng để xử lý song song các câu hỏi. Có hai dạng :

- Câu hỏi đồng thời phát sinh tại các trạm khác nhau.

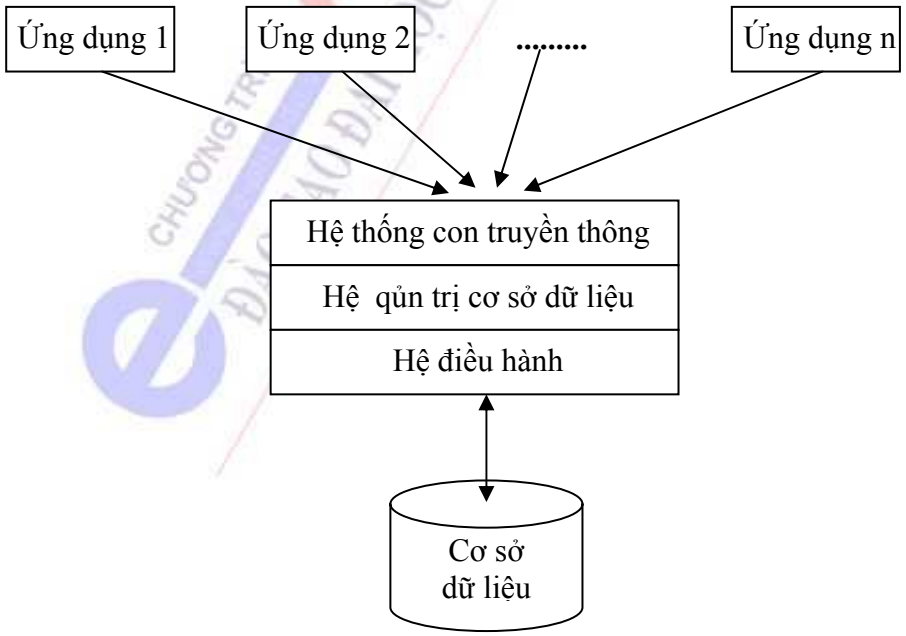
- Câu hỏi có thể được phân rã thành những câu hỏi thành phần được thực hiện song song tại các trạm khác nhau.

3. Trong tổ chức phân tán, tương tranh dịch vụ, CPU, vào/ra ít hơn so với tổ chức tập trung. Độ trễ trong truy nhập từ xa có thể giảm do việc thực hiện địa phương hoá dữ liệu một cách hợp lý.
4. Độ tin cậy và tính sẵn sàng được nâng cao trong tổ chức phân tán, là một trong những mục tiêu cơ bản của tổ chức dữ liệu phân tán. Việc tổ chức lặp dữ liệu cũng có thể đảm bảo cho việc truy nhập cơ sở dữ liệu không bị ảnh hưởng khi có sự cố xảy ra đối với trạm hoặc kênh truyền, không thể làm sụp đổ cả hệ thống.
5. Tổ chức dữ liệu phân tán kinh tế hơn so với tổ chức tập trung. Giá cho một hệ máy tính nhỏ rẻ hơn nhiều so với giá của một máy tính lớn khi triển khai cùng một mục đích ứng dụng. Giá chi phí truyền thông cũng ít hơn do việc địa phương hoá dữ liệu.
6. Khả năng mở rộng hệ thống và phân chia tài nguyên. Việc mở rộng khả năng cho một hệ xử lý phân tán là dễ dàng hơn và cho phép thực hiện tốt hơn.

1.10 HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU QUAN HỆ

1.10.1 Kiến trúc tổng quát

Hệ quản trị cơ sở dữ liệu quan hệ DBMS là một hệ thống phần mềm hỗ trợ mô hình quan hệ và ngôn ngữ quan hệ. DBMS khi thực hiện các giao dịch cần phải giao tiếp với 2 thành phần khác, đó là thành phần các hệ thống con truyền thông (Communication subsystem) và hệ điều hành (Operating system). Các hệ thống con truyền thông cho phép DBMS giao tiếp với các hệ thống truyền thông khác thông qua các ứng dụng. Hệ điều hành cung cấp giao diện giữa DBMS với các tài nguyên của máy. Kiến trúc tổng quát của hệ quản trị cơ sở dữ liệu quan hệ được mô tả trong hình 2.9 dưới đây.

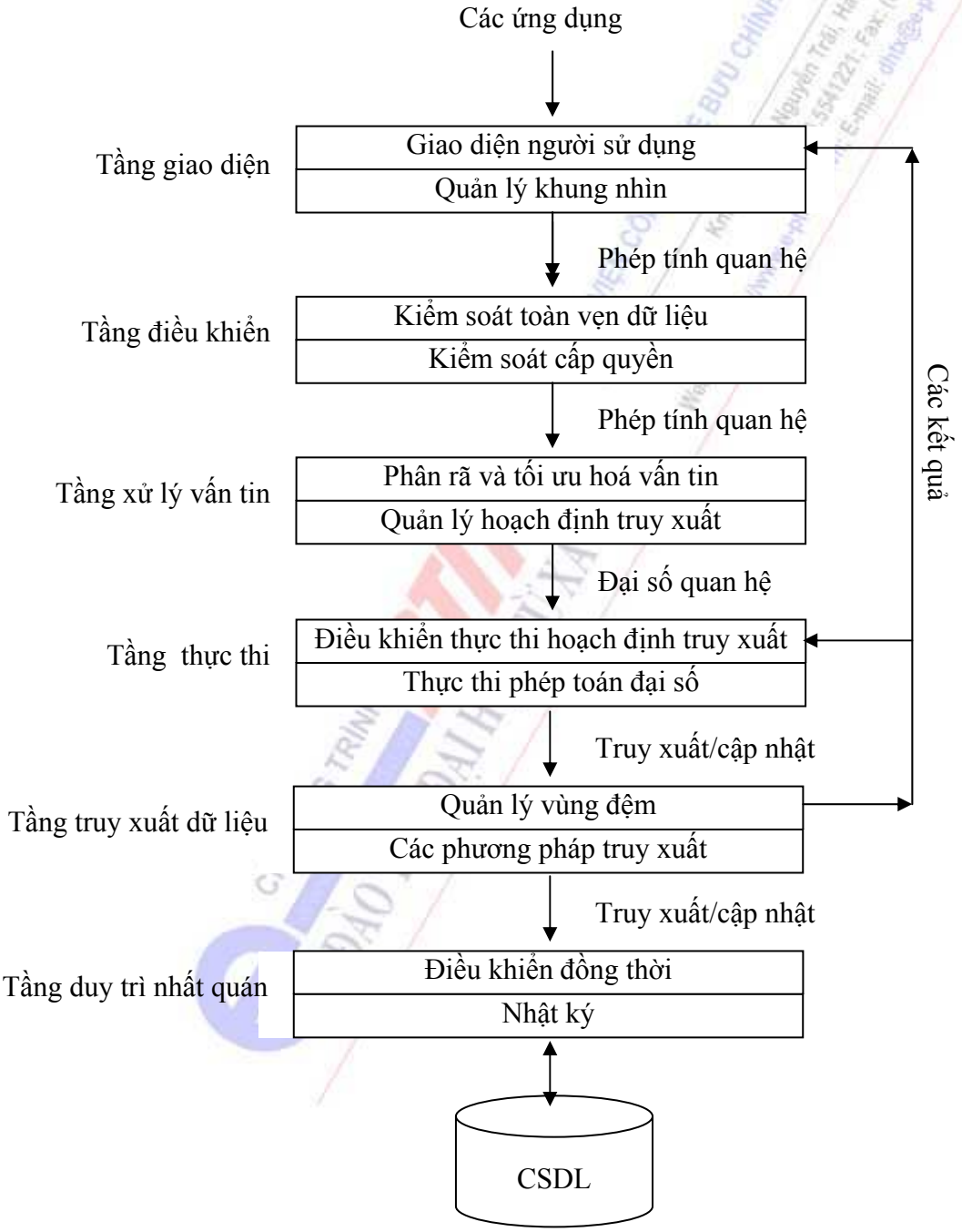


Hình 1.7: Kiến trúc tổng quát của mô hình hệ quản trị cơ sở dữ liệu quan hệ

1.10.2 Chức năng của hệ quản trị cơ sở dữ liệu quan hệ

Chức năng của hệ quản trị cơ sở dữ liệu quan hệ được phân thành nhiều tầng. Bao gồm các tầng giao diện, điều khiển, biên dịch, thực thi, tầng truy xuất dữ liệu và tầng duy trì nhất quán dữ liệu.

Tầng giao diện (Interface Layer): có chức năng quản lý giao diện với các các ứng dụng như giao diện trong SQL.. và các ứng dụng CSDL thực hiện trên các khung nhìn dữ liệu. Khung nhìn sẽ mô tả cách nhìn dữ liệu của các ứng dụng, là một quan hệ ảo dẫn xuất từ quan hệ cơ sở bằng cách áp dụng các phép đại số quan hệ. Việc quản lý khung nhìn bao gồm việc biên dịch câu vấn tin của người sử dụng thành dữ liệu khái niệm.



Hình 1.8: Các tầng chức năng của một hệ quản trị cơ sở dữ liệu quan hệ

Tầng điều khiển (Control Layer): Có chức năng điều khiển câu vấn tin bằng cách thêm các vị từ toàn vẹn dữ liệu và các vị từ cấp quyền truy nhập. Toàn vẹn dữ liệu và cấp quyền truy nhập đặc tả bằng các phép tính quan hệ. Kết quả của tầng này là câu vấn tin được biểu diễn bằng phép tính quan hệ.

Tầng xử lý vấn tin (Query Proccessing layer): Có chức năng ánh xạ câu vấn tin thành biểu thức đại số quan hệ - các chuỗi thao tác được tối ưu hoá. Tầng này có liên quan đến hiệu năng CSDL. Phân rã câu vấn tin thành một cây đại số, gồm các các phép toán đại số quan hệ. Kết quả sẽ được lưu trong một hoạch định truy xuất. Kết xuúat của tầng này là câu vấn tin được biểu diễn bằng đại số quan hệ.

Tầng thực thi (Execution Layer): Chịu trách nhiệm hướng dẫn việc thực hiện các hoạch định truy xuất, bao gồm các việc quản lý giao dịch và đồng bộ hoá các phép toán đại số quan hệ. Biên dịch các phép toán đại số quan hệ bằng cách gọi tầng truy xuất dữ liệu qua các yêu cầu truy xuất và cập nhật.

Tầng truy xuất dữ liệu (Data Access Layer): Tầng này thực hiện việc quản lý cấu trúc dữ liệu cài đặt các quan hệ. Quản lý các vùng đệm bằng cách lưu trữ tạm các dữ liệu thường được truy xuất nhiều nhất. Sử dụng tầng truy xuất dữ liệu làm giảm thiểu việc truy xuất dữ liệu trên đĩa từ.

Tầng duy trì nhất quán (Consistency Layer): Chức năng của tầng này là điều khiển các hoạt động đồng thời và ghi nhật ký các yêu cầu cập nhật. Cho phép khôi phục lại các giao dịch, hệ thống và thiết bị sau khi bị sự cố.

1.11 TỔNG QUAN VỀ HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU PHÂN TÁN

(Database Distributed Managment System)

1.11.1 Mở đầu

Một cách trực quan, một CSDL phân tán là một bộ sưu tập các loại dữ liệu có liên kết logic với nhau và được phân bố vật lý trên nhiều máy chủ của mạng máy tính. Khái niệm hệ CSDLPT bao gồm cả khái niệm CSDL và hệ quản trị CSDLPT.

Định nghĩa này nhấn mạnh hai khía cạnh quan trọng của CSDLPT:

- Tính phân tán: thực tế dữ liệu không cư trú trên cùng một site, vì vậy có thể phân biệt một CSDLPT với cơ sở dữ liệu tập trung (CSDLTT).
- Sự tương quan logic: các loại dữ liệu có một số tính chất ràng buộc lẫn nhau, như vậy có thể phân biệt CSDLPT với tập các CSDL địa phương hoặc với các tệp lưu trữ trên các site khác nhau.

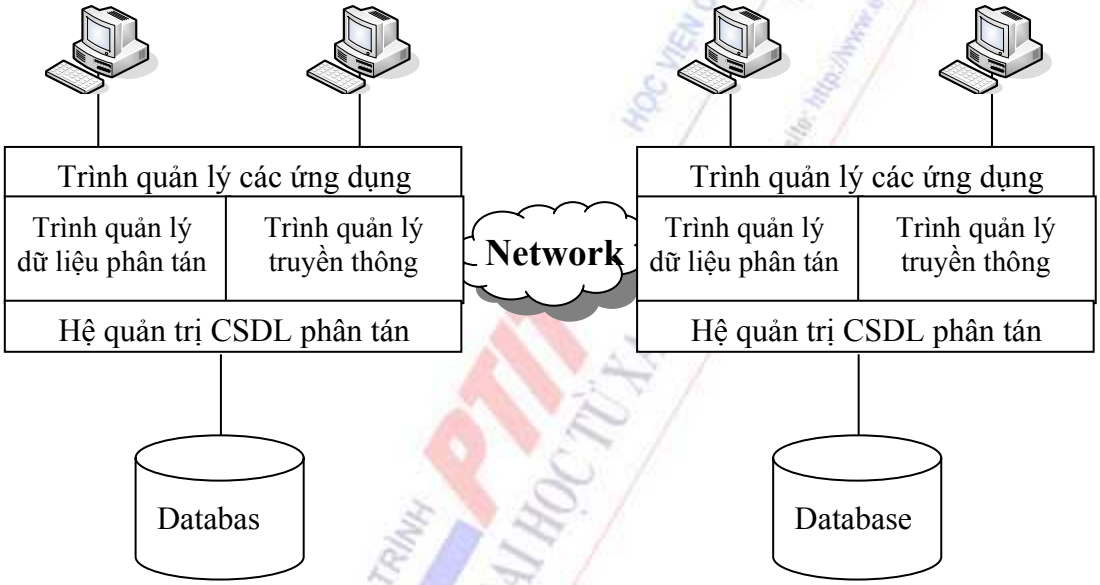
Hệ quản trị CSDL phân tán là hệ thống phần mềm cho phép quản trị CSDL phân tán và làm cho sự phân tán đó là trong suốt đối với người sử dụng. Nói cách khác CSDL phân tán là CSDL được phân tán một cách vật lý nhưng được thống nhất tổ chức như là một CSDL duy nhất.

Như vậy sự phân tán dữ liệu là trong suốt đối với người sử dụng. Việc quản lý các dữ liệu phân tán đòi hỏi mỗi trạm (site) cài đặt các thành phần hệ thống sau:

- Thành phần quản trị CSDL (Database Management DM)
- Thành phần truyền dữ liệu (Data Communication DC)
- Từ điển dữ liệu (Data Dictionary DD): thông tin về sự phân tán dữ liệu trên mạng
- Thành phần CSDLPT (Distributed Database DDB)

Các dịch vụ của hệ thống trên bao gồm:

- Các ứng dụng truy nhập CSDL từ xa .
- Cung cấp các mức trong suốt phân tán.
- Hỗ trợ quản trị và điều khiển CSDL, bao gồm các bộ công cụ, thu thập thông tin từ các trình tiện ích, cung cấp cách nhìn tổng quan về các file dữ liệu trên mạng.
- Khả năng mở rộng với các hệ thống khác nhau
- Cung cấp khả năng điều khiển đồng thời và phục hồi các giao tác phân tán.



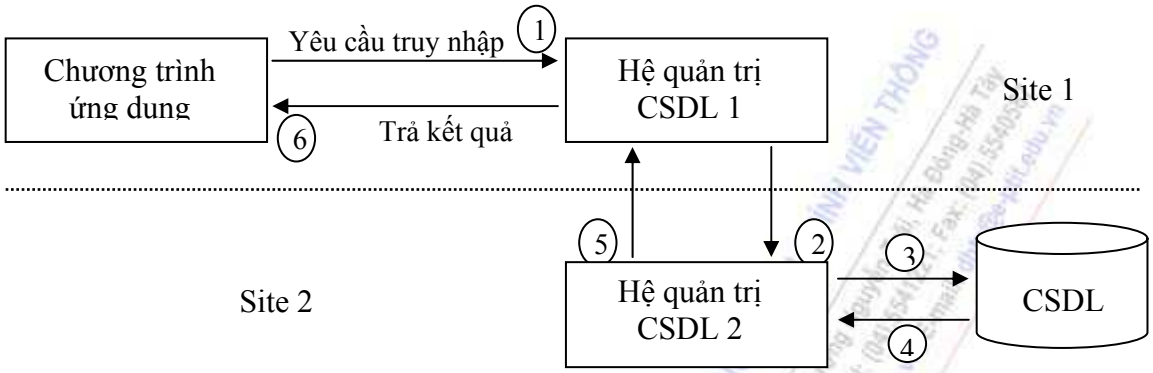
Hình 1.9 Hệ quản trị CSDL phân tán

Các hệ QTCSDL phân tán thường hỗ trợ về điều khiển tương tranh và khôi phục các tiến trình phân tán. Khả năng truy cập từ xa có thể thực hiện được bằng 2 cách.

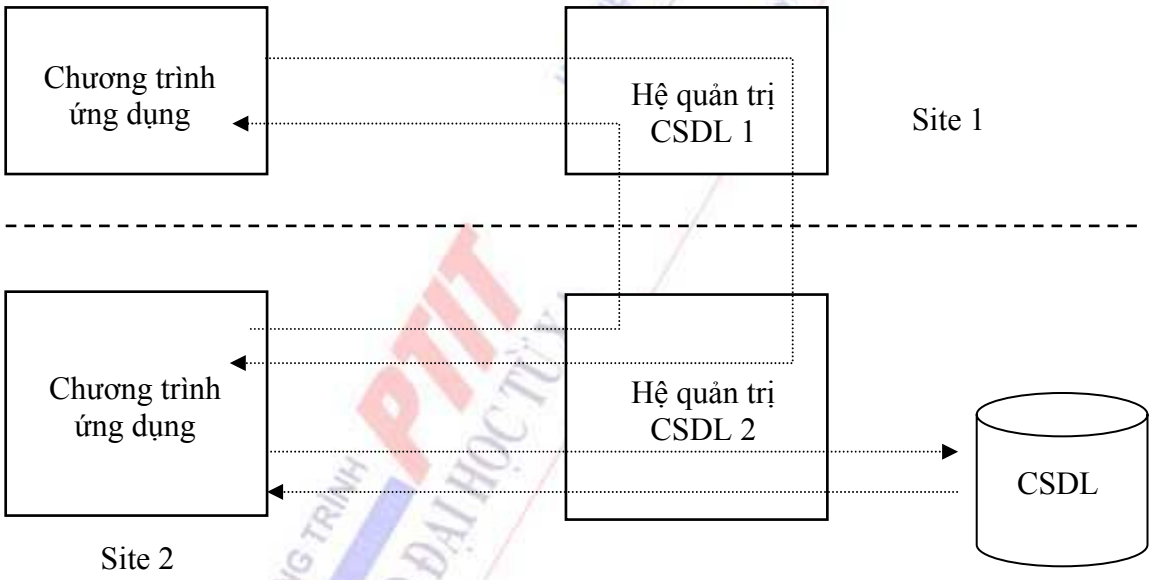
Cách thứ nhất (hình 1.10a) trình ứng dụng yêu cầu truy cập từ xa. Yêu cầu này được định tuyến tự động bởi DDBMS tới máy chủ chứa dữ liệu. Được thực hiện tại máy chủ chứa cơ sở dữ liệu và gửi lại kết quả. về trạm yêu cầu. Cách tiếp cận này được sử dụng cho truy cập từ xa, trong suốt phân tán có thể thực hiện được bằng việc cung cấp các file chung (global) và các truy nhập trước đó có thể địa chỉ hoá một cách tự động tới các trạm ở xa.

Hình 1.10 b chỉ ra một cách tiếp cận khác, chương trình phụ thực hiện tại các trạm ở xa (người lập trình phải tự lập), các kết quả trả lại cho chương trình ứng dụng.

Hệ quản trị CSDL phân tán hỗ trợ cả hai cách tiếp cận trên. Mỗi một cách tiếp cận đều có những thuận lợi và khó khăn riêng. Giải pháp thứ nhất cung cấp khả năng trong suốt phân tán cao hơn, trong khi giải pháp thứ hai có thể hiệu quả hơn nếu như có rất nhiều chương trìnắtngs dụng cùng yêu cầu truy nhập, bởi vì các chương trình phụ có thể thực hiện các yêu cầu từ các trạm ở xa và trả lại kết quả.



Hình 1.10a Truy nhập CSDL từ xa



Hình 1.10b Truy nhập từ xa bằng chương trình phụ

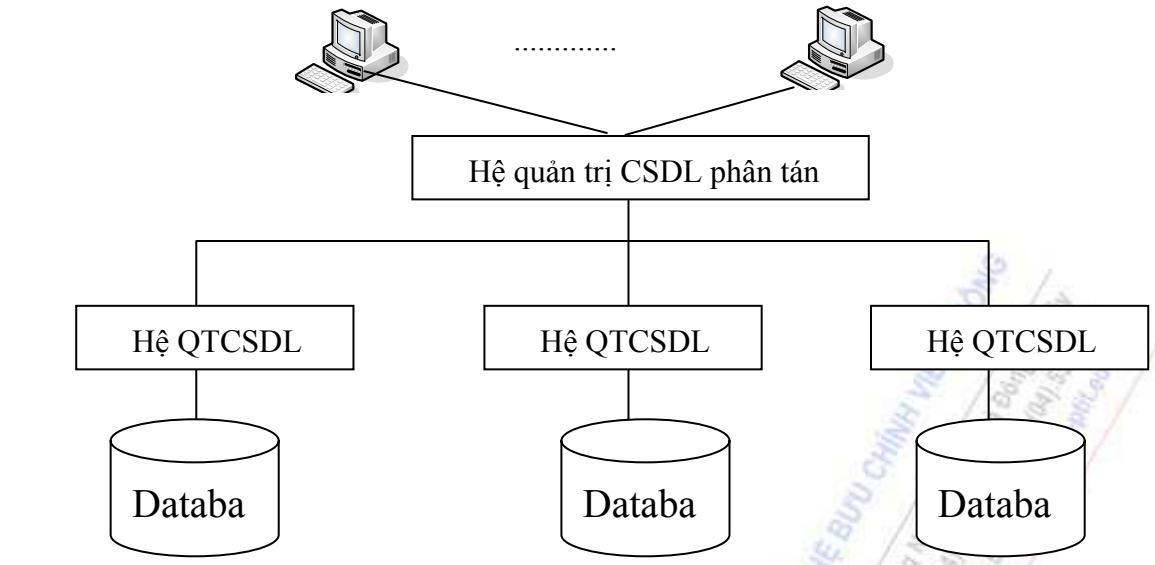
1.11.2 Hệ quản trị CSDL phân tán thuần nhất

CSDLPT có được bằng cách chia một CSDL thành một tập các CSDL cục bộ (Local) và được quản lý bởi cùng một hệ QTCSDL, trong hình 2.13

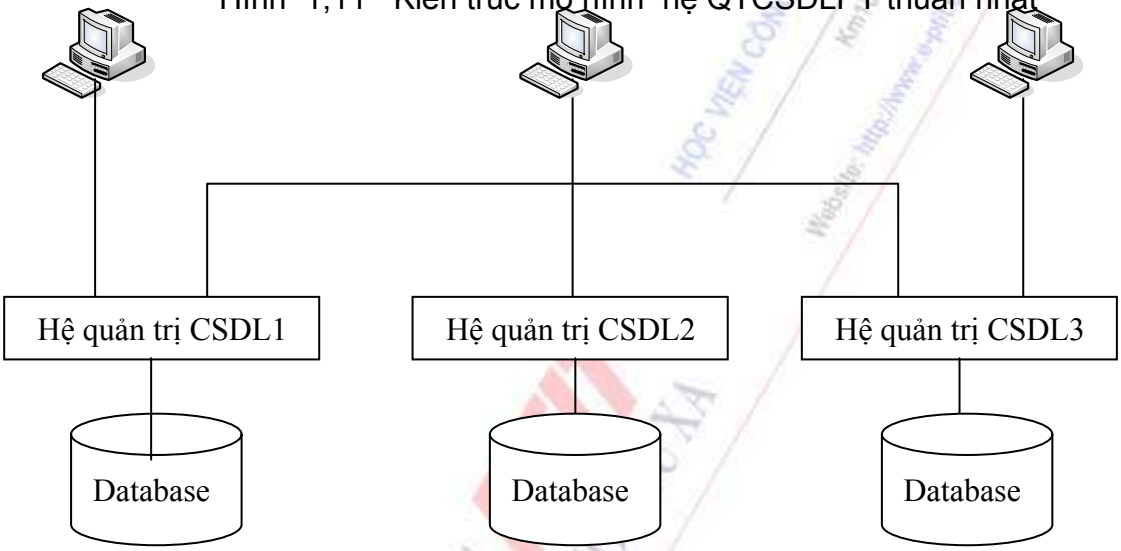
CSDLPT có thuần nhất hay không được phụ thuộc bởi các yêu tố phần cứng, hệ điều hành và các hệ quản trị CSDL cục bộ. Tuy nhiên, hạn chế quan trọng tại hệ QTCSDL cục bộ, bởi vì nó phụ thuộc vào sự quản lý hệ điều hành mạng truyền thông

1.11.3 Hệ quản trị CSDL phân tán không thuần nhất

CSDLPT không thuần nhất được tích hợp bởi một tập các CSDL cục bộ được quản lý bởi các hệ QTCSDL khác nhau. Hệ QTCSDLPT không thuần nhất thêm việc chuyển đổi các mô hình dữ liệu của các hệ QTCSDL khác nhau để thống nhất việc quản lý. Hình 14



Hình 1,11 Kiến trúc mô hình hệ QTCSDLPT thuần nhất



Hình 1.12 Kiến trúc mô hình hệ QTCSDLPT không thuần nhất

Nếu việc phát triển CSDL phân tán theo mô hình Top-down, không phụ thuộc vào hệ thống trước đó (hệ thống các CSDL cục bộ), thì việc phát triển một hệ thuần nhất là tốt nhất. Tuy nhiên, trong một số trường hợp cần xây dựng CSDL phân tán từ các CSDL đã có thì đòi hỏi phải phát triển một hệ không thuần nhất. Phương pháp tốt nhất là tiếp cận từ dưới lên (Bottum-up). Trình quản lý dữ liệu phân tán phải cung cấp các giao diện trao đổi giữa các hệ QTCSDL . Vấn đề quản trị CSDL phân tán không thuần nhất rất khó khăn.

1.12 MÔ HÌNH KIẾN TRÚC HỆ QUẢN TRỊ CSDL PHÂN TÁN

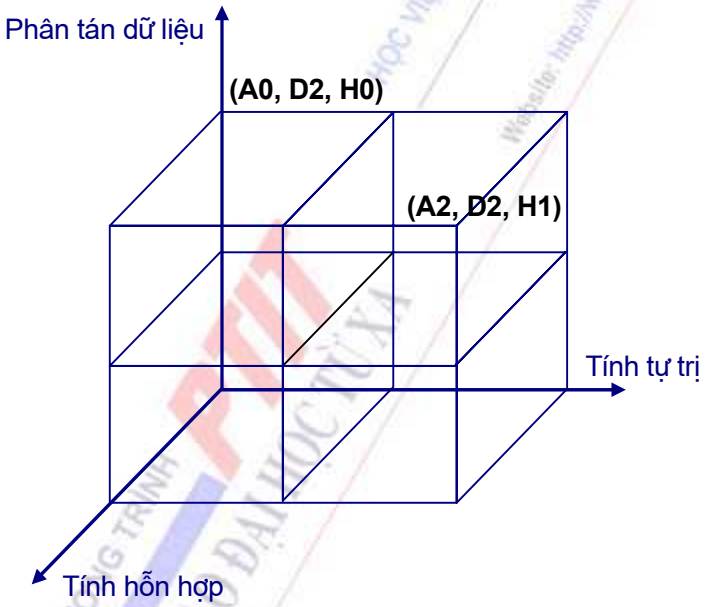
Có ba kiểu kiến trúc tham chiếu cho hệ quản trị CSDL phân tán , đó là hệ Client Server, hệ quản trị CSDL phân tán kiểu ngang hàng (Peer-to-Peer) và hệ đa CSDL.

Các lựa chọn cài đặt một hệ quản trị CSDLđược tổ chức hệ thống theo các đặc tính: (1) tính tự trị, (2) tính phân tán, (3) tính hỗn hợp (không thuần nhất) của hệ thống.

1.12.1 Tính tự vận hành

Tính tự vận hành hay còn gọi là tính tự trị, được hiểu là sự phân tán quyền điều khiển. Là mức độ hoạt động độc lập của từng hệ quản trị CSDL riêng lẻ. Tính tự vận hành được biểu hiện qua chức năng của một số yếu tố, như sự trao đổi thông tin giữa các hệ thống thành viên với nhau, thực hiện giao dịch độc lập/ không đôck lập và có được phép sửa đổi chúng hay không. Yêu cầu của hệ thống tự vận hành được xác định theo nhiều cách. Ví dụ,

- Các thao tác cục bộ của hệ quản trị CSDL riêng lẻ không bị ảnh hưởng khi tham gia hoạt động trong hệ đa CSDL (Multi Database System).
- Các hệ quản trị CSDL xử lý và tối ưu truy vấn cũng không bị ảnh hưởng bởi thực thi truy vấn toàn cục truy nhập nhiều hệ CSDL.
- Tính nhất quán của hệ thống hoặc thao tác không bị ảnh hưởng khi các hệ quản trị CSDL riêng lẻ kết nối hoặc tách rời khỏi tập các CSDL.



Hình 1.13. Lựa chọn cài đặt hệ quản trị CSDL

Mặt khác, xác định chiều của tính tự trị như sau:

1. Tự trị thiết kế: Mỗi hệ quản trị CSDL riêng lẻ có thể sử dụng mô hình dữ liệu và kỹ thuật quản lý giao dịch theo ý muốn.
2. Tự trị truyền thông: Mỗi hệ quản trị CSDL riêng lẻ tuỳ ý đưa ra quyết định của nó về loại thông tin mà nó cần cung cấp cho các hệ quản trị CSDL khác hoặc phần mềm điều khiển thực thi toàn cục của nó.
3. Tự trị thực thi: Mỗi hệ quản trị CSDL có thể thực thi các giao dịch được gửi tới nó theo bất kỳ cách nào mà nó muốn.

Ba lựa chọn xem xét ở trên cho các hệ thống tự trị không phải là những khả năng duy nhất, mà là ba lựa chọn phổ biến nhất.

1.12.2 Tính phân tán dữ liệu

Tính phân tán dữ liệu: Tính tự vận hành đề cập đến việc phân tán quyền điều khiển, thì tính phân tán dữ liệu đề cập đến dữ liệu. Hiển nhiên, sự phân tán vật lý của dữ liệu trên nhiều vị trí khác nhau. Người sử dụng nhìn dữ liệu bằng khung nhìn dữ liệu. Có hai cách phân tán dữ liệu: phân tán kiểu Client/Server và phân tán kiểu ngang hàng. Kết hợp với các tùy chọn không phân tán, trục kiến trúc cho ba loại kiến trúc khác nhau.

- Phân tán kiểu Client/Server ngày càng phổ biến. Quản trị dữ liệu tại Server, Client cung cấp môi trường ứng dụng và giao diện người sử dụng. Nhiệm vụ truyền thông được chia sẻ giữa các Client và Server. Hệ quản trị CSDL kiểu Client/Server là hệ phân tán chức năng. Có nhiều cáchễuây dựng, mỗi cách cung cấp một mức độ phân tán khác nhau.
- Trong kiểu ngang hàng không có sự khác biệt giữa chức năng Client và Server. Mỗi máy đều có đầy đủ chức năng của hệ quản trị CSDL và có thể trao đổi thông tin với các máy khác để thực hiện các truy vấn và giao dịch. Các hệ thống này cũng được gọi là phân tán đầy đủ,

1.12.3 Tính hỗn hợp

Tính hỗn hợp: Từ khác biệt về phần cứng và các giao thức mạng đến khác biệt trong cách quản lý dữ liệu, có một số dạng hỗn hợp trong các hệ phân tán. Sự khác biệt lớn nhất liên quan đến các mô hình dữ liệu, ngôn ngữ truy vấn và giao thức quản lý giao dịch. Biểu diễn dữ liệu bằng nhiều mô hình khác nhau tạo ra tính hỗn hợp. Tính hỗn hợp trong ngôn ngữ truy vấn không chỉ bao gồm việc sử dụng các dạng truy nhập dữ liệu khác nhau trong các mô hình dữ liệu khác nhau , mà còn bao gồm những khác biệt trong các ngôn ngữ ngay cả khi sử dụng cùng một mô hình dữ liệu. Ngôn ngữ truy vấn khác nhau sử dụng cùng một mô hình dữ liệu thường chọn các phương pháp khác nhau để diễn tả các yêu cầu giống nhau, ví dụ, DB2 sử dụng SQL, trong khi INGRES sử dụng QUEL.

1.12.4 Các kiểu kiến trúc

Xem xét các kiến trúc trong hình 2.15, bắt đầu từ gốc và di chuyển theo trục tự trị. Ký hiệu A là tự trị, D là phân tán và H là hỗn hợp. Các kiểu trên trục tự trị được định nghĩa, A0 là biểu diễn tích hợp chặt chẽ,A1 biểu diễn hệ bán tự trị và A2 biểu diễn hệ cô lập. Trên trục phân tán, D0 nghĩa là không phân tán, D1 là hệ Client/Server, và D2 là phân tán ngang hàng. Trên trục hỗn hợp, H0 xác định các hệ thống thuần nhất, H1 là các hệ hỗn hợp. Trong hình 2.15 định nghĩa hai loại kiến trúc: (A0, D2, H0) là hệ quản trị CSDL thuần nhất phân tán (ngang hàng) và (A2, D2, H1) là phức hệ CSDLhỗn hợp, phân tán ngang hàng.

- Loại kiến trúc (A0, D0, H0): Được gọi là hệ thống phức hợp (Composite System). Nếu không phân tán dữ liệu và hỗn hợp, thì hệ thống chỉ là một tập gồm nhiều hệ quản trị CSDL được tích hợp về mặt lôgic. Phù hợp với các hệ thống đa xử lý và tài nguyên đều dùng chung. Kiểu này không xuất hiện nhiều trong thực tế

- Loại kiến trúc (A0, D0, H1): Nếu hỗn hợp thì phải có nhiều bộ quản lý dữ liệu hỗn hợp có thể cung cấp một khung nhìn tích hợp cho người sử dụng. Trước đây được thiết kế truy nhập tích hợp CSDL mạng, phân cấp và quan hệ trên cùng một máy đơn.
- (A0, D1, H0):Trường hợp CSDL phân tán khi có một khung nhìn tích hợp về dữ liệu cung cấp cho người sử dụng. Hệ thống loại này thích hợp cho phân tán Client/Server.
- (A0, D2, H0): Biểu diễn môi trường phân tán hoàn toàn trong suốt cung cấp cho người sử dụng. Không phân biệt giữa Client và Server, cung cấp đầy đủ các chức năng.
- (A1, D0, H0): Là dạng các hệ thống bán tự trị. Các hệ thống thành viên có quyền tự trị nhất định trong các hoạt động của chúng. Kiến trúc này sử dụng thiết lập bộ khung cho hai dạng kiến trúc kế tiếp. Trong thực tế rất ít sử dụng
- (A1, D0, H1): Là hệ thống hỗn hợp và tự trị., rất phổ biến hiện nay. Một ví dụ hệ thống loại này bao gồm một hệ quản trị CSDL quan hệ quản lý dữ liệu có cấu trúc, một hệ quản trị CSDL xử lý hình ảnh tĩnh và một Server cung cấp video. Để cung cấp hình ảnh tích hợp cho người sử dụng, cần phải che dấu tính tự động và tính hỗn hợp của các hệ thống thành viên và thiết lập một giao diện chung.
- (A1, D1, H1): Trong các hệ thống loại này, các hệ thống thành viên được cài đặt trên các máy khác nhau. Được gọi là các hệ quản trị CSDL hỗn hợp phân tán. Đặc điểm phân tán ít quan trọng hơn so với tính tự trị và hỗn hợp. Các hệ quản trị CSDL kiểu (A0, D1, H0) và (A0, D2, H0) có thể giải quyết những vấn đề khó khăn khi phân tán dữ liệu
- (A2, D0, H0): Đặc điểm của các hệ thống loại này là các thành viên không có khái niệm thỏa hiệp và không biết cách liên lạc với nhau. Nếu không có tính hỗn hợp hoặc tính phân tán thì một phức hệ CSDL chỉ là một tập các CSDL tự trị được kết nối với nhau. Hệ quản trị phức hệ CSDL cho phép quản lý tập hợp các CSDL tự trị và cho phép truy nhập trong suốt đến nó. Dạng hệ thống này ít thực tế.
- (A2, D0, H1): Hệ thống loại này có tính thực tế cao, hơn cả (A1, D0, H1).Có khả năng xây dựng các ứng dụng truy nhập dữ liệu từ nhiều hệ thống lưu trữ khác nhau với các đặc tính khác nhau. Có thể là những hệ thống lưu trữ không phải là hệ quản trị CSDL và không được thiết kế phát triển có thể tương tác với các phần mềm khác. Cũng như trong hệ (A1, D0, H1), giả thiết các hệ thống thành viên không tham gia vào toàn bộ hệ thống.
- (A2, D1, H1) và (A2, D2, H1): Hai trường hợp này đều biểu diễn cho trường hợp các CSDL thành viên tạo ra phức hệ CSDLđược phân tán trên mọt số vị trí – gọi là các phức hệ CSDL phân tán. Cả hai trường hợp các giải pháp phân tán và xử lý tương tác tương tự nhau. Trong trường hợp phân tán Client/Server (A2, D1, H1), các vấn đề tương tác được trao cho hệ thống trung gian (Middleware System), tạo ra kiến trúc ba tầng.

Tổ chức của một phức hệ CSDL phân tán và việc quản lý nó hoàn toàn khác với các hệ quản trị CSDL phân tán. Sự khác biệt cơ bản của chúng là ở mức độ tự trị của các chương trình quản lý dữ liệu cục bộ. Các phức hệ CSDL phân tán hoặc tập trung đều có thể thuần nhất hoặc hỗn hợp, không thuần nhất

Sự phân tán, tính hỗn hợp và tính tự trị của CSDL là các vấn đề liên quan đến nhau. Mục tiêu của tài liệu là các hệ phân tán nên chú ý nhiều hơn tính hỗn hợp và tính tự trị.

1.13 KIẾN TRÚC HỆ QUẢN TRỊ CSDL PHÂN TÁN

Phần này sẽ xem xét chi tiết ba kiến trúc hệ thống trong số các kiến trúc đã được trình bày ở trên. Ba loại kiến trúc là:

- Hệ Client/Server, bỏ qua các vấn đề hỗn hợp và tự trị có dạng (Ax, D1, Hy).
- Các CSDL phân tán, ứng với (A0, D2, H0).
- Hệ đa CSDL, ứng với (A2, Dx, Hy).

1.13.1 Các hệ Client/Server

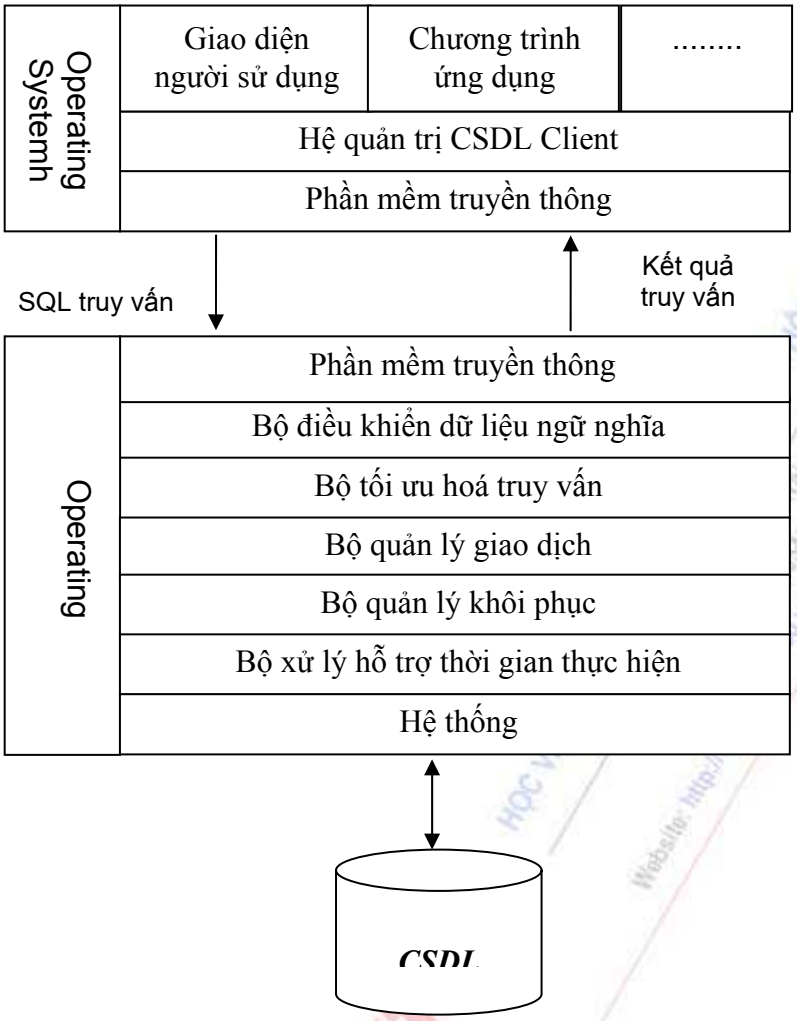
Các hệ quản trị CSDL Client/Server cung cấp kiến trúc hai lớp chức năng Server và chức năng Client, nhằm tạo ra sự dễ dàng trong việc quản lý tính phức tạp của các hệ quản trị CSDL hiện đại và tính phức tạp của việc phân tán dữ liệu

Server thực hiện hầu hết các công việc quản lý dữ liệu. Nghĩa là tất cả mọi xử lý và tối ưu hoá truy vấn, quản lý giao dịch và quản lý lưu trữ đều được thực hiện trên Srver. Client, ngoài ứng dụng và giao diện người sử dụng, có một module hệ quản trị CSDL Client trách nhiệm quản lý dữ liệu và khóa giao dịch được gửi đến Client. Client và Server trao đổi với nhau bởi các câu lệnh SQL. Cụ thể hơn, Client chuyển truy vấn SQL đến Server, Server sẽ thực hiện và trả lại kết quả cho Client.

Loại kiến trúc Client/Server đơn giản chỉ có một Server được truy nhập bởi nhiều Client, gọi là đa Client-một Server. Việc quản lý dữ liệu không khác so với CSDL tập trung. CSDL được lưu chỉ trên Server và có phần mềm quản lý nó. Tuy nhiên, sự khác biệt quan trọng so với các hệ thống tập trung là cách thực thi giao dịch và quản lý bộ nhớ Cache.

Loại kiến trúc có nhiều Server trong hệ thống, được gọi là đa Client-đa Server. Có hai chiến lược quản lý: hoặc Client quản lý kết nối của nó tới Server hoặc Client chỉ biết Server chủ của nó và liên lạc với các Server khác qua Server chủ khi có yêu cầu. Chiến lược thứ nhất làm đơn giản cho các Server, nhưng lại gắn thêm nhiều trách nhiệm cho các máy Client. Điều này dẫn đến một hệ thống được gọi là hệ máy khách tự phục vụ. Mặt khác, với chiến lược thứ hai, tập trung vào chức năng quản lý dữ liệu tại Server. Vì vậy, tính trong suốt của truy nhập dữ liệu được cung cấp tại giao diện Server.

Mô hình CSDLlogic Client/Server là duy nhất. Mô hình mức vật lý của nó có thể phân tán. Vì vậy phân biệt giữa Client/Server và ngang hàng không phải ở mức độ trong suốt được cung cấp cho người sử dụng và cho ứng dụng mà ở mô hình kiến trúc được dùng để nhận ra mức độ trong suốt..

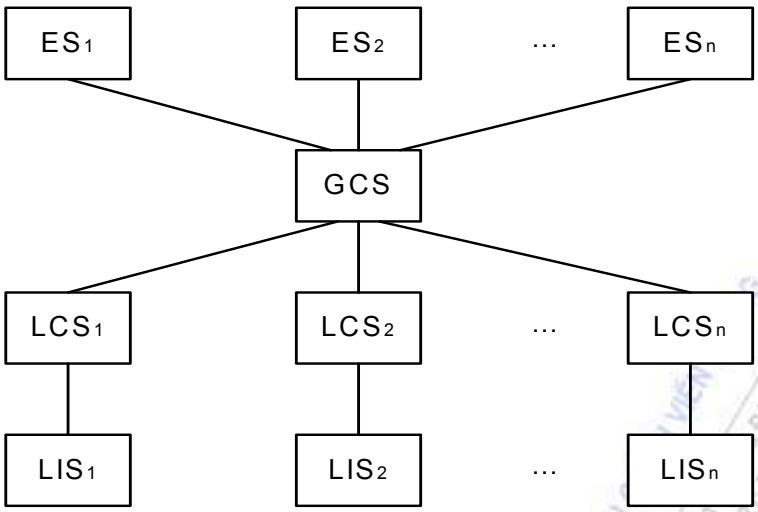


Hình 1.14 Kiến trúc tham chiếu Client/Server

1.13.2 Các hệ phân tán ngang hàng(Peer to Peer)

Trước tiên khảo sát về tổ chức dữ liệu vật lý trong các hệ mgamg hàng. Tổ chức lưu trữ trên các máy khác nhau có thể khác mhau. Điều này có nghĩa là cần phải có một định nghĩa nội tại riêng cho mỗi vị trí, được gọi là lược đồ nội tại cục bộ LIS (Local Internal Schema). Lược đồ khái niệm toàn cục mô tả cấu trúc logic của dữ liệu ở mọi vị trí.

Dữ liệu trong một CSDL phân tán thường được phân mành và nhân bản trên các vị trí khác nhau. Vì vậy cần phải mô tả tổ chức lưu trữ dữ liệu vật trên mọi vị trí. Cần bổ sung thêm tầng thứ trong kiến trúc cơ sở dữ liệu 3 mức, đó là lược đồ khái niệm cục bộ LCS (Local Conceptual Schema). Vì vậy lược đồ khái niệm toàn cục GCS (Global Conceptual Schema) là hợp của các lược đồ khái niệm cục bộ. Mức trên cùng là khung nhìn dữ liệu của người sử dụng, lược đồ ngoài ES (External Schema). Người sử dụng khác nhau có cách nhìn dữ liệu cũng khác nhau. Như vậy kiến trúc của hệ cơ sở dữ liệu phân tán có 3 mức: Có nhiều khung nhìn dữ liệu khác nhau trong mức lược đồ ngoài, nhưng chỉ có duy nhất một mô hình khái niệm toàn cục và có nhiều mô hình khái niệm cục bộ, ứng với lược đồ trong cục bộ trên mỗi vị trí.

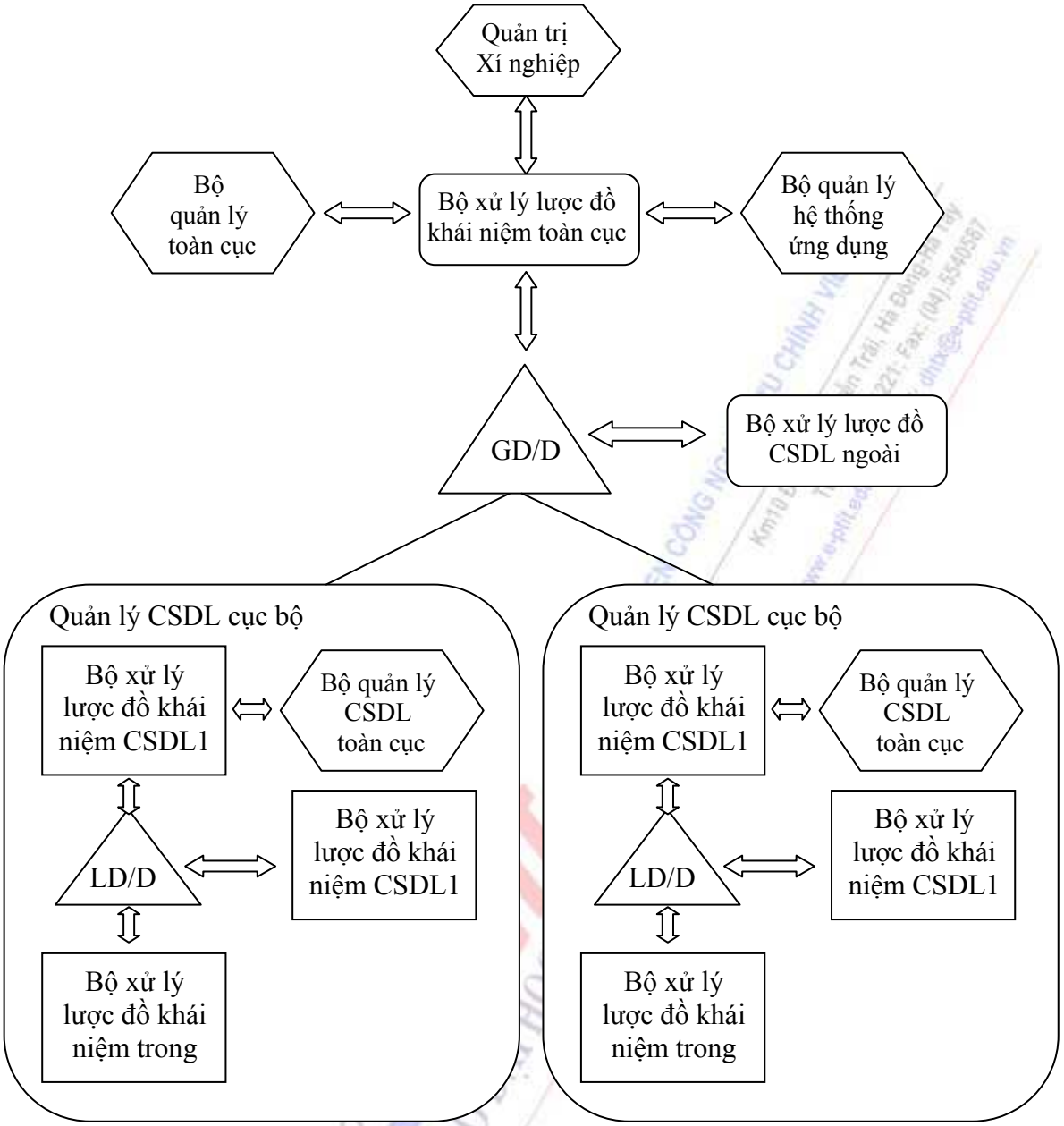


Hình 1.15 Kiến trúc tham chiếu CSDL phân tán.

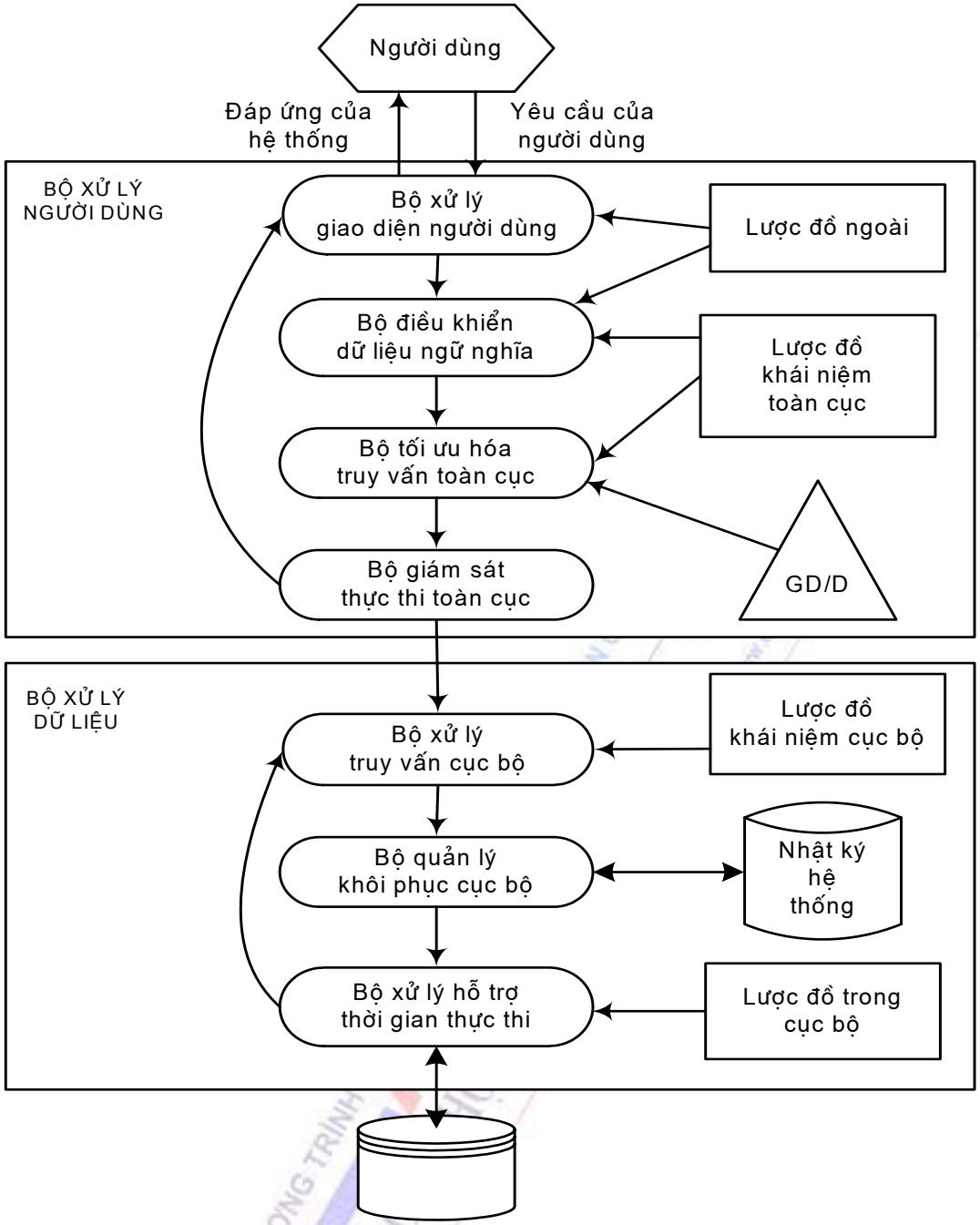
Mô hình trong hình 1.15 được mở rộng từ mô hình ANSI/SPARC. Nó phản ảnh tính trong suốt và tính độc lập dữ liệu .Trong suốt định vị và trong suốt nhân bản được hỗ trợ bằng các lược đồ khái niệm cục bộ và toàn cục và ánh xạ giữa chúng. Mặt khác, trong suốt mạng được hỗ trợ bằng lược đồ khái niệm toàn cục. Người sử dụng truy vấn dữ liệu không cần biết đến vị trí hay các thành phần CSDLcục bộ. Hệ quản trị CSDL phân tán dịch truy vấn toàn cục thành các nhóm truy vấn cục bộ và được thực hiện bởi các thành phần quản trị CSDL phân tán tại các trạm khác nhau và giữa các trạm giao tiếp với nhau.

Mô hình đang xét là mô hình ANSI/SPARC được mở rộng bằng cách thêm vào từ điển/thư mục toàn cục GD/D (Glocal Directory/Directionary) cho phép ánh xạ yêu cầu toàn cục. Ánh xạ cục bộ được thực hiện bởi từ điển/thư mục cục bộ LD/D (Local Directory/Directionary). Vì vậy, các thành phần quản lý CSDL cục bộ được tích hợp thành các chức năng của hệ quản trị CSDL toàn cục .

Trong hình 1.16 lược đồ khái niệm cục bộ ánh xạ đến lược đồ trong tại mỗi vị trí. Lược đồ khái nhiệm toàn cục ánh xạ vào lược đồ khái niệm cục bộ. Tất cả các định nghĩa khung nhìn của mô hình ngoài đều có phạm vi toàn cục. Các ánh xạ nó đảm bảo cho tính trong suốt của cơ sở dữ liệu phân tán và tính độc lập của cơ sở phân tán.



Hình 1.16 sơ đồ chức năng của hệ quản trị CSDL phân tán tích hợp



Hình 1.17 Các thành phần của một hệ quản trị CSDL phân tán.

Một hệ DBMS phân tán gồm 2 phần như trong hình 2.19. Bộ xử lý phía người sử dụng (User Procesor), xử ly tất cả tương tác với người sử dụng và bộ phận thứ 2 của DBMS phân tán là bộ phận xử lý dữ liệu (Data Processor). Bộ xử lý phía người sử dụng bao gồm:

1. *Bộ xử lý giao diện người sử dụng*: Có trách nhiệm dịch các lệnh của người sử dụng khi họ gửi đến và định dạng dữ liệu kết quả để gửi nó lại cho người sử dụng .

2. *Bộ kiểm soát dữ liệu ngữ nghĩa*: sử dụng ràng buộc toàn vẹn và xác thực, được định nghĩa như là một phần của lược đồ khái niệm cục bộ, để kiểm tra xem truy vấn của người sử dụng có được xử lý hay không. Thành phần này cũng có trách nhiệm xác thực và một số chức năng khác.
3. *Bộ phân rã và bộ tối ưu hoá truy vấn toàn cục* xác định chiến lược thực thi để giảm thiểu chức năng chi phí, và dịch các truy vấn toàn cục ra thành các truy vấn cục bộ bằng cách sử dụng các lược đồ khái niệm cục bộ, toàn cục và thư mục toàn cục. Bộ tối ưu hoá truy vấn có trách nhiệm tạo ra chiến lược thực thi các hoạt động kết nối phân tán.
4. *Bộ giám sát thực thi phân tán phối hợp* thực thi phân tán yêu cầu của người sử dụng .Bộ giám sát thực thi cũng được gọi là bộ quản lý giao dịch phân tán. Việc thực thi truy vấn trong hệ phân tán, bộ giám sát thực thi tại một số trạm có thể, và thường, liên lạc với một bộ giám sát thực thi khác.

Phần thứ hai của hệ quản trị CSDL phân tán là bộ xử lý dữ liệu gồm ba thành phần:

1. *Bộ tối ưu hoá truy vấn cục bộ* hoạt động như là bộ chọn đường dẫn truy nhập. Chọn đường truy nhập tốt nhất vào bất kỳ mục dữ liệu nào.
2. *Bộ quản lý khôi phục cục bộ* có trách nhiệm đảm bảo duy trì tính nhất quán trong CSDL cục bộ ngay cả khi có lỗi xảy ra.
3. *Bộ hỗ trợ thời gian thực thi* truy nhập vào CSDL tùy vào các lệnh trong lịch biểu được tạo ra bởi bộ tối ưu hóa truy vấn. Bộ xử lý hỗ trợ thời gian thực thi là giao diện với hệ điều hành và chứa bộ quản lý vùng đệm CSDL (buffer hoặc cache), có trách nhiệm quản lý vùng đệm của bộ nhớ chính và quản lý việc truy nhập dữ liệu.

1.14 KIẾN TRÚC TỔNG QUAN CỦA MỘT HỆ QUẢN TRỊ PHỨC HỆ CSDL PHÂN TÁN (Multi Database Management System)

1.14.1 Mô hình kiến truc tổng quan của một phức hệ

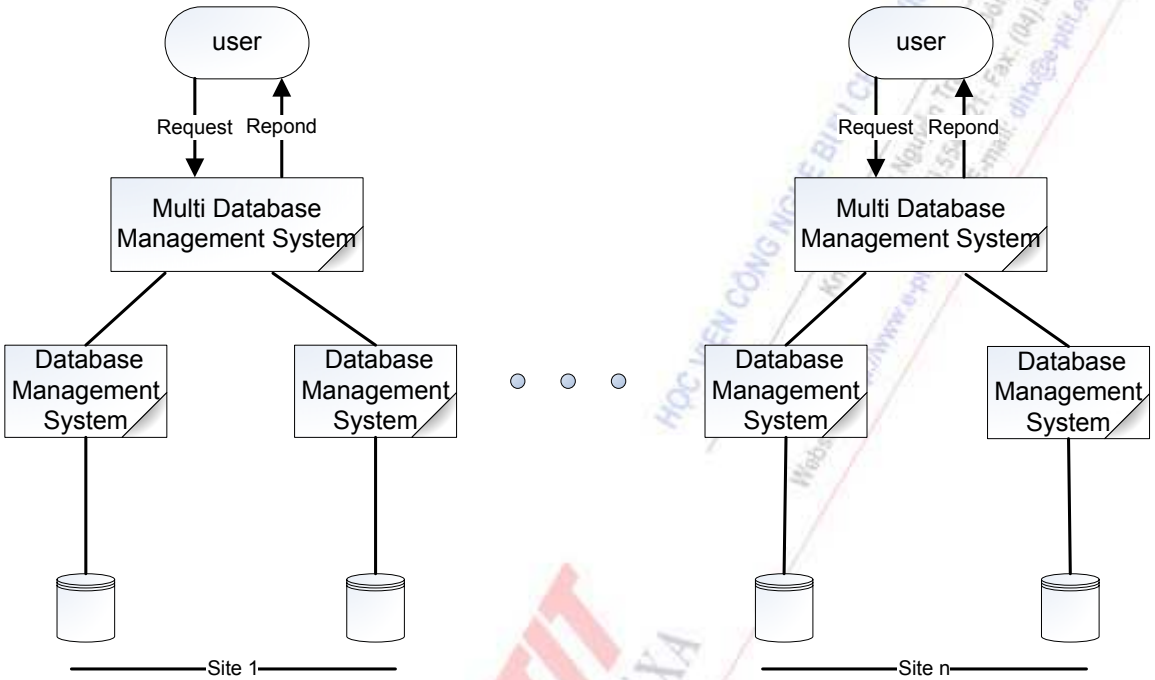
Một phức hệ phân tán bao gồm nhiều phức hệ quản trị được cài đặt tại nhiều vị trí (site) khác nhau. Mỗi phức hệ tại mỗi vị trí bao gồm hai thành phần chính đó là

- Hệ quản trị dữ liệu DBMS (Database Management System) là các hệ quản trị cơ sơ dữ liệu ở mức thấp nhất của hệ thống. Nó có chức năng tổ chức lưu trữ và thực hiện các thao tác vấn tin được chuyển cho nó và trả về kết quả cho hệ thống
- Các phức hệ quản trị (Multi-DataBase Management System) tại mỗi site, các phức hệ này có chức năng quản lý các DBMS thành phần trong hệ thống và các mối tương tác qua lại giưa chúng. Đồng thời nhận và và xử lý các câu vấn tin trước khi chuyển giao cho các DBMS thành phần.

Hình 1.18 mô tả mô hình kiến trúc của một phức hệ.

1.14.2 Phân loại các phức hệ dựa vào cấu trúc

Hệ quản trị phức hệ CSDL phân tán khác với hệ quản trị CSDL phân tán được phản ánh trong định nghĩa lược đồ khái niệm toàn cục. Trong các hệ quản trị CSDL phân tán tích hợp logic, lược đồ khái niệm toàn cục là hợp của các CSDL cục bộ, trong khi đó ở hệ quản trị phức hệ CSDL phân tán chỉ định nghĩa một tập con gồm một số CSDL cục bộ mà các hệ quản trị CSDL cục bộ chia sẻ. Vì vậy định nghĩa CSDL toàn cục đa CSDL có khác với định nghĩa CSDL toàn cục trong hệ quản trị CSDL phân tán.



Hình 1.18 Mô hình kiến truc tổng quan của một phức hệ phân tán

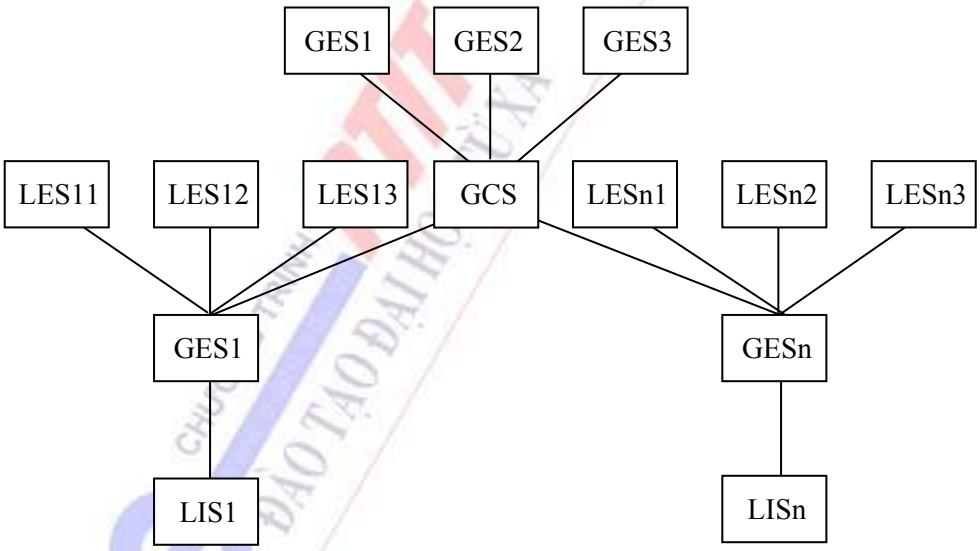
a) Các mô hình sử dụng lược đồ khái niệm toàn cục: Trong một hệ đa CSDL, lược đồ khái niệm toàn cục GCS được định nghĩa bằng cách tích hợp các lược đồ ngoài của các CSDL tự vận hành cục bộ hoặc các thành phần của lược đồ khái niệm cục bộ của chúng. Mặt khác, Vì hệ thống là tự trị, người sử dụng của một DBMS cục bộ sẽ định nghĩa khung nhìn riêng của họ trên CSDL cục bộ và không cần phải thay đổi các ứng dụng khi truy xuất vào các CSDL cục bộ khác.

Việc thiết kế lược đồ khái niệm toàn cục GCS trong phức hệ CSDL thường được thực hiện từ dưới lên (Bottom-Up), ánh xạ đi từ lược đồ khái niệm cục bộ đến lược đồ toàn cục (trong khi quản trị CSDL phân tán, việc thiết kế lược đồ khái niệm toàn cục GCS thường được thiết kế từ trên xuống (Top-Down), ánh xạ lại theo hướng ngược lại). Khung nhìn của người sử dụng trên lược đồ quan hệ toàn cục được định nghĩa theo yêu cầu truy nhập toàn cục. Các lược đồ ngoài toàn cục GES (Global External Schema) và lược đồ khái niện toàn cục

GCS không nhất thiết phải sử dụng theo cùng một mô hình và ngôn ngữ dữ liệu, không cần xác định hệ thống là thuần nhất hay hỗn hợp.

Nếu hệ thống là hỗn hợp, thì có hai lựa chọn cài đặt: đơn ngữ (Unilingual) và đa ngữ (Multilingual).

- Hệ quản trị phức hệ CSDL đơn ngữ cho phép sử dụng các mô hình và ngôn ngữ dữ liệu khác nhau khi truy xuất CSDL cục bộ và CSDL toàn cục. Bất kỳ ứng dụng nào truy xuất dữ liệu từ các phức hệ CSDL đều phải thực hiện qua một khung nhìn đã được định nghĩa trong lược đồ khái niệm toàn cục, nghĩa là truy xuất CSDL toàn cục khác với truy xuất CSDL cục bộ. Vì vậy, phải định nghĩa *lược đồ ngoài cục bộ* LES (Local External Schema) trên lược đồ khái niệm cục bộ và *lược đồ ngoài toàn cục* GES trên lược đồ khái niệm toàn cục. Khung nhìn ngoài khác nhau cũng có thể sử dụng các ngôn ngữ truy nhập khác nhau. Hình 2.21 mô tả mô hình logic của hệ CSDL đơn ngữ tích hợp với lược đồ khái niệm cục bộ trong lược đồ khái niệm toàn cục.
- Kiến trúc đa ngữ (Multilingual) cho phép người sử dụng truy nhập tới CSDL toàn cục bằng lược đồ ngoài được định nghĩa bởi ngôn ngữ của hệ quản trị CSDL cục bộ. Định nghĩa GCS trong kiến trúc đơn ngữ và đa ngữ giống nhau. Tuy nhiên định nghĩa các lược đồ ngoài trong kiến trúc đa ngữ được mô tả bằng ngôn ngữ của lược đồ ngoài của CSDL cục bộ. Truy vấn được xử lý một cách chính xác như truy vấn trong các hệ quản trị CSDL tập trung. Truy vấn CSDL toàn cục được sử dụng ngôn ngữ của hệ quản trị CSDL.Người sử dụng truy vấn CSDL dễ dàng hơn với cách tiếp cận đa ngữ .



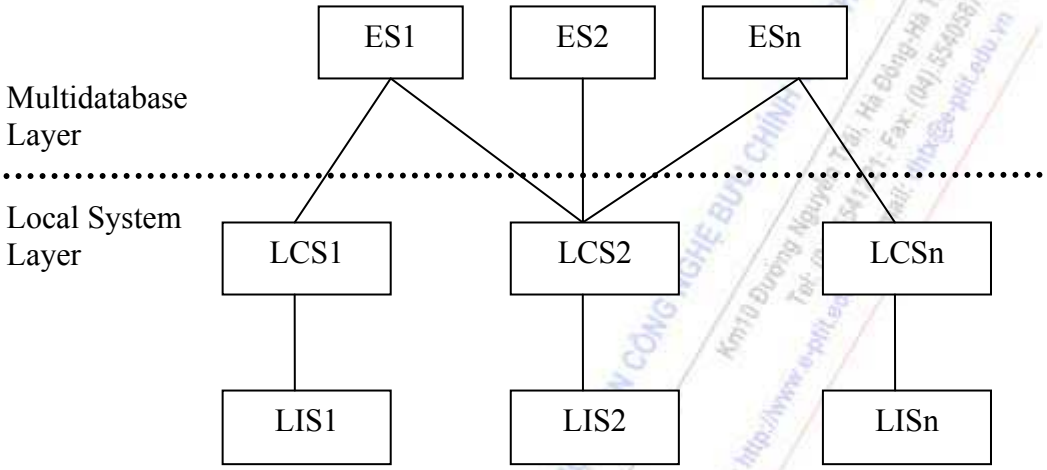
Hình 1.19 kiến trúc phức hệ CSDL với một lược đồ khái niệm toàn cục.

1.14.3 Các mô hình không sử dụng lược đồ khái niệm toàn cục

Một hệ thống có nhiều CSDL nhưng không có lược đồ toàn cục có nhiều ưu điểm hơn, so với hệ thống có lược đồ khái niệm toàn cục. Hình 2.22. mô tả kiến trúc của một mô hình gồm hai tầng: tầng hệ thống cục bộ và tầng phức hệ CSDL. Tầng hệ thống cục bộ gồm một số hệ quản trị CSDL với chức năng là trình bày cho tầng phức hệ CSDL các phần của CSDL cục bộ dùng chung bởi nhiều người sử dụng các CSDL khác. Dữ liệu dùng chung được trình bày bởi

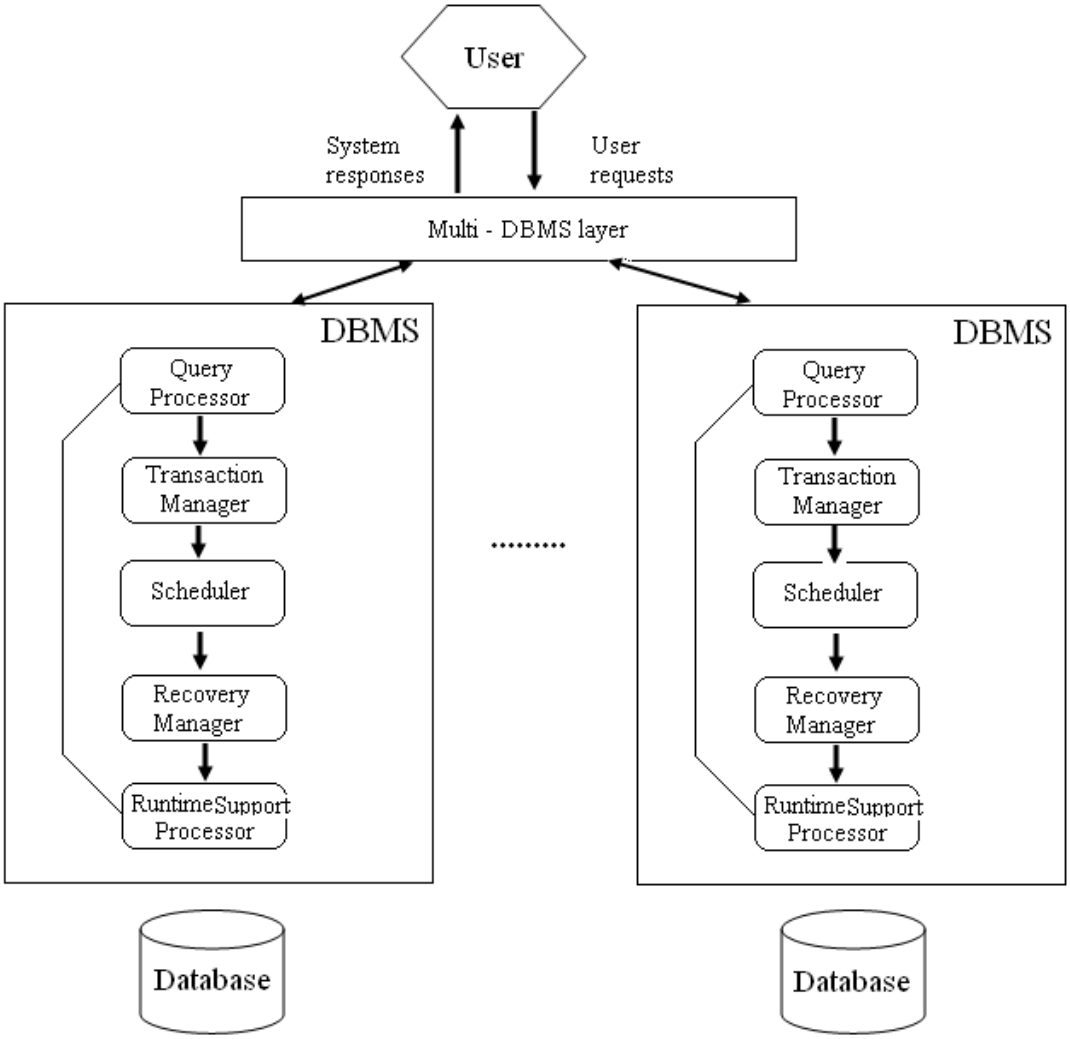
lược đồ khái niệm cục bộ hoặc qua một định nghĩa lược đồ ngoài cục bộ. Tầng này được trình bày bằng một tập các lược đồ khái niệm cục bộ LCS. Nếu hỗn hợp, mỗi lược đồ LCS có thể sử dụng mỗi mô hình dữ liệu khác nhau.

Trên tầng hệ thống cục bộ là tầng hệ đa CSDL, bao gồm các khung nhìn được định nghĩa trên một hay nhiều lược đồ khái niệm cục bộ. Vì vậy có thể cấp quyền truy xuất vào nhiều CSDL bởi ánh xạ giữa khung nhìn và lược đồ khái niệm cục bộ



Hình 1.20 Kiến trúc phức hệ CSDL không sử dụng GCS

Mô hình kiến trúc dựa trên các thành phần của hệ đa CSDL khác biệt nhiều so với một hệ quản trị CSDL phân tán. Khác biệt cơ bản là sự tồn tại của các DBMS hoàn chỉnh quản lý từng CSDL khác nhau. Đa CSDL cung cấp một tầng phần mềm chạy trên những DBMS riêng biệt này và cung cấp cho người dùng những tiện ích để truy xuất nhiều CSDL khác nhau. Tuỳ thuộc vào vấn đề có lược đồ khái niệm toàn cục hoặc là có vấn đề đa chủng hay không, nội dung của tầng phần mềm sẽ thay đổi. hinh 4.10 trình bày một hệ quản trị đa CSDL phi phân tán. nếu là hệ phân tán, chúng ta cần phải sao chép tầng đa CSDL hco mỗi vị trí có hệ quản trị CSDL cục bộ tham gia vào hệ thống. cũng cần chú ý rằng nếu xem xét từng DBMS riêng biệt, tầng phức hệ CSDL chỉ đơn giản là một ứng dụng có nhiệm vụ “đệ trình” các yêu cầu và nhận kết quả trả lời.



Hình 1.21 Các thành phần của một phức hệ CSDL

CÂU HỎI TRẮC NGHIỆM

1. Cơ sở dữ liệu phân tán là sự:
    A. Hợp nhất lý thuyết cơ sở dữ liệu và công nghệ mạng máy tính.
    B. Hợp nhất công nghệ viễn thông và tin học.
    C. Tích hợp công nghệ tin học và cơ sở dữ liệu.
2. Khái niệm hệ cơ sở dữ liệu phân tán bao gồm khái niệm về:
    A. Cơ sở dữ liệu phân tán và công nghệ mạng máy tính.
    B. Cơ sở dữ liệu tập trung và tối ưu hoá câu hỏi
    C. Cơ sở dữ liệu phân tán và hệ quản trị cơ sở dữ liệu phân tán.
3. Cơ sở dữ liệu phân tán là:

A. Một tập các cơ sở dữ liệu có quan hệ với nhau về mặt logic và được phân tán trên một mạng máy tính.

B. Một tập các cơ sở dữ liệu được phân tán trên một mạng máy tính.

C. Một tập các cơ sở dữ liệu được cài đặt lưu trữ trên các máy chủ.

4. Hệ quản trị cơ sở dữ liệu phân tán là:

A. Hệ thống phần mềm quản trị cơ sở dữ liệu phân tán và làm cho sự phân tán đó là trong suốt đối với người sử dụng.

B. Hệ thống phần mềm điều khiển truy nhập cơ sở dữ liệu phân tán .

C. Hệ thống phần mềm thực hiện các phép lưu trữ và tìm kiếm dữ liệu trên mạng máy tính.

5. Đặc trưng của cơ sở dữ liệu phân tán là:

A. Dữ liệu được phân tán trên mạng máy tính và có quan hệ logic. với nhau

B. Tập các file dữ liệu được lưu trữ trên các thiết bị nhớ của mạng máy tính.

C. Tập các file dữ liệu có quan hệ với nhau về mặt logic

6. Độc lập dữ liệu được hiểu là:

A. Tổ chức lưu trữ dữ liệu là trong suốt đối với người sử dụng dụng.

B. Các chương trình ứng dụng không phụ thuộc vào tổ chức lưu trữ dữ liệu.

C. Tổ chức lưu trữ dữ liệu trên các máy chủ của mạng

7. Đặc trưng về độc lập dữ liệu trong các hệ cơ sở dữ liệu phân tán là:

A. Sự trong suốt phân tán

B. Các ứng dụng được phân tán.

C. Cơ sở dữ liệu tổ chức lưu trữ tập trung

8. Mô hình cơ sở dữ liệu tập trung:

A. Ứng dụng, hệ quản trị CSDL và CSDL được cài đặt trên cùng một bộ xử lý.

B. Ứng dụng, hệ quản trị CSDL cài đặt khác hệ thống máy tính với CSDL

C. Ứng dụng, hệ quản trị CSDL và CSDL được cài đặt trên các vị trí khác nhau

9. Mô hình cơ sở dữ liệu Client/Server

A. CSDL được cài đặt trên Server, các ứng dụng trên các máy Client và phần mềm cơ sở dữ liệu được cài đặt trên cả Client lẫn Server.

B. CSDL được cài đặt trên Server, các ứng dụng trên các máy Client

C. CSDL, các ứng dụng và hệ quản trị CSDL được cài đặt trên Server

10. Mô hình kiến trúc cơ sở dữ liệu phân tán tại các site gồm:

A. Lược đồ tổng thể, lược đồ phân mảnh và lược đồ cấp phát.

B. Lược đồ khái niệm, lược đồ quan hệ và lược đồ cấp phát.

C. Lược đồ tổng thể, lược đồ cục bộ và các chiến lược truy nhập.

11. Lược đồ toàn cục trong cơ sở dữ liệu phân tán được định nghĩa như:

A. Lược đồ cơ sở dữ liệu quan hệ

B. Lược đồ khái niệm.

C. Lược đồ cục bộ

12. Khái niệm phân mảnh cơ sở dữ liệu được hiểu là:

A. Quan hệ toàn cục có thể chia thành nhiều mảnh không chồng lặp

B. Các quan hệ được cài đặt trên các site khác nhau.

C. Các quan hệ được sao chép và cài đặt trên các site khác nhau.

13. Khái niệm cấp phát trong cơ sở dữ liệu phân tán được hiểu là:

A. Phương pháp cài đặt các bản sao, phân mảnh trên mạng máy tính

B. Phương pháp phân mảnh dữ liệu

C. Phương pháp cài đặt cơ sở dữ liệu trên mạng máy tính

14. Các kiểu thiết kế cơ sở dữ liệu phân tán trên mạng máy tính.

D. Bản sao, phân mảnh và kết hợp bản sao và phân mảnh.

E. Bản sao và phân mảnh

F. Phân mảnh

15. Tính tự trị địa phương, nghĩa là

A. Dữ liệu được chia sẻ bởi một nhóm người sử dụng, kiểm soát cục bộ.

B. Dữ liệu được phân tán trên nhiều vị trí, kiểm soát toàn cục.

C. Dữ liệu lưu trữ tập trung.

16. Tính song song trong các hệ cơ sở dữ liệu phân tán nghĩa là:

A. Các câu hỏi truyền về vị trí chính và xử lý.

B. Xử lý đồng thời các câu hỏi tại các vị trí khác nhau.

C. Câu hỏi phân rã thành các câu hỏi thành phần, thực hiện song song tại các vị trí khác nhau.

17. Một trong những ưu điểm cơ bản của tổ chức dữ liệu phân tán là:

A. Độ tin cậy và tính sẵn sàng được nâng cao.

B. Đảm bảo an toàn cho việc truy nhập cơ sở dữ liệu khi có sự cố xảy, không thể làm sụp đổ cả hệ thống.

C. Nâng cao hiệu quả

18. Tổ chức dữ liệu phân tán kinh tế hơn so với tổ chức tập trung, vì:

A. Giá cho một hệ máy tính nhỏ rẻ hơn

B. Hiệu quả hơn khi triển khai cùng một mục đích ứng dụng.

C. Giá chi phí truyền thông thấp hơn.

19. Ưu điểm cách tiếp cận mô hình cơ sở dữ liệu quan hệ

A. Tính đơn giản, tính độc lập dữ liệu, đối xứng, cơ sở lý thuyết vững chắc.

B. Tính đơn giản, tính độc lập dữ liệu và toàn vẹn dữ liệu.

C. Tính độc lập dữ liệu và toàn vẹn dữ liệu.

20. Quy tắc toàn vẹn dữ liệu (Integrity Rule) là các quy tắc:

A. Ràng buộc các trạng thái nhất quán của dữ liệu.

B. Ràng buộc cấu trúc và các ràng buộc về hành vi.

C. Các mối quan hệ, ràng buộc lẫn nhau trong cơ sở dữ liệu

21. Kiến trúc tổng quát của một hệ quản trị cơ sở dữ liệu quan hệ là:

A. Một hệ thống phần mềm

B. Các hệ thống truyền thông và hệ điều hành

C. Các hệ thống các ứng dụng.

22. Các tầng chức năng kiến trúc của một hệ QTCSDL quan hệ gồm:

A. Tầng giao diện, điều khiển, xử lý vấn tin, thực thi, truy xuất dữ liệu và tầng duy trì nhất quán.

B. Tầng ứng dụng, trình bày, giao vận, mạng, liên kết dữ liệu và tầng vật lý.

C. Tầng điều khiển, xử lý vấn tin, và tầng truy xuất dữ liệu.

23. Hệ quản trị CSDL phân tán thuần nhất, nghĩa là

A. CSDL phân mảnh thành các CSDL cục bộ (Local) và được quản lý bởi cùng một hệ QTCSDL.

B. CSDL toàn cục bộ được quản lý bởi một hệ QTCSDL.

C. CSDL phân tán trên các vị trí khác nhau được quản lý bởi nhiều hệ QTCSDL.khác nhau

24. Hệ quản trị CSDL phân tán không thuần nhất:

A. CSDL cục bộ được quản lý bởi các hệ QTCSDL khác nhau.

B. CSDL toàn cục bộ được quản lý bởi một hệ QTCSDL.

C. CSDL phân mảnh thành các CSDL cục bộ (Local)

25. Đặc tính hệ quản trị cơ sở dữ liệu phân tán

A. Client Server, ngang hàng và hệ đa CSDL.

B. Tự trị, phân tán, hỗn hợp.

C. Đơn quan hệ và đa CSDL.

CHƯƠNG II: THIẾT KẾ CÁC HỆ CSDL PHÂN TÁN

Trong chương này sẽ trình bày những khái niệm cơ bản về các phương pháp phân mảnh dữ liệu và bài toán cấp phát dữ liệu trên các vị trí của mạng máy tính. Nội dung của chương bao gồm các phần:

- Các vấn đề về phân mảnh dữ liệu
- Phương pháp phân mảnh ngang
- Phân mảnh ngang dẫn xuất
- Phân mảnh dọc
- Phương pháp phân mảnh hỗn hợp(Hybrid Fragmentation)
- Cấp phát và mô hình cấp phát

2.1 CÁC VẤN ĐỀ VỀ PHÂN MẢNH DỮ LIỆU

Phần lớn các hệ cơ sở dữ liệu phân tán được thiết kế theo hướng từ trên xuống (Top-Down). Thiết kế phân mảnh dữ liệu là công việc đầu tiên phải thực hiện. Mục đích của việc phân mảnh dữ liệu là tạo ra các đơn vị cấp phát logic, sao cho chi phí để thực hiện truy vấn thông tin là thấp nhất. Các bộ hoặc các thuộc tính của quan hệ không thể được xem như một đơn vị cấp phát, vì sẽ làm cho việc cấp phát trở lên phức tạp hơn. Thiết kế phân mảnh bằng cách nhóm một số bộ trong trường hợp phân mảnh ngang hay nhóm các thuộc tính trong trường hợp phân mảnh dọc có cùng đặc tính theo quan điểm cấp phát. Các mảnh hình thành bằng các phương pháp phân mảnh tạo ra các đơn vị cấp phát dữ liệu khác nhau.

2.1.1 Lý do phân mảnh

Trong thiết kế CSDL phân tán , cần thiết phải thực hiện phân mảnh dữ liệu vì những lý do sau đây:

- Trong các hệ quản trị CSDL, các quan hệ được lưu trữ dưới dạng các bảng 2 chiều. Các thao tác đối với CSDL được thực hiện trên các bảng. Tuy nhiên trong thực tế, các ứng dụng chỉ yêu cầu thao tác trên các tập con của các quan hệ, là khung nhìn dữ liệu của người sử dụng. Vì vậy việc xem tập con của quan hệ là đơn vị truy xuất thông tin để phân tán dữ liệu là hợp lý.
- Việc phân rã một quan hệ thành nhiều mảnh, mỗi mảnh được xử lý như một đơn vị dữ liệu, sẽ cho phép thực hiện nhiều giao dịch đồng thời. Đồng thời việc phân mảnh các quan hệ cũng cho phép thực hiện song song một câu vấn tin bằng cách chia nó thành một tập các câu vấn tin con hoạt tác trên các mảnh. Vì thế việc phân mảnh sẽ làm tăng mức độ hoạt động đồng thời và tăng lưu lượng hoạt động của hệ thống.

Tuy nhiên không phải việc phân mảnh chỉ có ưu điểm hoàn toàn, mà nó cũng thể hiện những hạn chế nhất định như:

- Nếu ứng dụng có những yêu cầu "xung đột" ngăn cản phân rã thành các mảnh được sử dụng độc quyền.

- Những ứng dụng có các khung nhìn được định nghĩa trên nhiều mảnh khác nhau sẽ làm giảm hiệu suất hoạt động của hệ thống, làm tăng chi phí truy xuất dữ liệu đến các mảnh và tăng chi phí kết nối các mảnh
- Việc kiểm soát ngữ nghĩa, đặc biệt là vấn đề kiểm tra tính toàn vẹn sẽ khó khăn hơn

2.1.2 Các kiểu phân mảnh

Các quan hệ cơ sở dữ liệu thường được biểu diễn dưới dạng bảng. Việc phân mảnh một quan hệ thành nhiều quan hệ con khác nhau theo các cách khác nhau, sẽ có các cách phân mảnh tương ứng. Có hai kiểu phân mảnh tương ứng với việc chia quan hệ theo chiều dọc và chia quan hệ theo chiều ngang .

Phân mảnh theo chiều dọc: Các quan hệ được chia theo chiều dọc. Nghĩa là thiết lập một quan hệ mới chỉ có một số thuộc tính từ quan hệ gốc. Thực chất đây là phép chiếu trên tập con các thuộc tính của quan hệ.

Ví dụ 2.1 Tách dọc quan hệ PROJ thành 2 quan hệ PROJ1 và PROJ2 như sau:

$\pi_{PNO,BUDGET}$ (PROJ) và $\pi_{PNO,PNAME,LOG}$ (PROJ) và

PROJ

PNO	BUDGET
P1	150000
P2	135000
P3	250000
P4	310000

PROJ

PNO	PNAME	LOG
P1	Instrumentation	Montreal
P2	Database Develop	New York
P3	CAD/CAM	New York
P4	Maintenance	Paris

Hình 2.1. Phân mảnh dọc

Phân mảnh ngang: Quan hệ được chia theo chiều ngang. Thực chất đây là phép chọn trong quan hệ. Chọn những bộ của quan hệ thỏa mãn một biểu thức điều kiên cho trước.

Ví dụ 2.2 Tách ngang quan hệ PROJ thành 2 quan hệ PROJ1 và PROJ2 thoả theo điều kiện: BUDGET $\leq$ 200000 và BUDGET $>$ 200000 như sau:

$\sigma_{BUDGET \leq 200000}$ (PROJ)

$\sigma_{BUDGET > 200000}$ (PROJ)

PROJ1

PNO	PNAME	BUDGET
P1	Instrumentation	150000
P2	Database Develop	135000

PROJ2

PNO	PNAME	BUDGET
P3	CAD/CAM	250000
P4	Maintenance	310000

Hình 2.2. Phân mảnh ngang

2.1.3 Mức độ phân mảnh

Phân mảnh cơ sở dữ liệu đến mức độ nào là đủ, không làm ảnh hưởng đến hiệu năng của việc thực hiện truy vấn. Mức độ phân mảnh có thể là phân mảnh một quan hệ chưa được phân mảnh, có thể phân mản các quan hệ đã được phân mảnh. Có thể phân mảnh theo chiều dọc (theo từng thuộc tính) hoặc theo chiều ngang (theo từng bộ trong quan hệ).

Một mức độ ứng phân mảnh thích hợp sao cho tránh được các hạn chế khi phân mảnh chỉ được định nghĩa ứng với các ứng dụng sẽ chạy trên cơ sở dữ liệu.

2.1.4 Các quy tắc phân mảnh

Các nguyên tắc để đảm bảo cơ sở dữ liệu khi phân mảnh sẽ đảm bảo tính không thay đổi về ngữ nghĩa. Dưới đây là ba qui tắc phải tuân thủ khi phân mảnh cơ sở dữ liệu quan hệ.

1. *Tính đầy đủ:* Quan hệ R được phân rã thành các mảnh R_1, R_2…R_n, thì mỗi mục dữ liệu có trong quan hệ R sẽ được chứa trong ít nhất một mảnh R_i (i=1,..,n). Quy tắc này đảm bảo cho các mục dữ liệu trong R được ánh xạ hoàn toàn vào các mảnh và không bị mất. Mục dữ liệu có thể hiểu là bộ trong phân mảnh ngang và thuộc tính trong phân mảnh dọc.
2. *Tính phục hồi:* Nếu một quan hệ R được phân rã thành các mảnh R_1, R_2,…,R_n khi đó: $R = \nabla R_i, \ \forall Ri \in F_R$. Toán tử ∇ thay đổi tùy theo từng loại phân mảnh.. Khả năng phục hồi quan hệ từ các mảnh sẽ đảm bảo bảo toàn các phụ thuộc.
3. *Tính tách biệt:* Nếu quan hệ R được phân rã ngang thành các mảnh Ri, i=1,..n và mục dữ liệu d_i nằm trong một mảnh R_i thì nó sẽ không nằm trong mảnh R_k , ($k \neq j$). Quy tắc này đảm bảo các mảnh phân rã rời nhau. Trong trường hợp phân mảnh dọc, khóa chính của quan hệ phải được lập lại trong tất cả các mảnh. Vì vậy tính tách biệt trong phân mảnh dọc được hiểu không liên quan gì đến khóa chính của quan hệ.

2.1.5 Các kiểu cấp phát

Giả sử CSDL đã được phân mảnh, thích hợp và thoả các yêu cầu phải cấp phát cho các vị trí trên mạng. Khi dữ liệu được cấp phát, có thể không nhân bản hoặc có thể được nhân bản. Không nhân bản, thường được gọi là CSDL phân hoạch, các mảnh chỉ được cấp phát trên các trạm và không có bản sao nào trên mạng. Trong trường hợp nhân bản, hoặc toàn bộ CSDL đều có ở trên tất cả các từng trạm (CSDL được nhân bản đầy đủ), hoặc các mảnh của CSDL được phân tán tới các trạm bằng cách các mảnh sao được đặt trên nhiều trạm (CSDL được nhân bản từng phần). Một số các bản sao của mảnh có thể là đầu vào cho thuật toán cấp phát hoặc quyết định giá trị của biến được xác bởi thuật toán.

Nhân bản làm tăng độ tin cậy và tăng hiệu quả của các câu vấn tin chỉ đọc (Read-only Query).. Đặc biệt có thể truy xuất CSDL khi gặp sự cố. Hơn nữa các câu truy vấn đọc truy xuất đến cùng một mục dữ liệu có thể cho thực hiện song song vì các bản sao có mặt tại nhiều vị trí. Hình 2.3 so sánh ba cách nhân bản theo một số chức năng của hệ quản trị CSDL phân tán.

Việc cấp phát dữ liệu phải được thực hiện sao cho thỏa mãn hai yêu cầu sau:

- Chi phí nhỏ nhất
- Hiệu năng lớn nhất: Giảm thiểu thời gian đáp ứng và tăng tối đa lưu lượng hệ thống tại mỗi vị trí.

	Nhân bản hoàn toàn	Nhân bản một phần	Phân hoạch
Xử lý truy vấn	Dễ	Cùng mức độ khó khăn	
Quản lý thư mục	Dễ hoặc không tồn tại	Cùng mức độ khó khăn	
Điều khiển đồng thời	Vừa phải	Khó	Dễ
Độ tin cậy	Rất cao	Cao	Thấp
Tính thực tế	Có thể áp dụng	Thực tế	Có thể áp dụng

Hình 2.3 So sánh các phương pháp nhân bản

2.1.6 Các yêu cầu thông tin

Công việc thiết kế cơ sở dữ liệu phân tán phụ thuộc rất nhiều vào các yếu tố có là ảnh hưởng đến một thiết kế tối ưu, tổ chức logic cơ sở dữ liệu, vị trí các ứng dụng, đặc tính truy xuất của các ứng dụng đến cơ sở dữ liệu và các đặc tính của hệ thống máy tính tại mỗi vị trí. Điều này làm cho việc diễn đạt bài toán phân tán trở nên hết sức phức tạp.

Các thông tin cần thiết cho thiết kế phân tán bao gồm:

- Thông tin cơ sở dữ liệu
- Thông tin ứng dụng
- Thông tin về mạng
- Thông tin về hệ thống máy tính

Yêu cầu thông tin về mạng và thông tin về hệ thống máy tính chỉ được sử dụng trong các mô hình cấp phát, không sử dụng trong các thuật toán phân mảnh dữ liệu

2.2 PHƯƠNG PHÁP PHÂN MẢNH NGANG

2.2.1 Giới thiệu

Phân mảnh ngang chính là việc chia quan hệ thành nhiều các nhóm bộ. Kết quả của quá trình phân mảnh ngang là các quan hệ con, số lượng quan hệ con phụ thuộc vào điều kiện ràng buộc của các thuộc tính. Và các bộ trong các quan hệ con là tách biệt nhau. Phân mảnh ngang thực chất là phép chọn quan hệ thỏa mãn một biểu thức điều kiên cho trước.

Có hai loại phương pháp phân mảnh ngang là:

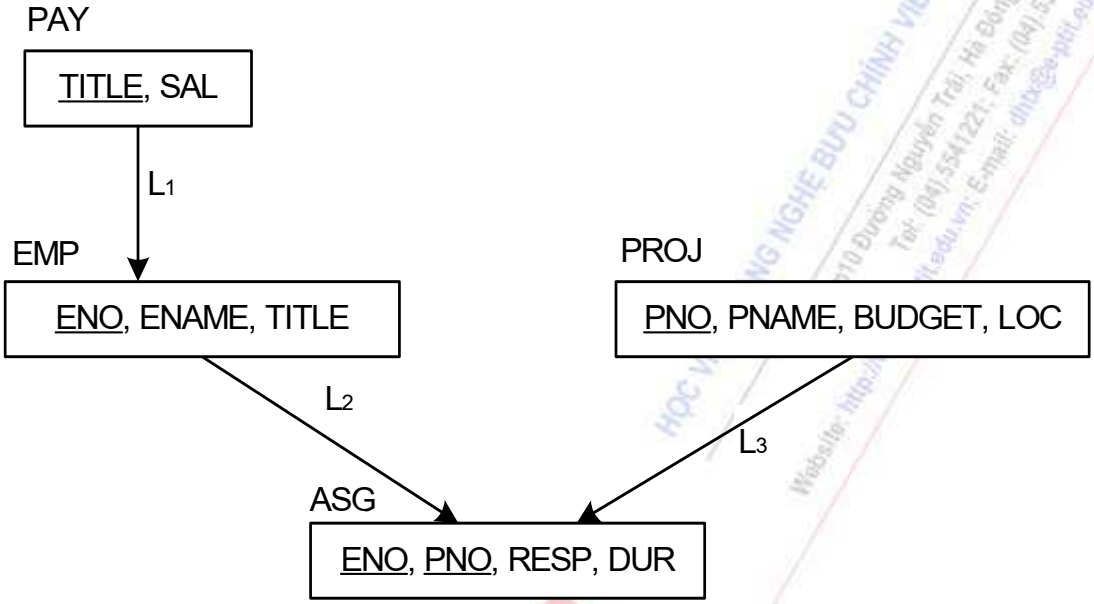
- Phân mảnh ngang nguyên thủy: Là phân mảnh ngang được thực hiện trên các vị từ của chính quan hệ đó.
- Phân mảnh ngang dẫn xuất: Là phân rã một quan hệ dựa trên các vị từ của quan hệ khác.

2.2.2 Thông tin cần thiết của phân mảnh ngang

a) Thông tin về CSDL có liên quan tới lược đồ khái niệm toàn cục. Trong mô hình quan hệ, các mối quan hệ giữa các thực thể được mô tả như là những quan hệ. Trong mô hình quan

hệ thực thể (ER), các mối liên hệ giữa các đối tượng CSDL được mô tả rõ ràng. Nhìn chung mối quan hệ giữa các đối tượng trong CSDL thường mô tả bằng các mối quan hệ một - một, một - nhiều và mối quan hệ nhiều - nhiều. Với mục đích cho thiết kế ,đường nối (Link) có hướng giữa các quan hệ được sử dụng cho việc biểu diễn bởi thao tác nối bằng (Equijoin).

Ví dụ 2.3: Trong hình 2.4, mỗi một chức vụ (Title) có nhiều nhiều nhân viên (Employee) giữ chức vụ đó. Đây là mối quan hệ môt - nhiều được biểu diễn bằng một đường nối có hướng L1 trỏ từ quan hệ PAY đến EMP. Mối quan hệ nhiều - nhiều được trỏ từ các quan hệ EMP và PROJ đến quan hệ ASG.được biểu diễn bằng hai đường nối L2 và L2.



Hình 2.4 Mô tả mối quan hệ giữa các quan hệ bởi các đường nối

Quan hệ tại điểm cuối của đường nối được gọi là quan hệ chủ (quan hệ đích) và các quan hệ tại điểm đầu được gọi là các quan hệ thành viên (quan hệ nguồn). Ánh xạ Owner và Member từ tập đường nối tới tập quan hệ. Khi cho trước một đường nối, hàm sẽ trả về quan hệ đích hay quan hệ nguồn của đường nối. Ví dụ trong hình 2.4: owner(L_1) = PAY và member(L_1) = EMP

Ký hiệu lực lượng (cardinality) của mỗi quan hệ R là Card(R).

b) Thông tin về ứng dụng: Để thực hiện phân mảnh, cần phải có thông tin định tính và thông tin định lượng. Thông tin định tính hướng dẫn cho hoạt động phân mảnh, thông tin định lượng chủ yếu sử dụng trong các mô hình cấp phát.

Thông tin định tính cơ bản gồm các vị từ dùng trong câu truy vấn. Sau đây là các định nghĩa về vị từ đơn giản (Simple Predicate) và vị từ hội sơ cấp (Minterm Predicate) như sau:

- Cho quan hệ R(A_1, A_2,..,A_n), trong đó A_i là thuộc tính được định nghĩa trên một miền biến thiên D_i , một vị từ đơn giản P_j được định nghĩa trên R có dạng:

  P_j: $A_i \theta$ *Value*

Trong đó θ thuộc {=, <, ≠, ≤, >, ≥} và Value được chọn từ miền A_i (Value thuộc Di). Chúng ta sử dụng Pr_i để biểu thị tập tất cả các vị từ đơn giản được định nghĩa trên quan hệ R_i. Các phần tử của Pr_i được ký hiệu là p_{ij}.

Ví dụ 2.5. Cho quan hệ PROJ

PNAME = "Maintenance"

BUDGET ≤ 200000

là các vị từ đơn giản.

- Trong thực tế các câu truy vấn là tổ hợp của rất nhiều vị từ đơn giản. Mỗi tổ hợp được gọi là một vị từ hội sơ cấp. Cho tập $Pr_i = \{p_{i1}, p_{i2}, ..., p_{im}\}$ là các vị từ đơn giản trên quan hệ R_i, tập các vị từ hội sơ cấp $M_i = \{m_{i1}, m_{i2}, ..., m_{iz}\}$ được định nghĩa như sau: Cho tập $Pr_i = \{p_{i1}, p_{i2}, ..., p_{im}\}$ là các vị từ đơn giản trên quan hệ R_i, tập các vị từ hội sơ cấp $M_i = \{m_{i1}, m_{i2}, ..., m_{iz}\}$ được định nghĩa như sau:

$$M_i = \left\{ m_{ij} \middle| m_{ij} = \wedge_{p_{ik} \in pr_i} p_{ik}^* \right\} \quad , \ 1 \le k \le m \ , \ 1 \le j \le z \ .$$

Trong đó, $p_{ik}^* = p_{ik}$ hoặc $p_{ik}^* = \neg p_{ik}$. Vì thế mỗi vị từ đơn giản có thể xuất hiện trong vị từ hội sơ cấp dưới dạng tự nhiên hoặc dạng phủ định của nó.

Phủ định của một vị từ sẽ có thể:

Attribute = Value không có phủ định.

Attribute ≤ Value, phủ định là *Attribute > Value*

Cận_dưới ≤ Attribute_1, phủ định là *¬(Cận_dưới ≤ Attribute_1)*

Attribute_1 ≤ Cận_trên, phủ định là *¬(Attribute_1 ≤ Cận_trên)*

Cận_dưới ≤ Attribute_1 ≤ Cận_trên, phủ định là

¬(Cận_dưới ≤ Attribute_1 ≤ Cận_trên)

Ví dụ 2.4: Xét một số vị từ đơn giản có thể định nghĩa được trên quan hệ PAY.

p_1: TITLE = "Elect.Eng"

p_2: TITLE = "Syst. Anal"

p_3: TITLE = "Mech. Eng"

p_4: TITLE = "Programmer"

p_5: SAL ≤ 30000

p_6: SAL > 30000

Các vị từ hội sơ cấp được định nghĩa dựa trên các vị từ đơn giản:

m_1: TITLE = "Elect.Eng" ^ SAL ≤ 30000

m_2: TITLE = "Elect.Eng" ^ SAL > 30000

m_3: ¬(TITLE = "Elect.Eng") ^ SAL ≤ 30000

m_4: ¬(TITLE = "Elect.Eng") ^ SAL > 30000

m_5: TITLE = "Programmer" ^ SAL ≤ 30000

m_6: TITLE = "Programmer" ^ SAL > 30000

PAY

TITLE	SAL
Elect.Eng	40000
Mech.Eng	27000
Programmer	24000
Syst.Anal	34000

Thông tin số lượng về ứng dụng cần phải có hai tập dữ liệu:

1. *Độ tuyển hội sơ cấp* (Minterm Selectivity): số các bộ của quan hệ sẽ được chọn theo vị từ hội sơ cấp cho trước, ký hiệu chọn của hội sơ cấp m là sel(m).Ví dụ, không có bộ nào được chọn trong PAY thoả mãn vị từ hội sơ cấp m1. Có 1 bộ thoả m2.
2. *Tần số ứng dụng người dùng truy nhập dữ liệu*. Nếu $Q = \{q_1, q_2, \ldots, q_q\}$ là tập truy vấn, ký hiệu $acc(q_i)$ là tần số truy nhập của truy vấn q_i trong một khoảng thời gian đã cho.
3. *Tần số truy nhập hội sơ cấp* là tần số truy nhập của hội sơ cấp *m,* ký hiệu là acc(m).

2.2.3 Phân mảnh ngang cơ sở

Phân mảnh ngang cơ sở được định nghĩa bằng phép chọn trên quan hệ đích của lược đồ CSDL. Cho quan hệ R, các mảnh ngang ,

$$R_i = \sigma_{F_i}(R) \quad , \text{ i-1 ...n}$$

Trong đó F_i là biểu thức đại số quan hệ. Nếu F_i có dạng chuẩn hội, thì nó là vị từ hội sơ cấp (m_i). Thực tế, thuật toán sẽ thảo luận khẳng định F_i là vị từ hội sơ cấp.

Ví dụ 2.5: Xết quan hệ PROJ

F_1:= {BUDGET≤200000} và

F_2:= {BUDGET >200000}

Khi đó quan hệ PROJ được phân rã thành các mảnh ngang PROJ1 và PROJ2 như sau:

$PROJ_1 = \sigma_{BUDGET \le 200000}(PROJ)$

$PROJ_2 = \sigma_{BUDGET > 200000}(PROJ)$

Giả sử tập các biểu thức đại số quan hệ:

F_1:= {BUDGET≤200000}

F_2:= { 200000 < BUDGET ≤ 400000}

F_3:= { 400000 < BUDGET ≤ 600000}

F_4:= { 600000 < BUDGET}

Khi đó quan hệ PROJ được phân rã thành các mảnh ngang như sau:

$PROJ_1 = \sigma_{BUDGET \le 200000}(PROJ)$

$PROJ_2 = \sigma_{200000 < BUDGET \le 400000}(PROJ)$

$PROJ_3 = \sigma_{400000 < BUDGET \le 600000}(PROJ)$

$PROJ_4 = \sigma_{600000 < BUDGET}(PROJ)$

Ví dụ 2.6: Xét quan hệ PROJ, các mảnh ngang sau đây được định nghĩa dựa vào vị trí các dự án. Các mảnh như sau:

$PROJ_1 = \sigma_{LOC=\text{"Montreal"}}(PROJ)$;

$PROJ_2 = \sigma_{LOC=\text{"New York"}}(PROJ)$;

$PROJ_3 = \sigma_{LOC=\text{"Paris"}}(PROJ)$

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal

$PROJ_2$

PNO	PNAME	BUDGET	LOC
P2	Database Develop	135000	New York
P3	CAD/CAM	250000	New York

$PROJ_3$

PNO	PNAME	BUDGET	LOC
P4	Maintenance	310000	Paris

Hình 2.5. Phân mảnh ngang cơ sở quan hệ PROJ

Phân mảnh ngang R_i của quan hệ R gồm tất cả các bộ của R thoả một vị từ hội sơ cấp m_i. Vì vậy, cho một tập M các vị từ hội sơ cấp, số lượng phân mảnh ngang của quan hệ R bằng số lượng các vị từ hội sơ cấp. Tập các mảnh ngang được gọi là tập các mảnh hội sơ cấp (Minterm Fragment). Định nghĩa phân mảnh ngang phụ thuộc vào vị từ hội sơ cấp, vì vậy, cần phải xác định các vị từ đơn giản rạo ra vị từ hội sơ cấp.

2.2.4 Tính đầy đủ và tính cực tiểu của vị từ đơn giản

Vị từ đơn giản có tính đầy đủ (Completeness) và tính cực tiểu (Minimality). Tập các vị từ đơn giản Pr là đầy đủ khi và chỉ khi xác suất truy nhập bởi mỗi ứng dụng tới bộ bất kỳ của một mảnh hội sơ cấp bất kỳ được định nghĩa theo Pr là như nhau. Vị từ đầy đủ sẽ đảm bảo cho các mảnh thoả mãn các vị từ sơ cấp, nhất quán về mặt logic. Đồng nhất về mặt thống kê theo cách ứng dụng truy nhập. Vì vậy, sử dụng một tập vị từ đầy đủ làm cơ sở cho việc phân mảnh ngang cơ sở.

Ví dụ 2.8: Trong ví dụ 2.7, các ứng dụng truy nhập PROJ theo vị trí, thì tập vị từ tương ứng là đầy đủ, vì mỗi bộ của các mảnh PROJi , i=1,2,3, đều có xác xuất được truy nhập bằng nhau. Tuy nhiên, nếu thêm ứng dụng dự án có ngân sách trên 200.000$ vào tập Pr, thì Pr sẽ là không đầy đủ, vì một số bộ trong các mảnh $PROJ_i$ có xác suất được truy nhập lớn hơn. Để tập vị từ này là đầy đủ thì cần phải thêm các vị từ

(BUDGET≤200000, BUDGET>200000) vào Pr :

Pr = {LOC= “Montreal”, LOC=”New York”, LOC=”Paris”,
BUDGET≤200000, BUDGET > 200000}

Đặc tính thứ hai của tập các vị từ có tính cực tiểu. Nếu một vị từ f phân mảnh thành các mảnh nhỏ hơn f1 và f2, thì phải có ít nhất một ứng dụng truy nhập đến f1 và f2 theo các cánh khác nhau. Khi đó nói rằng vị từ đơn giản f có tính liên đới (Relevant), ảnh hưởng đến việc xác định một phân mảnh. Nếu tất cả các vị từ của tập Pr đều có liên đới thì tập Pr được gọi là cực tiểu.

Tính liên đới của vị từ đơn giản được định nghĩa hình thức lnhư sau: Gọi m_i và m_j là hai vị từ hội sơ cấp. Vị từ hội sơ cấp m_i chứa vị từ đơn giản p_i, vị từ hội sơ cápp m_j chứa p_j dạng phủ định $\neg p_i$. Gọi f_i và f_j là hai mảnh tương ứng được định nghĩa theo m_i và m_j. Khi đó p_i là có liên đới khi và chỉ khi:

$$\frac{acc(m_i)}{card(f_i)} \neq \frac{acc(m_j)}{card(f_j)}$$

Ví dụ 2.9: Tập Pr = {LOC= "Montreal", LOC="New York", LOC="Paris", BUDGET≤200000, BUDGET > 200000} là tập vị từ đầy đủ và cực tiểu. Nếu thêm vị từ PNAME = ""Instrumentation" và tập Pr, khi đó sẽ không đảm bảo tính cực tiểu, vì vị từ thêm vào không có tính liên đới ứng với Pr. Không một ứng dụng nào truy xuất khác nhau đến các mảnh được tạo ra.

2.2.5 Thuật toán xác định tập vị từ đầy đủ và cực tiểu từ tập Pr cho trước

Thuật toán COM_MIN tạo ra ra một tập đầy đủ và cực tiểu các vị từ Pr' từ một tập các vị từ đơn giản Pr cho trước theo quy tắc: một quan hệ hoặc một mảnh "được phân hoạch thành ít nhất hai phần và chúng được truy nhập khác nhau bởi ít nhất bởi một ứng dụng"

Mảnh f_i được định nghĩa theo một vị từ hội sơ cấp trên Pr', qui ước là f_i của Pr'

Thuật toán 2.1. COM_MIN

```
Input:   R là quan hệ cần phân mảnh ngang cơ sở. Pr là tập các vị từ đơn giản
Output:  Pr' là tập các vị từ đơn giản.
Khai báo: F là tập các mảnh hội sơ cấp.
Begin
    Tìm một vị từ  p_i ∈ Pr sao cho p_i phân hoạch R theo qui tắc
    Pr' ← p_i
    Pr ← Pr – p_i
    F ← f_i                {f_i là mảnh hội sơ cấp theo p_i }
    do
        begin
            Tìm một p_j∈Pr sao cho p_j phân hoạch một mảnh f_k của Pr' theo qui tắc
            P_r' ← P_r' ∪ p_j
            Pr ← Pr - p_j
            F ← F ∪ f_i
            If ∃p_k ∈ Pr', một vị từ không có liên đới then
                begin
                    P_r' ← Pr' - p_k
                    F ← F - p_k
                end-if
        end-begin
    until Pr đầy đủ
End. { COM_MIN}
```

Thuật toán thực hiện như sau:

- Bắt đầu bằng cách tìm một vị từ có liên đới và phân hoạch quan hệ đã cho. Vòng lặp do-until thêm các vị từ vào tập Pr', đảm bảo Pr' là cực tiểu và đầy đủ.

- Bước thứ hai: Trong quá trình phân mảnh ngang cơ sở suy dẫn ra tập các vị từ hội sơ cấp có thể được định nghĩa trên các vị từ trong tập Pr'. Các vị từ hội sơ cấp này xác định các mảnh cấp phát.
- Bước thứ ba: Các vị từ hội sơ cấp có thể rất lớn, tỷ lệ hàm mũ theo số lượng các vị từ đơn giản Vì vậy cần phải loại bỏ những mảnh không có ý nghĩa, bằng cách xác định những vị từ mâu thuẫn với tập các phép kéo theo (Implication).

2.2.6 Thuật toán phân mảnh ngang nguyên thuỷ

Thuật toán 2.2. ***PHORIZONTAL***

```
Input:    R là quan hệ cần phân mảnh ngang cơ sở.
          Pr là tập các vị từ đơn giản
Output:   M là tập các vị từ hội sơ cấp
Begin
    Pr' ← COM_MIT(R, Pr)
    Xác định tập M các vị từ hội sơ cấp
    Xác định tập I các phép kéo theo giữa các pi ∈ Pr'
    For mỗi mi ∈ M do
        If mi mâu thuẫn với I then
            M ← M - mi
        End_if
    End_for
End.    {PHORIZONTAL}
```

Ví dụ 2.10. Giả sử chỉ có một ứng dụng kiểm tra thông tin lương và xác định số lương sẽ tăng trên quan hệ PAY. Giả sử có hai vị trí, một vị trí xử lý các bộ có lương thấp hơn hoặc bằng 30000$ và vị trí còn lại xử lý các bộ có lương cao hơn 30000$. Câu truy vấn sẽ thực hiện trên cả hai vị trí. Tập vị từ đơn giản sử dụng để phân hoạch quan hệ PAY là:

p_1: SAL $\leq$30000

p_2: SAL >30000

Tập vị từ đơn giản khởi đầu là Pr = {p1, p2}. Áp dụng thuật toán COM_MIN với i=1 làm giá trị khởi đầu tạo ra Pr'={p1} là tập đầy đủ và cực tiểu vì p2 không phân hoạch f_1 (là mảnh hội sơ cấp được tạo ra ứng với p1) theo quy tắc. Như vậy, các vị từ hội sơ cấp sau đây là các phần tử của M:

m_1: (SAL $\leq$ 30000)

m_1: $\neg$(SAL $\leq$ 30000) = SAL > 30000

Khi đó, hai mảnh F_s = {PAY_1, PAY_2} theo M là:

PAY_1

TITLE	SAL
Mech. Eng.	27000
Programmer	24000

PAY_2

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000

Hình 2.6. Phân mảnh ngang cho quan hệ PAY

Giả sử có hai ứng dụng trên quan hệ PROJ. Ứng dụng thứ nhất xác định tên các dự án và ngân sách của chúng trên ba vị trí.

```
SELECT      PNAME, BUDGET
FROM        PROJ
WHERE       LOC = Value
```

Ứng dụng này, các vị từ đơn giản có thể được sử dụng là:

p_1: LOC = "Montreal"

p_2: LOC = "New York"

p_3: LOC = "Paris"

Ứng dụng thứ hai liên quan đến các dự án có ngân sách nhỏ hơn hoặc bằng 200000$ được quản lý tại một vị trí và các dự án có ngân sách lớn hơn 200000 được quản lý tại vị trí thứ hai. Vì vậy, các vị từ đơn giản được sử dụng để phân mảnh ứng dụng thứ hai là:

p_4: BUDGET $\leq$ 200000

p_5: BUDGET $>$ 200000

Sử dụng thuật toán COM_MIN kiểm tra tập Pr' = {p1, p2, p3, p4, p5} là đầy đủ và cực tiểu. Có thể định nghĩa sáu vị từ hội sơ cấp tạo ra M dựa trên Pr' như sau::

m_1: (LOC="Montreal") $\wedge$ (BUDGET$\leq$20000)

m_2: (LOC="Montreal") $\wedge$ (BUDGET>20000)

m_3: (LOC="New York") $\wedge$ (BUDGET$\leq$20000)

m_4: (LOC="New York") $\wedge$ (BUDGET>20000)

m_5: (LOC="Paris") $\wedge$ (BUDGET$\leq$20000)

m_6: (LOC="Paris") $\wedge$ (BUDGET>20000)

Kết quả phân mảnh ngang cơ sở PROJ tạo ra sáu mảnh FPROJ = {$PROJ_1$, $PROJ_2$, $PROJ_3$, $PROJ_4$, $PROJ_5$, $PROJ_6$} theo các vị từ hội sơ cấp M. Các mảnh $PROJ_2$, $PROJ_5$ rỗng.

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal

$PROJ_3$

PNO	PNAME	BUDGET	LOC
P2	Database	135000	New York

$PROJ_4$

PNO	PNAME	BUDGET	LOC
P3	CAD/CAM	250000	New York

$PROJ_6$

PNO	PNAME	BUDGET	LOC
P4	Maintenance	31000	Paris

Hình 2.7 . Phân hoạch ngang quan hệ PROJ

2.3 PHÂN MẢNH NGANG DẪN XUẤT

Phân mảnh ngang dẫn xuất dựa trên các quan hệ thành viên của một đường nối theo phép toán chọn trên quan hệ chủ. Mục tiêu của phân mảnh ngang dẫn xuất là phân chia các quan

hệ thành viên thành các mảnh của quan hệ chủ được định nghĩa trên các thuộc tính của quan hệ thành viên. Vì vậy liên kết giữa quan hệ chủ và quan hệ thành viên được định nghĩa như là một nối bằng (Equijoin) và kết nối bằng có thể được cài đặt nối nửa (Semijoin).

Cho một đường nối L trong đó, Owner(L) = S và Member(L) = R. Phân mảnh ngang dẫn xuất của R được định nghĩa như sau:

$R_i = R \alpha S_i \quad 1 \le i \le k$, trong đó, k là số mảnh

$S_i = \sigma_{Fi}(S)$, Fi là biểu thức định nghĩa mảnh ngang nguyên thuỷ S_i.

Ví dụ 2.11.

Cho Owner(L) = PAY và Member(L) = EMP. Nhóm các kỹ sư (Engineer) thành hai nhóm theo lương SAL≤30000 và SAL>30000.

$PAY1 = \sigma_{SAL \le 30000}(PAY)$

$PAY2 = \sigma_{SAL > 30000}(PAY)$

Hai mảnh EMP1 và EMP2 được định nghĩa như sau:

$EMP_1 = EMP \propto PAY_1$

$EMP_2 = EMP \propto PAY_2$

EMP_1

ENO	ENAME	TITLE
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E7	R. David	Mech. Eng.

EMP_2

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Syst. Anal.
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E8	J. Jones	Syst. Anal.

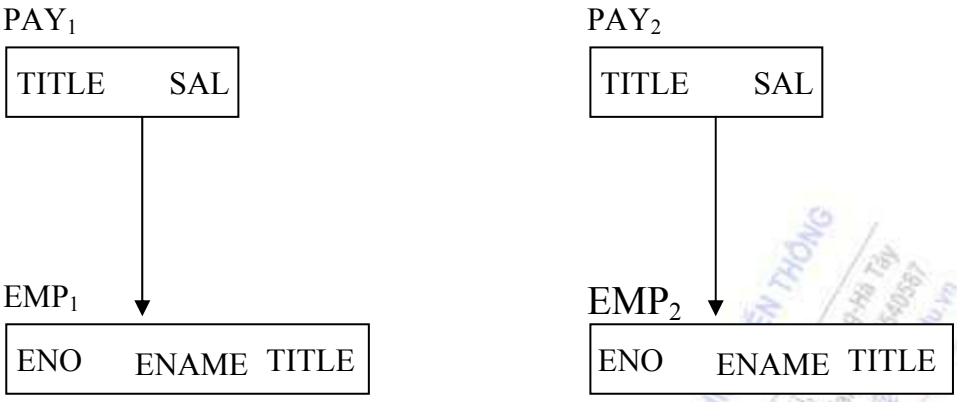
Hình 2.8. Phân mảnh ngang dẫn xuất quan hệ EMP

Trong một số trường hợp, như thí dụ 2.11 ở trên, mỗi một mảnh chỉ có một đường liên kết đến hoặc đi. Đồ thị có cấu trúc như vậy, được gọi là đồ thị đơn giản. Ưu điểm của thiết kế loại này là mối liên kết giữa các mảnh đơn giản. Quan hệ thành viên và quan hệ chủ nhân của đường kết nối có thể cấp phát cho một vị trí và các nối giữa các cặp mảnh khác nhau có thể tiến hành độc lập và song song với nhau. Ngoài việc thực hiện được nhiều câu vấn tin tại nhiều vị trí khác nhau, còn có thể thực hiện song song một câu vấn tin với thời gian đáp ứng và lưu lượng của hệ thống tối ưu hơn. Ví dụ giữa các mảnh EMP và PAY. Trong những trường hợp này, thuật toán phân mảnh hoà toàn tầm thường.

Trong một số trường hợp khác, có nhiều hơn hai liên kết đến một quan hệ R. Khi đó người ta thường chọn một thiết kế tạo ra một đồ thị nối phân hoạch (Partioned Join Graph). Một đồ thị phân hoạch có thể chứa hai hoặc nhiều đồ thị con và không có đường kết nối giữa chúng. Các đồ thị như vậy thường có thể cấp phát, nhưng rất khó phân tán để thực hiện song song. Trong hình 2.4 có hai đường nối đến quan hệ ASG. Như vậy sẽ có nhiều cách phân mảnh ngang dẫn xuất quan hệ R. Việc quyết định chọn phân mảnh nào tối ưu hơn cần dựa trên hai tiêu chuẩn sau:

1. Phân mảnh có đặc tính kết nối tốt hơn

2. Phân mảnh được sử dụng cho nhiều ứng dụng hơn



Hình 2.9. Đồ thị đơn giản nối giữa các mảnh

Ví dụ 2.12 Trong hình 2.4 ta có

Owner(L1) = PROJ và Member(L1) = ASG.

Owner(L2) = EMP và Member(L2) = ASG.

Thực hiện phân mảnh dẫn xuất của ASG ứng PROJ và EMP như sau: Xét quan hệ ASG có hai ứng dụng trên nó:

ASG_1

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P2	Analyst	24

ASG_3

ENO	PNO	RESP	DUR
E3	P3	Consultant	10
E6	P3	Engineer	36
E7	P3	Manager	40

ASG_2

ENO	PNO	RESP	DUR
E2	P2	Analyst	6
E4	P2	Programmer	18
E5	P2	Manager	24

ASG_4

ENO	PNO	RESP	DUR
E3	P4	Engineer	48
E6	P4	Manager	48

Hình 2.10 Phân mảnh dẫn xuất của ASG ứng với PROJ

- Ứng dụng 1: Danh sách các kỹ sư làm việc tại một vị trí nào đó. Ứng dụng này thực hiện trên ba trạm và truy xuất thông tin về các kỹ sư làm việc trong các dự án tại chỗ với xác suất cao hơn các kỹ sư làm việc trong các dự án trên các vị trí khác.

Phân mảnh ASG theo các mảnh $PROJ_1$, $PROJ_2$, $PROJ_4$ và ,$PROJ_4$, như sau:

$$PROJ_1 = \sigma_{LOC=\text{“Montreal”} \wedge BUDGET \leq 20000}(PROJ)$$

$$PROJ_3 = \sigma_{LOC=\text{“New York”} \wedge BUDGET \leq 20000}(PROJ)$$

$$PROJ_4 = \sigma_{LOC=\text{“New York”} \wedge BUDGET > 20000}(PROJ)$$

$$PROJ_6 = \sigma_{LOC=\text{“Paris”} \wedge BUDGET > 20000}(PROJ)$$

Phân mảnh dẫn xuất của ASG theo $\{PROJ_1, PROJ_3, PROJ_4, PROJ_6\}$ có thể được định nghĩa như sau:

$$ASG_1 = ASG \propto PROJ_1$$

$ASG_2 = ASG \propto PROJ_3$

$ASG_3 = ASG \propto PROJ_4$

$ASG_4 = ASG \propto PROJ_6$

- Ứng dụng thứ 2: Tại các trạm quản lý nhân viên, ứng dụng sẽ truy xuất thông tin về thời gian thực hiện các dự án của các nhân viên. Câu vấn tin có thể viết:

```
SELECT    RESP, DUR
FROM      ASG, EMPi
WHERE     ASG..ENO = EMPi ..ENO
```

Trong đó i=1 hoặc i=2 tùy thuộc vào vị trí đưa ra truy vấn. Phân mảnh dẫn xuất của ASG theo phân mảnh của EMP được định nghĩa nhw sau:

$ASG_1 = ASG \propto EMP_1$

$ASG_2 = ASG \propto EMP_2$

ASG_1

ENO	PNO	RESP	DUR
E3	P3	Consultant	02
E3	P4	Engineer	48
E4	P2	Programmer	18
E7	P3	Engineer	36

ASG_2

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E3	P2	Analyst	6
E4	P2	Manager	24
E5	P4	Manager	48
E6	P3	Manager	40

Hình 2.11. Phân mảnh dẫn xuất của ASG ứng với EMP

Nhận xét

1. Phân mảnh dẫn xuất có thể xảy ra dây chuyền, trong đó một quan hệ được phân mảnh như là hệ quả của một phân mảnh cho một quan hệ khác, và đến lượt nó lại làm cho các quan hệ khác phải phân mảnh (như dây chuyền PAY-EMP-ASG).
2. Một quan hệ có thể có nhiều cách phân mảnh. Chọn lựa một lược đồ phân mảnh nào cho tối ưu phụ thuộc vào ứng dụng và cấp phát.

2.4 PHÂN MẢNH DỌC

2.4.1 Khái niệm phân mảnh dọc

Phân mảnh dọc quan hệ R sinh ra các mảnh R_1, R_2, ..., R_r, sao cho mỗi mảnh chứa một tập con các thuộc tính của quan hệ R và khoá của nó. Mục đích của phân mảnh dọc là phân chia quan hệ R thành tập các quan hệ nhỏ hơn để có nhiều ứng dụng có thể chỉ cần thực hiện trên một mảnh. Mảnh tối ưu là mảnh sinh ra một lược đồ phân mảnh cho phép giảm tối thiểu thời gian thực hiện của ứng dụng trên mảnh đó.

Kỹ thuật phân mảnh dọc phức tạp hơn so với kỹ thuật phân mảnh ngang, vì số lựa chọn phân hoạch rất lớn. Trong phân mảnh ngang, nếu số vị từ đơn giản trong Pr là n, khi đó sẽ có 2^n vị từ hội sơ cấp có thể thể được định nghĩa trên đó. Mặt khác, một số vị từ hội sơ cấp có

thể mâu thuẫn với các phép kéo theo. Vì vậy sẽ làm giảm số lượng các mảnh dự tuyển cần được xem xét. Trong trường hợp phân mảnh dọc, nếu quan hệ có m thuộc tính không phải là khoá chính, thì số mảnh có thể có là m^m .

Để có được các lời giải tối ưu cho bài toán phân mảnh dọc rất không hiệu quả, phải sử dụng hai phương pháp Heuristic cho phân mảnh dọc các quan hệ toàn cục:

1. *Nhóm thuộc tính*: bắt đầu gán mỗi thuộc tính cho một mảnh và trong mỗi bước, nối một số mảnh lại với nhau cho đến khi thỏa điều kiện.
2. *Tách mảnh* bắt đầu bằng một quan hệ và quyết định cách phân chia dựa trên hành vi truy nhập của các ứng dụng trên các thuộc tính.

Kỹ thuật tách mảnh thích hợp với phương pháp thiết kế từ trên xuống. Các mảnh không gối chồng lấn lên nhau (không phải là khoá chính). Ngược lại, với phương pháp nhóm thuộc tính thường tạo ra các mảnh gối chồng lên nhau. Trong các hệ CSDL phân tán, các mảnh không gối chồng lên nhau được quan tâm, nghiên cứu.

Việc nhân bản các thuộc tính khóa tại mỗi trạm sẽ bảo đảm tính toàn vẹn ngữ nghĩa và làm giảm đi quá trình trao đổi dữ liệu.

2.4.2 Thông tin cần thiết của phân mảnh dọc

a) Ma trận giá trị sử dụng thuộc tính: Gọi $Q=\{q_1, q_2,.., q_q\}$ là tập các câu vấn tin của người dùng hay còn gọi là các ứng dụng của người sử dụng., sẽ thực hiện trên quan hệ $R(A_1, A_2,\ldots, A_n)$. Khi đó ma trận giá trị sử dụng thuộc tính có q hàng và n cột. Các phần tử được ký hiệu là giá trị câu vấn tin q_i i=1..q sử dụng thuộc tính A_j , j=1.. n, ký hiệu là use(q_i, A_j) được định nghĩa như sau:

$$Use(q_i,A_j)= \begin{cases} 1 & \text{Nếu thuộc tính } A_j \text{ được vấn tin qi tham chiếu} \\ 0 & \text{Ngược lại} \end{cases}$$
$$i=1..q \text{ và } j=1..n$$

Ví dụ 2.13: Xét quan hệ PROJ .

PNO	PNAME	LOG
P1	Instrumentation	Montreal
P2	Database Develop	New York
P3	CAD/CAM	New York
P4	Maintenance	Paris

Giả sử có các ứng dụng sau ssược sử dụng trên quan hệ PROJ:

q1: Cho biết ngân sách của một dự án cụ thể

```
SELECT      BUDGET
FORM        PROJ
WHERE       PNO=Value
```

q2: Tên và ngân sách của tất cả dự án.

```
SELECT      BUDGET, PNAME
FORM        PROJ
```

q3: Tên của các dự án được thực hiện tại một thành phố biết trước

SELECT PNAME
FORM PROJ
WHERE PNO=Value

q4: Tổng ngân sách dự án cho mỗi thành phố

SELECT SUM(BUDGET)
FORM PROJ
WHERE LOC=Value

Ký hiệu A1 là thuộc tính PNO, A2 là thuộc tính PNAME, A3 l à thuộc tính BUDGET và A4 là thuộc tính LOC. Khi đó ma trận biểu diễn giá trị sử dụng các thuộc tính được biểu diễn như sau:

	A1	A2	A3	A4
q1	1	0	1	0
q2	0	1	1	0
q3	0	1	0	1
q4	0	0	1	1

Hình 2.12 .Một ví dụ về ma trận giá trị sử dụng thuộc tính

b) Ma trận hấp dẫn (Affinity) của thuộc tính: Hầu hết các thông tin cho phân mảnh dọc có mối quan hệ mật thiết với các ứng dụng. Các thuộc tính thường được truy nhập chung với nhau, cần phải định nghĩa một giá trị để chỉ ra mức độ liên hệ giữa các thuộc tính, được gọi là sự hấp dẫn (Affinity) của thuộc tính.

Số đo hấp dẫn giữa hai thuộc tính Ai, Aj của quan hệ R(A_1,A_2,…,A_n) ứng với tập ứng dụng Q={q_1,q_2,…,q_q} được định nghĩa như sau:

$$aff\ (A_i, A_j) = \sum_{k\,[(use\ (q_k, A_i) \wedge use\ (q_k, A_j)} \sum_{l \forall S_l} ref_l(q_k)\, acc_l(q_k)$$

Trong đó $ref_l(q_k)$ là số truy suất các thuộc tính (Ai,Aj) cho q_k tại vị trí S_l và $acc_l(q_k)$ là số đo tần số truy suất ứng dụng q_k tại vị trí S_l . Kết quả tính toán là một ma trận vuông cấp n x n, phần tử (i, j) là số đo sự hấp dẫn của các thuộc tính Ai, Aj.

Ví dụ 2.14

Giả sử $ref_l(q_k)$=1 cho tất cả q_k và S_l. Nếu tần số ứng dụng là:

$acc_1(q_1)$=15 $acc_2(q_2)$=20 $acc_3(q_1)$=10
$acc_1(q_2)$=5 $acc_2(q_2)$=0 $acc_3(q_2)$=0
$acc_1(q_3)$=25 $acc_2(q_3)$=25 $acc_3(q_3)$=25
$acc_1(q_4)$=3 $acc_2(q_4)$=0 $acc_3(q_4)$=0

Khi đó số đo sự hấp dẫn giữa thuộc tính A_1 và A_3 có thể đo bằng:

$$aff\ (A_1, A_3) = \sum_{k=1}^{l} \sum_{l=1}^{3} acc_1(q_k) = acc_1(q_1) + acc_2(q_1) + acc_3(q_1) = 45$$

Vì ứng dụng duy nhất truy xuất đến cả hai thuộc tính này là q1. Ma trận hấp dẫn thuộc tính đầy đủ được trình bày trong hình 2.13 Chú ý rằng để cho đầy đủ ,các giá trị ở đường chéo cũng được tính dù rằng chúng hoàn toàn vô nghĩa

	A1	A2	A3	A4
A1	45	0	45	0
A2	0	80	5	75
Â3	45	5	53	3
A4	0	75	3	78

Hình 2.13 Ma trận ái lực thuộc tính

2.4.3 Thuật toán tụ nhóm

Thuật toán năng lượng nối BEA (Bond Energy Algorithm) nhóm các thuộc tính của một quan hệ dựa trên các giá trị hấp dẫn thuộc tính trong ma trận AA. Thuật toán hoán hoán vị các hàng và các cột của ma trận hấp dẫn thuộc tính AA, sao cho số đo hấp dẫn chung AM là lớn nhất (Global Affinity Measure). Kết quả sẽ là một ma trận gọi là ma trận hấp dẫn tụ CA (Cluster Affinity). Thuật toán gồm 3 bước :

Bước 1: Đặt cố định một cột của AA vào trong CA. Thuật toán chọn cột 1.

Bước 2: Giả sử có i cột đã được thực hiện và đặt vào CA. Lấy lần lượt một trong (n-i) cột còn lại và thử đặt vào (i+1) vị trí còn lại trong ma trận CA .Ssao cho số đo hấp dẫn chung AM (Global Affinity Measure) tại vị trí đó là lớn nhất. Thuật toán tiếp tục cho đến khi không còn cột nào để đặt.

Bước 3: Cuối cùng là sắp thứ tự hàng. Khi thứ tự cột đã được xác định thì các hàng cũng cần phải sắp xếp vị trí phù hợp với các vị trí tương đối của các cột.

Thuật toán BEA (Thuật toán năng lượng nối)

```
Input:   AA:ma trận hấp dẫn thuộc tính
Output:  CA : Ma trận hấp dẫn tụ nhóm
Begin
   {Khởi tạo , nhớ rằng AA là  ma trận n x n}
   CA(.,1)←AA(.,1)
   CA(.,2)←AA(.,2)
   index ← 3
   While  index <= n  do    { chọn vị trí tốt nhất cho thuộc tính  AAindex}
      Begin
         For  i   From 1 to   index-1  by  1  do
              Tính cont(A_i-1,A_index,A_i )
         End-for
         Tính cont(A_index-1,A_index,A_index+1)  {Ddieeuf kiện biên}
         loc←vị trí được đặt bởi giá trị cont lớn nhất
         For  j  From  index  To loc By  -1  Do
              CA(.,j) ←AA(.,j-1)
         End-for
         CA(.,loc) ←AA(.,index)
         index←index+1
      End-While
End {BEA}
```

Ma trận hấp dẫn thuộc tính AA có tính đối xứng, số đo hấp dẫn chung AM được định nghĩa như sau:

$$AM=\sum_{i=1}^{n}\sum_{j=1}^{n} aff\ (A_i, A_j)[\ aff\ (A_i, A_{j-1}) + aff\ (A_i, A_{j+1})$$

Và có thể được viết lại:

$$AM=\sum_{i=1}^{n}\sum_{j=1}^{n} [aff\ (A_i, A_j) aff\ (A_i, A_{j-1}) + aff\ (A_i, A_j) aff\ (A_i, A_{j+1})]$$

$$= \sum_{j=1}^{n}[\sum_{i=1}^{n} aff(A_i, A_j) aff(A_i, A_{j-1}) + \sum_{i=1}^{n}\ aff(A_i, A_j) aff(A_i, A_{j+1})]$$

Cầu nối (Bond) giữa 2 thuộc tính A_x và A_y định nghĩa như sau:

$$bond\ (A_x, A_y) = \sum_{z=1}^{n}\ aff(A_z,A_x)\ aff(A_z,A_y)$$

Khi đó AM được biểu diễn:

$$AM=\sum_{j=1}^{n} [\ bond\ \ (A_j, A_{j-1}) + bond\ \ (A_j, A_{j+1})\ ]$$

Xét n thuộc tính: $\underbrace{A_1\ A_2 . . . A_{i-1}}_{AM'}\ A_i\ A_j\ \underbrace{A_{j+1} . . . An}_{AM''}$

Khi đó độ đo hấp dẫn chung cho n thuộc tính sẽ là:

$AM_{old} = AM'+AM'' + bond(A_{i-1}, Ai)+bond(A_i, A_j)+ bond(A_j, A_i)+ bond(Aj,A_{j+1})$

$$= \sum_{l=i}^{n}\ [bond\ (A_l, A_{l-1}) + bond\ (A_i, A_{l+1})]$$

$$+ \sum_{l=i+1}^{n}\ [bond(A_l,A_{l-1}) + bond(A_i, A_{l+1})+ 2*bond\ (A_i, A_j).$$

Số đo hấp dẫn chung mới khi đặt thuộc tính A_k vào giữa các thuộc tính A_i và A_j trong ma trận tụ hấp dẫn được tính như sau:

$AM_{new} = AM\ '+ AM''+ bond(A_i, A_k) + bond(A_k, A_i)+ bond(A_k, A_j)+bond(A_j,A_k)$

$= AM' + AM'' + 2*\ bond\ (A_i, A_k) + 2*bond(A_k, A_j)$

Vì thế đóng góp thực (Net contribution) cho số đo hấp dẫn chung khi đặt thuộc tính A_k giữa A_i và A_j là:

$Cont(A_i, A_k, A_j) = AM_{new} - AM_{old}$

$= 2*cont\ (A_i, A_k) + 2*cont\ (A_k, A_j) – 2*cont(A_i, A_j)$

Vị trí A_k nào cho số đóng góp hấp dẫn chung AM lớn nhất sẽ được chọn để đặt .Và tiếp tục chọn vị trí đặt tiếp theo cho đến khi không còn cột nào trống.

Cuối cùng sắp xếp lại các hàng theo thứ tự của các cột trong ma trận CA kết quả . Thuật toán kết thúc.

Ví dụ 2.15: Xét ma trận AA sau để tính phần đóng góp khi chuyển thuộc tính A_4 vào giữa thuộc tính $\hat{A}_1$ và A_2:

.

	A1	A2	A3	A4
A1	45	0	45	0
A2	0	80	5	75
Â3	45	5	53	3
A4	0	75	3	78

$cont(A_1, A_4, A_2) = 2*bond(A_1, A_4) + 2*bond(A_4, A_2) – 2*bond(A_1 A_2)$

$bond(A_1, A_4) = 45*0 + 0*75 + 45*3 + 0*78 = 135$

$bond(A_4, A_2) = 0*0 + 75*80 + 3*5 + 78*75 = 11865$

$bond(A_1, A_2) = 45*0 + 0*80 + 45*5 + 0*75 = 225$

$cont(A_1, A_4, A_2) = 2*135 + 2*11865 – 2*225 = 23550$

Một số điểm cần lưu ý

- Độ đo cầu nối giữa hai thuộc tính được tính là tổng của tích 2 phần tử cùng hàng của hai cột. Vì ma trận AA đối xứng , có thể thực hiện tương tự theo hàng.
- Thuật toán BEA được cải thiện hiệu quả khi cột thứ 2 được cố định và đặt cạnh cột thứ nhất trong bước khởi gán. Điều này có thể chấp nhận được, vì theo thuật toán A_2 có thể được đặt ở bên trái hoặc bên phải của A_1. Tuy nhiên cầu nối giữa hai thuộc tính này độc lập với vị trí tương đối của chúng.
- Nếu thuộc tính A_i đang được xét tìm chỗ đặt ở bên trái thuộc tính tận trái ,một trong các phương trình cầu nối được tính giữa phần tử không có ở bên trái và A_k (nghĩa là bond (A_0,A_k)), vì vậy cần sử dụng điều kiện đặt ra cho định nghĩa số đo hấp dẫn chung AM, $CA(0,k)=0$. Nếu A_j là thuộc tính tận phải đã được đặt trong ma trận CA và đang kiểm tra đóng góp khi đặt thuộc tính A_k vào bên phải của A_j, Khi đó bond(k,k+1) cần được tính. Tuy nhiên vì chưa có thuộc tính nào được đặt ở cột k+1 của ma trận CA, số đo hấp dẫn này chưa được định nghĩa.Vì thế theo điều kiện đầu tận , giá trị bond này cũng là 0.

Ví dụ 2.16: Xét quá trình gom tụ các thuộc tính của quan hệ PROJ và dùng ma trận hấp dẫn thuộc tính AA như sau:

Trước tiên chép cột 1 và cột 2 của ma trận AA vào ma trận CA (hình 14.a). Xét cột 3 (thuộc tính A_3). Có 3 vị trí có thể đặt cột 3: ở bên trái của cột 1 tạo ra thứ tự (3-1-2), đặt giữa cột 1 và 2 tạo ra thứ tự (1-3-2) và đặt bên phải cột 2 tạo ra thư tự (1-2-3). Lưu ý, để tính đóng góp của sắp xếp (1-2-3) là phải tính $cont(A_2,A_3,A_4)$ chứ không phải cont(1-2-3). Hơn nữa A_4 chỉ cột thứ tư trong ma trận CA, đó là vị trí rỗng không phải là cột thuộc tính của ma trận AA. Đóng góp số đo hấp dẫn chung của mỗi khả năng được tính như sau:

- *Thứ tự (0-3-1)*

  cont (A0, A3, A1) = 2cont (A0, A3) + 2cont(A3, A1) – 2cont(A0, A1)

  Vì cont (A0, A3) = cont(A0, A1)= 0

  Vì vậy cont (A0, A3, A1) = 2cont(A3, A1) =2*(45*48+5*0+53*45+3*0) =8820
- *Thứ tự (1-3-2)*

  $cont (A_1, A_3, A_2) = 2cont (A_1, A_3) + 2cont(A_3, A_2) – 2cont(A_1, A_2)$

  Vì cont (A1, A_3) = cont (A_3, A_1) =4410

cont (A_3, A_2) = 890

cont(A1, A_2) =225

Vì thế : cont (A1, A3, A2) = 2*4410 + 2*890 – 2*225 =10150

- *Thứ tự (2-3-4)*

cont (A_2, A_3, A_4) = 2cont (A_2, A_3) + 2cont(A_3, A_4) – 2cont(A_2, A_4)

cont (A_2, A_3) =890

cont (A_3, A_4) =0

cont (A_2, A_4) =0

cont (A_2, A_3, A_4) = 2*890

Như vậy, đóng góp của thứ tự (1-3-2) là lớn nhất ,nên A_3 đặt vào giữa A_1 và A_2. Tương tự, tính toán với A_4 thấy rằng cần phải đặt nó vào bên phải của A_2 .Cuối cùng các hàng được được sắp xếp theo thứ tự như thứ tự của các cột

	A1	A2	A3	A4
A1	45	0	45	0
A2	0	80	5	75
Â3	45	5	53	3
A4	0	75	3	78

Hình 2.14a Ma trận hấp dẫn thuộc tính quan hệ PROJ

	A1	A2		
A1	45	0		
A2	0	80		
Â3	45	5		
A4	0	75		

Hình 2.14b Bước khởi gán

	A1	A3	A2	
A1	45	45	0	
A2	0	5	80	
Â3	45	53	5	
A4	0	3	75	

Hình 2.14C Đặt A_3 bên phải A_1

	A1	A3	A2	A4
A1	45	45	0	0
A2	0	5	80	75
Â3	45	53	5	3
A4	0	3	75	78

Hình 2.14d Đặt A_4 bên phải A_2

	A1	A3	A2	A4
A1	45	45	0	0
Â3	45	53	5	3
A2	0	5	80	75
A4	0	3	75	78

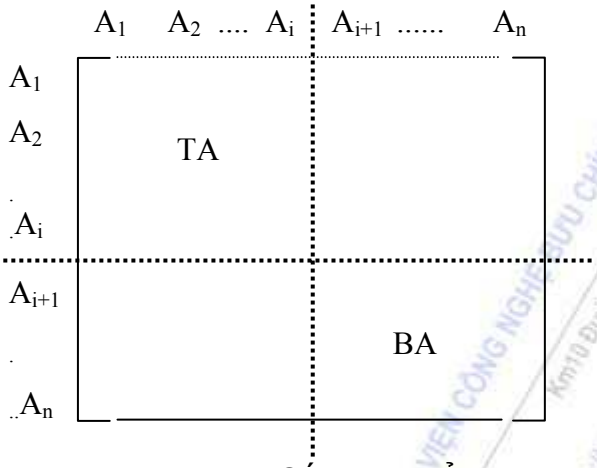
Hình 2.14e Hoán vị hàng

Hình 2.14a Ma trận hấp dẫn tụ CA

Trong hình 2.14e tụ thuộc tính xẩy ra góc trên trái ma trận chứa các giá trị hấp dẫn nhỏ và tụ kia ở góc dưới phải chứa các giá trị hấp dẫn cao. Quá trình tự đã chỉ ra phương thức phân mảnh dọc quan hệ PROJ. Nói chung khi ma trận CA lớn thường có nhiều tụ hơn được tạo ra và có nhiều phân hoach hơn.

2.4.4 Thuật toán phân mảnh

Mục đích của việc phân mảnh dọc quan hệ là xác định các tập thuộc tính được truy nhập bởi các tập ứng dụng. Giả sử trong ma trận tụ hấp dẫn CA của quan hệ, hình 2.16 nếu một điểm trên đường chéo chính được chọn, hai tập thuộc tính sẽ được xác định. Một tập {A1,A2,.....,Ai} ở góc trái cao nhất, gọi là tập đỉnh TA (Top) và tập thứ hai {Ai+1,....,An} ở góc phải thấp nhất, gọi là tập đáy BA (Bottom).



Hình 2.15 : Cấp phát điểm tách

Ký hiệu $Q=\{q_1,q_2,...,q_n\}$ là tập các ứng dụng. Khi đó

- $AQ(q_i) = \{A_j \mid use(qi, A_j) = \}$: Tập các thuộc tính được ứng dụng q_i truy nhập
- $TQ = \{q_i \mid AQ(q_i) \subseteq TA\}$: Tập các ứng dụng truy nhập trên các thuộc tính TA.
- $BQ = \{q_i \mid AQ(q_i) \subseteq BA\}$: Tập các ứng dụng truy nhập trên các thuộc tính BA.
- $OQ = Q - \{TQ \cup BQ\}$: Tập các ứng dụng truy nhập trên BA và TA

Nếu bậc của quan hệ là n, khi đó sẽ có n-1 vị trí có thể của điểm phân chia dọc trên đường chéo ma trận tụ hấp dẫn CA. Vị trí tốt nhất để phân chia là vị trí tạo ra các tập TQ và BQ sao cho tổng truy nhập từng mảnh là tối đa và tổng truy nhập tới cả hai mảnh là tối thiểu. Các phương trình chi phí được định nghĩa như sau:

$$CQ = \sum_{q_i \in Q} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$$

$$CTQ = \sum_{q_i \in TQ} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$$

$$CBQ = \sum_{q_i \in BQ} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$$

$$COQ = \sum_{q_i \in OQ} \sum_{\forall S_j} ref_j(q_i) acc_j(q_i)$$

Mỗi biểu thức tính tổng các truy nhập các thuộc tính của mỗi ứng dụng trong các lớp tương ứng. Bài toán tối ưu hoá phân mảnh chính là bài toán xác định điểm z ($1 \le z \le n$) sao cho $z = CTQ * CBQ - COQ^2$ là nhỏ nhất. Biểu thức phải xác định hai mảnh sao cho giá trị CTQ và CBQ xấp xỉ nhau. Điều này cho sẽ phép giải quyết được vấn đề cân bằng tải khi xử lý phân tán các mảnh trên nhiều vị trí. Trong trường hợp này, thuật toán có độ phức tạp là tuyến tính O(n), với n là số các thuộc tính của quan hệ, hay là bậc của quan hệ.

Thuật toán phân mảnh PARTITION: Giả thiết thủ tục chuyển đổi SHIFT đã được cài đặt.

Input: Ma trận tụ hấp dẫn CA
R: Quan hệ cần phân mảnh
ref: Ma trận sử thuộc tính
acc: Ma trận tần số truy nhập
Output: F là tập các mảnh
Begin
{Xác định giá trị z cho cột thws nhất, các chỉ số trong các công thức tính chi phí chỉ ra điểm tách}
calculate CTQ_{n-1}
calculate CBQ_{n-1}
calculate COQ_{n-1}
best← $CTQ_{n-1} + CBQ_{n-1} - (COQ_{n-1})^2$
do {xác định giải pháp phân chia tốt nhất}
begin
for i from *n-2* to 1 by -1 do
begin
calculate CTQ_i
calculate CBQ_i
calculate COQ_i
z← $CTQ_i * CBQ_i - COQ_i^2$
if z 〉 *best* then
begin
best ← *z*
Ghi lại điểm phân chia trong Shift
end-if
end-for
call SHIFT(*CA*)
end-begin
Tới khi không còn SHIFT
Xây dựng lại ma trận theo vị trí chuyểnđổi
$R_1 \leftarrow \prod_{TA}(R) \cup K$ {K là tập các thuộc tính khoá chính của R}
$R_2 \leftarrow \prod_{BA}(R) \cup K$
F ← { R_1,R_2}
end {PARTITION}.

Ví dụ 2.17: Ứng dụng thuật toán phân mảnh PẢTITION cho quan hệ PROJ với ma trận tụ hấp dẫn CA.trong hình 2.15e

F_{PROJ} ={$PROJ_1$,$PROJ_2$}
$PROJ_1$={A_1,A_2} = {PNO,BUDGET}
$PROJ_2$ ={A_1,A_2,A_3}={PNO,PNAME,LOC}

2.4.5 Kiểm tra tính đúng đắn

Tương tự như phân mảnh ngang, cần chứng minh rằng thuật toán phân mảnh dọc PARTITION cho một kết quả phân mảnh đúng.

Tính đầy đủ: Thuật toán đảm bảo tính đầy đủ, vì mỗi một thuộc tính của quan hệ toàn cục được gán cho một mảnh. Tập các thuộc tính của quan hệ toàn cục đúng bằng hợp của các thuộc tính các mảnh thành viên.

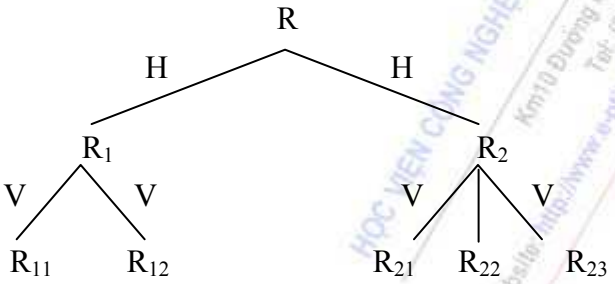
Tính khôi phục lại: Quan hệ toàn cục ban đầu R có thể đươc khôi phục lại bằng các phép kết nối bằng nhau trên các thuộc tính khoá. giả sử quan hệ toàn cục R có phân mảnh dọc F_R

$=\{R_1,R_2,....,R_r\}$ và tập các thuộc tính khoá chính K. Trong mỗi quan hệ R_i chứa các thuộc tính khoá K và là các quan hệ đầy đủ Khi đó: $R = \triangleright\triangleleft_K R_i$, i=1..r

Tính tách biệt: Tính tách biệt trong phân mảnh dọc không quan trọng bằng trong phân mảnh ngang. Tính phân biệt ở đây được hiểu là các thuộc tính không khoá hoàn toàn tách biệt nhau trong mỗi quan hệ tách thành viên.

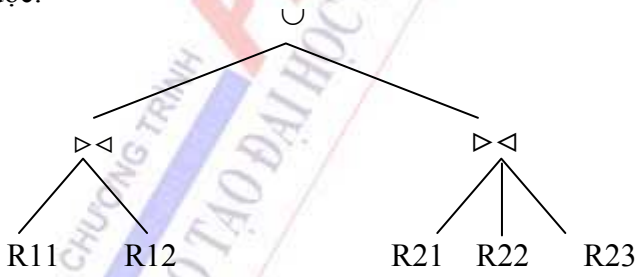
2.5 PHƯƠNG PHÁP PHÂN MẢNH HỖN HỢP (HYBRID FRAGMENTATION)

Trong thực tế, hầu hết các trường hợp phân mảnh ngang hay phân mảnh dọc đơn giản của CSDL không thoả mãn yêu cầu người sử dụng. Thường CSDLđược phân mảnh dọc sau đó phân mảnh ngang, tạo ra sự phan chia theo cấu trúc cây (Hình 2.17). Chiến lược phân mảnh này gọi là phân đoạn lai, hay phân mảnh trộn, hay phân mảnh lồng



Hình 2.16 Phân đoạn hỗn hợp.

Trong ví dụ 2.10, quan hệ PROJ được phân hoạch thành 6 mảnh ngang dựa vào 2 ứng dụng . Trong ví dụ 2.17, nó lại được phân hoạch thành 2 mảnh dọc. Như vậy, kết quả phân hoạch quan hệ PROJ là một tập mảnh ngang, trong đó mỗi một mảnh lại được tiếp tục phân chia thành 2 mảnh dọc.



Hình 2.17 Tái xây dựng phân đoạn hỗn hợp

Tính đúng đắn của phân mảnh hỗn hợp được đảm bảo, vì chiến lược phân mảnh này dựa theo phân mảnh ngang và phân mảnh dọc, đảm bảo tính đúng đắn như đã trình bày. Để khôi phục quan hệ toàn cục trong trường hợp phân mảnh hỗn hợp, có thể bắt đầu từ các node lá của của cây phân hoạch và dịch chuyển lên bằng cách thực hiện các phép kết nối và phép hợp (Hình 2.17). Phân mảnh hỗn hợp đầy đủ nếu các mảnh lá và các mảnh trung gian là đầy đủ. Tương tự, có thể kiểm tra tính tách biệt của chiến lược phân mảnh hỗn hợp. Tính tách biệt được đảm bảo khi các mảnh lá và mảnh trung gian cũng tách biệt..

2.6 CẤP PHÁT

Cấp phát tài nguyên cho các node mạng máy tính là một bài toán đã được nhiều người nghiên cứu. Tuy nhiên, phần lớn các kết quả nghiên cứu không tiếp cận bài toán thiết kế CSDL phân tán, mà chỉ quan tâm đến việc cài đặt các file riêng lẻ trên trên mạng máy tính. Vì sao có sự khác biệt giữa hai vấn đề này. Trước hết cần định nghĩa bài toán cấp phát một cách chính xác hơn.

2.6.1 Bài toán cấp phát (AllocationProblem)

Giả sử có một tập các mảnh dữ liệu $F = \{F_1, F_2, \ldots, F_n\}$ và một mạng máy tính bao gồm các node $S = \{S_1, S_2, \ldots, S_m\}$ trên đó có một tập các ứng dụng $Q = \{q_1, q_2, \ldots, q_q\}$ đang chạy. Bài toán cấp phát là tìm một phân phối tối ưu theo nghĩa nào đó cho tập F cho S.

Tính tối ưu cấp phát có thể được định nghĩa::

1. *Chi phí nhỏ nhất*: Hàm chi phí bao gồm chi phí lưu trữ mảnh F_i tại node S_j ,chi phí truy vấn F_i tại trạm S_j, chi phí cập nhật F_i tại tất cả các node lưu trữ nó và chi phí trao đổi thông tin. Mục tiêu của bài toán cấp phát là xác định một lược đồ cấp phát với hàm chi phí nhỏ nhất.
2. *Hiệu năng*: Chiến lược cấp phát phải nhằm duy trì hiệu năng. Làm giảm thời gian đáp ứng và tăng tối đa lưu lượng hệ thống tại mỗi node mạng.

Cần xây dựng một lược đồ cấp phát sao cho trả lời các truy vấn trong thời gian ngắn nhất mà vẫn duy trì được chi phí xử lý thấp nhất., thời gian đáp ứng thấp và tăng tối đa lưu lượng. Đây là vấn đề phức tạp, không đơn giản.

Để bài toán đơn giản hơn, giả sử rằng:

1. Tập các ứng dụng Q chỉ có khả năng vấn tin cập nhật (Update Query) và vấn tin chỉ đọc (Retrieval only Query). Trên mảnh $F_{k.}$ bao ggồm các đại lượng sau:

   $T = \{t_1, t_2, \ldots, t_m\}$, với t_i là *lưu lượng chỉ đọc* tại S_i cho F_k, và

   $U = \{u_1, u_2, \ldots, u_m\}$, với u_i là *lưu lượng cập nhật* tại S_i cho $F_{k.}$

2. Giả sử $C(T) = \{c_{12}, c_{13}, \ldots, c_{1m}, \ldots, c_{m-1,m}\}$, với c_{ij} là chi phí truyền một đơn vị cho các yêu cầu chỉ đọc giữa node S_i và S_j là không thay đổi.
   Chi phí cập nhật và chỉ đọc ký hiệu là:

   $C'(U) = \{c'_{12}, c'_{13}, \ldots, c'_{1m}, \ldots, c'_{m-1,m},\}$ với c'_{ij} là chi phí truyền một đơn vị cho các yêu cầu cập nhật giữa các node S_i và S_j.

3. Gọi chi phí lưu trữ mảnh F_k tại trạm S_i là $d_{i.}$ Ký hiệu $D = \{d_1, d_2, \ldots, d_m\}$ là chi phí lưu trữ mảnh F_k tại mọi vị trí.
4. Giả sử không có ràng buộc về khả năng lưu trữ tại các trạm hoặc cho đường truyền.

Khi đó, bài toán cấp phát có thể được đặc tả như là một bài toán cục tiểu hoá chi phí trong đó tìm tập $I \subseteq S$ các nơi lưu trữ các bản sao của các mảnh.

Ký hiệu x_j là biến quyết định (Decisio Variable) chọn nơi đặt sao cho:

$$x_i = \begin{cases} 1 & \text{Nếu mảnh } F_k \text{ được đặt tại vị trí } S_j \\ 0 & \text{Trong trường hợp ngược lại.} \end{cases}$$

Khi đó đặc tả chính xác như sau:

$$\min\left[\sum_{i=1}^{m}\left(\sum_{j|S_j\in I} x_j u_j c'_{ij} + t_j \min_{j|S_j\in I} c_{ij}\right) + \sum_{j|S_j\in I} x_j d_j\right]$$

- Trong đó x_j bằng 0 hoặc 1.
- Số hạng thứ nhất tương ứng với chi phí truyền các cập nhật đến các vị trí có lưu trữ bản sao của mảnh và với chi phí thực hiện các yêu cầu chỉ đọc tại trạm đó nhưng với chi phí truyền dữ liệu nhỏ nhất.
- Số hạng thứ hai của hàm tính tổng chi phí lưu tất cả các bản sao của mảnh.

Đặc tả trên là đơn giản và không thích hợp cho việc thiết kế CSDL phân tán. Tuy nhiên nó là bài toán NP đầy đủ. Không có mô hình Heuristic tổng quát nào nhận một tập các mảnh và sinh ra một chiến lược cấp phát gần tối ưu với các ràng buộc ở trên. Vì vậy bài toán cấp phát tối ưu hoá chi phí sẽ được trình bày dưới dạng một mô hình tổng quát và một số heuristic có thể được sử dụng để giải quyết bài toán này.

2.6.2 Thông tin cần thiết cho bài toán cấp phát

Cần xác định các thông tin về CSDL, thông tin về các ứng dụng trên CSD, cấu trúc mạng, khả năng xử lý và giới hạn lưu trữ trên mỗi một vị trí cảu mạng.

a) Thông tin về CSDL: Để thực hiện việc phân mảnh ngang, cần định nghĩa độ tuyển hội sơ cấp. Mở rộng định nghĩa này cho các mảnh.

- Ký hiệu là $sel_i(F_j)$ là độ tuyển của mảnh F_j ứng với truy vấn q_i. là số lượng các bộ F_j được truy nhập để xử lý q_i.
- Kích thước của một mảnh F_j được định nghĩa bởi: $size(F_j) = card(F_j) * length(F_j)$, Trong đó $length(F_j)$ là chiều dài (tính theo byte) của một bộ trong mảnh F_j.

b) Thông tin về ứng dụng:

- Ký hiệu là RR_{ij} là số truy nhập đọc do truy vấn q_i thực hiện trên mảnh F_j.
- Ký hiêu UR_{ij} tương ứng với RR_{ij} là các truy nhập cập nhật
- Ma trận UM gồm các phần tử tương ứng u_{ij} như sau:

$$u_{ij} = \begin{cases} 1 & \text{Nếu truy vấn } q_i \text{ có cập nhật mảnh } F_j \\ 0 & \text{Trong trường hợp ngược lại.} \end{cases}$$

- Ma trận RM gồm các phần tử tương ứng r_{ij} như sau:

$$r_{ij} = \begin{cases} 1 & \text{Nếu truy vấn qi cần đọc mảnh } F_j \\ 0 & \text{Trong trường hợp ngược lại.} \end{cases}$$

- Một vectơ O gồm các giá trị o(i) mô tả trạm đưa ra câu truy vấn q_i.

c) Thông tin về vị trí: Cần phải biết khả năng lưu trữ và xử lý của mỗi một vị trí. Những giá trị này có thể tính được bằng các hàm thích hợp hoặc bằng các phương pháp đánh giá đơn giản. Chi phí đơn vị để lưu trữ dữ liệu tại trạm S_k được ký hiệu là USC_k. Chi phí xử lý một công việc tại vị trí S_k là LPC_k. Đơn vị công việc phải giống với đơn vị của RR và UR.

d) Thông tin về mạng: Giả sử tồn tại một mạng đơn giản, chi phí để truyền được định nghĩa theo đơn vị *khung dữ liệu*. Ký hiệu g_{ij} là chi phí truyền một khung giữa hai vị trí S_i và S_j. Hàm fsize tính được kích thước tính theo byte của một khung dữ liệu.

2.6.3 Mô hình cấp phát

Mục tiêu của mô hình cấp phát là giảm tối thiểu tổng chi phí xử lý và lưu trữ, đáp ứng được các đòi hỏi về thời gian đáp ứng. Mô hình có dạng sau:

min(Total Cost)

Ứng với ràng buộc thời gian đáp ứng, ràng buộc lưu trữ, ràng buộc xử lý.

Biến quyết định x_{ij} được định nghĩa là:

$$x_{ij} = \begin{cases} 1 & \text{Nếu mảnh } F_i \text{ được lưu tại vị trí } S_j \\ 0 & \text{Ngược lại.} \end{cases}$$

a) *Hàm tổng chi phí* gồm hai thành phần: xử lý truy vấn và lưu trữ. Có thể được biểu diễn như sau:

$$TOC = \sum_{\forall q_i \in Q} QPC_i + \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} STC_{jk}$$

Trong đó QPC_i là chi phí xử lý truy vấn của ứng dụng q_i và STC_{jk} là chi phí lưu mảnh F_j tại trạm S_k.

- Chi phí lưu trữ : $STC_{jk} = USC_k * Size(F_j) * x_{jk}$ .Tổng chi phí lưu tại tất cả vị trí cho mọi mảnh.
- Chi phí xử lý truy vấn QPC cho ứng dụng q_i là: $QPC_i = PC_i + TC_i$ .Thành phần xử lý PC gồm ba hệ số chi phí: chi phí truy nhập AC, chi phí duy trì toàn vẹn IE và chi phí điều khiển đồng thời CC: $PC_i = AC_i + IE_i + CC_i$

$$AC_i = \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} (u_{ij} * UR_{ij} + r_{ij} * RR_{ij}) * x_{jk} * LPC_k$$

Hai số hạng đầu là số truy nhập của truy vấn q_i đến mảnh F_j. Biểu thức (URij + RRij) là tổng số các truy nhập đọc và cập nhật với giả thiết chi phí xử lý là như nhau. Ký hiệu tổng cho biết tổng các truy nhập cho tất cả các mảnh được q_i tham chiếu. Nhân với LPC_k sẽ cho chi phí truy nhập tại vị trí S_k. Nhân x_{jk} cho các giá trị chi phí các trạm lưu các mảnh.

Hàm chi phí truy nhập của việc xử lý một câu truy vấn bao gồm việc phân rã thành các tập con truy vấn. Mỗi truy vấn con hoạt tác trên một mảnh được lưu tại trạm đó. Tiếp theo là truyền kết quả về trạm đã đưa ra truy vấn. Ở đây mô hình cấp phát sẽ bỏ qua các vấn đề phức tạp của xử lý truy vấn.

Thành phần cập nhật của hàm truyền dữ liệu là:

$$TCU_i = \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} u_{ij} * x_{jk} * g_{o(i),k} + \sum_{\forall S_k \in S} \sum_{\forall F_j \in F} u_{ij} * x_{jk} * g_{h,o(i)}$$

Số hạng thứ nhất: thông báo cập nhật từ vị trí gốc o(i) của q_i đến tất cả các bản sao cần cập nhật. Số hạng thứ hai dành cho thông báo xác nhận.

Thành phần chi phí chỉ đọc có thể là:

$$TCR_i = \sum_{\forall F_j \in F} \min_{S_k \in S} (u_{ij} * x_{jk} * g_{o(i),k} + r_{ij} * x_{jk} \frac{sel_i(F_j) * length(F_j)}{fsize} * g_{h,o(i)})$$

- Số hạng thứ nhất trong TCR biểu diễn chi phí truyền yêu cầu chỉ đọc đến những vị trí có bản sao của mảnh cần truy nhập.

- Số hạng thứ hai là truyền các kết quả từ trạm này đến trạm yêu cầu. Phương trình này khẳng định rằng trong số các trạm có bản sao của cùng một mảnh, chỉ trạm sinh ra tổng chi phí truyền thấp nhất mới được chọn để thực hiện thao tác này.

Hàm chi phí truyền cho truy vấn q_i có thể được tính là:

$$TC_i = TCU_i + TCR_i$$

b) Ràng buộc

Các hàm ràng buộc có thể được đặc tả tương tự.

- Ràng buộc thời gian đáp ứng là ràng buộc về thời gian thực thi của $q_i \leq$ thời gian đáp ứng lớn nhất của q_i, $\forall q_i \in Q$. Thường đặc tả số đo chi phí của hàm theo thời gian , vì đặc tả đơn giản ràng buộc thời gian thực thi.
- Ràng buộc lưu trữ là:

$$\sum_{\forall F_j \in F} STC_{jk} \leq \text{khả năng lưu trữ tại trạm } S_k, \forall S_k \in S$$

- Ràng buộc xử lý là:

$$\sum_{\forall q_i \in Q} \text{tải trọng xử lý của } q_i \text{ tại trạm } S_k \leq \text{khả năng xử lý của } S_k, \forall S_k \in S$$

2.7 KIỂM SOÁT DỮ LIỆU NGỮ NGHĨA

Cơ sở dữ liệu không chỉ tập trung ở một nơi mà phân tán trên nhiều site khác nhau. Khi có một yêu cầu về cập nhật dữ liệu, làm thế nào để dữ liệu vẫn đảm bảo được tính nhất quán và toàn vẹn dữ liệu. Định nghĩa các quy tắc nhằm kiểm soát các thao tác dữ liệu là một trong những yêu cầu quan trọng của một hệ quản trị cơ sở dữ liệu tập trung hay phân tán. Kiểm soát dữ liệu ngữ nghiâ bao gồm:

- Quản lý khung nhìn.
- Đảm bảo an toàn, bảo mật dữ liệu
- Kiểm soát tính toàn vẹn ngữ nghĩa.

Trong chương này, các giải pháp kiểm soát dữ liệu ngữ nghĩa chỉ được đề xét trong môi trường tập trung mà không xét cho môi trường phân tán. Bởi vì chi phí truyền thông cao, sử dụng nguồn tài nguyên lớn.

2.8 QUẢN LÝ KHUNG NHÌN

Một khung nhìn dữ liệu là một cửa sổ động theo nghĩa là nó phán ánh các hoạt động cập nhật trên cơ sở dữ liệu, là một quan hệ ảo được định nghĩa như là một kết quả truy vấn trên quan hệ cơ sở hoặc trên các quan hệ trung gian. Quản lý khung nhìn có tác dụng bảo đảm được tính an toàn dữ liệu. Nó cung cấp một cách nhìn tổng quát và trực diện nhất về các thao tác đươc thực hiện trên CSDL. Người sử dụng chỉ được phép truy nhập CSDL qua khung nhìn, không thể nhìn thấy hoặc không thể thao tác trên các dữ liệu ẩn, vì vậy dữ liệu được bảo vệ.

2.8.1 Khung nhìn trong các hệ quản trị cơ sử dữ liệu tập trung

Một khung nhìn là một quan hệ được dẫn xuất từ các quan hệ nguồn như kết quả của một câu truy vấn. Khung nhìn được định nghĩa bằng cách gán tên của khung cho câu truy vấn.

Ví dụ 2.1: Xét câu truy vấn sau:

CREAT VIEW SYSAN(ENO,ENAME)

```
AS      SELECT    ENO, ENAME
        FROM      EMP
        WHERE     TITLE = "Syst. Anal,"
```

Kết quả câu truy vấn là định nghĩa một khung nhìn, một quan hệ gán bằng tên SYSAN gồm có các thuộc tính ENO và ENAME và các bộ là tên của các phân tích viên hệ thống "Syst. Anal,." và lưu vào bộ nhớ.

EMP

ENO	ENAME	TITLE
E1	J.Doe	Elect.Eng
E2	M.Smith	Syst Anal
E2	M.Smith	Syst Anal
E3	A.Lee	Mech.Eng
E3	A.Lee	Mech.Eng
E4	J.Miller	Programmer
E5	B.Casey	Syst.Anal
E6	L.Chu	Elect.Eng
E7	R.David	Mech.Eng
E8	J.Jones	Syst.Anal

(a) Quan hệ gốc

SYSAN

ENO	ENAME
E2	M.Smith
E5	B.Casey
E8	J.Jones

(b) Khung nhìn

Hình 2.18 Quan hệ kết quả truy vấn tương ứng với khung nhìn

Khung nhìn SYSAN là quan hệ ảo trung gian, vì vậy có thể thao tác câu truy vấn bất kỳ có liên quan đến quan hệ này:

Ví dụ 2.2: Tìm tên của các phân tích viên hệ thống cùng với mã dự án mà họ tham gia và nhiệm vụ của họ. Câu truy vấn SQL có liên quan đến khung nhìn SYSAN như sau:

```
SELECT    ENAME, PNO, RESP
FROM      ASG, SYSAN
WHERE     SYSAN.ENO=ASG.ENO
```

ASG

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

Có thể hiệu chỉnh câu truy vấn trong thí dụ 2.2 bằng cách các biến chạy trên các quan hệ có sở và lượng từ hoá truy vấn được nối với lượng từ hoá khung nhìn.

```
SELECT    ENAME, PNO, RESP
FROM      ASG, EMP
WHERE     EMP.ENO = ASG.ENO
          AND   TITLE = "Syst. Anal,"
```

ENAME	PNO	RESP
M.Smith	P1	Analyst
M.Smith	P2	Analyst
B.Casey	P3	Manager
J.Jones	P4	Manager

Hình 2.19 Kết quả truy vấn sử dụng khung nhìn

Khung nhìn được sử dụng để hạn chế người sử dụng truy xuất cơ sở dữ liệu, chỉ cho phép truy xuất phạm vi dữ liệu nhất định.

Ví dụ 2.3: Xét câu truy vấn sau:

```
CREAT     VIEW      ESAME
AS        SELECT    *
          FROM      EMP  E1, EMP E2
          WHERE     E1.TITLE  =  E2.TITLE
                    AND   E1.ENO  =  USER
```

Khung nhìn ESAM chỉ cho phép người sử dụng truy xuất thông tin của nhân viên có cùng nhiệm vụ. Trong định nghĩa khung nhìn ESAM, ký tự "*" nghĩa là tất cả các thuộc tính trong quan hệ EMP và các biến E1 và E2 được gán tên cho quan hệ EMP thành hai qiuan hệ E1 và E2, biểu diễn phép kết nối E1 và E2.trên TITLE nhiệm vụ như nhau. Chẳng hạn câu truy vấn sau sẽ trả về quan hệ kết quả bao gồm các nhân viên có cùng nhiệm vục với J. Doe (hình 2.3). Nếu người tạo ra ESAM là một kỹ sư điện thì kết quả khung nhìn biểu thị các nhân viên cũng là kỹ sư điện.

```
SELECT    *
FROM      ESAM
```

ENO	ENAME	TITLE
E1	J.Doe	Elect.Eng
E6	L.Chu	Elect.Eng

Hình 2.20 Kết quả vấn tin khung nhìn ESAM

2.8.2 Cập nhật qua khung nhìn

Khung nhìn có thể được định nghĩa bằng các câu truy vấn bao gồm các phép chiếu, chọn, kết nối hoặc bằng các hàm gộp nhóm...và được truy vấn như là một quan hệ cơ sở. Cập nhật qua khung nhìn được xử lý tự động nếu chúng được lan truyền chính xác đến các quan hệ cơ

sở. Có hai loại khung nhìn: loại khung nhìn cập nhật được và loại khung nhìn không cập nhật được. Khung nhìn cập nhật được là khung nhìn nếu khi thực hiện các phép cập nhật trên nó sẽ lan truyền chính xác đến các quan hệ cơ sở mà không có sự nhầm lẫn nào. Khung nhìn SYSAN trong các thí dụ ở trên là khung nhìn cập nhật được, vì khi thực hiện việc chèn thêm thông tin về một phân tích viên mới vào khung nhìn SYSAN, hệ thống sẽ ánh xạ thành thao tác chèn thông tin về một nhân viên mới vào quan hệ EMP. Nếu các thuộc tính bị che khuất khung nhìn, chúng có thể nhận giá trị không hoặc null.

Các hệ thống hỗ trợ cập nhật được qua khung nhìn rất hạn chế. Các khung nhìn chỉ có thể cập nhật được khi chúng được dẫn xuất từ một quan hệ duy nhất bằng phép chọn hoặc phép chiếu. Các khung nhìn được định nghĩa bởi phép kết nối hay các phép gộp nhóm thường không thuộc loại cập nhật được.

Xét khung nhìn sau đây:

```
CREAT   VIEW    EG(ENAME, RESP)
AS      SELECT  ENAME, RESP
        FROM    EMP, ASG
        WHERE   EMP.ENO = ASG.ENO
```

Khung nhìn EG thuộc laọi khung nhìn khung cập nhật được. Vì khi thực hiện phép xoá bộ <Smith, Syst. Anal> qua khung nhìn EG sẽ không lan truyền được, bởi vì xoá "Smith" trong quan hệ EMP hoặc xoá "Syst, Anal"" trong quan hệ ASG là có nghĩa và rỗ ràng, nhưng hệ thống không thể phân biệt như thế nào là đúng, lệnh xoá không rõ ràng.

Khung nhìn được dẫn xuất từ phép kết nối có thể là khung nhìn cập nhật được nếu có chứa khoá của các quan hệ cơ sở.

2.8.3 Khung nhìn trong các hệ quản trị cơ sở dữ liệu phân tán

Định nghĩa khung nhìn trong các hệ quản trị cơ sở dữ liệu phân tán giống như trong các hệ quản trị cơ sở dữ liệu tập trung. Tuy nhiên, trong các hệ thống phân tán, các quan hệ cơ sở dẫn xuất là các mảnh quan hệ được lưu trữ trên các vị trí khác nhau. Vì vậy định nghĩa khung nhìn, tên khung nhìn , cấu trúc của khung nhìn và câu truy vấn truy xuất của các mảnh được sử dụng làm các quan hệ cơ sở trong các ứng dụng phải được lưu trữ trong các thư mục mô tả quan hệ cơ sở. Phụ thuộc vào mức độ tự trị của vị trí, các định nghĩa khung nhìn có thể tập trung tại một vị trí, hoặc được nhân bản một phần hoặc toàn bộ. Trong các trường hợp, thông tin liên kết khung nhìn với vị trí định nghĩa của nó phải được nhân bản. Nếu định nghĩa khung nhìn không có tại vị trí của câu truy vấn thì phải truy xuất từ xa đến vị trí có định nghĩa khung nhìn.

Ánh xạ một câu truy vấn được diễn tả theo khung nhìn thành một câu truy vấn được diễn tả theo quan hệ cơ sở giống như trong các hệ thống tập trung. Nghĩa là qua phương pháp hiệu chỉnh. Vì vậy lượng từ hoá dùng để định nghĩa khung nhìn được lấy từ thư mục phân tán, được trộn với câu truy vấn thành câu truy vấn theo các quan hệ cơ sở. Một câu truy vấn như vậy được gọi là câu truy vấn phân tán.

Khung nhìn trong các hệ thống phân tán được dẫn xuất từ các quan hệ phân tán, vì vậy có thể có chi phí cao khi ước lượng. Một khung nhìn có trhể có nhiều người cùng sử dụng, vì vậy cần đưa ra các giải pháp nhằm tối ưu hoá dẫn xuất khung nhìn.

2.9 AN TOÀN DỮ LIỆU

An toàn dữ liệu là nhiệm vụ quan trọng của các hệ thống cơ sở dữ liệu, nhằm bảo vệ dữ liệu không bị truy xuất "bất hợp pháp". An toàn dữ liệu bao gồm 2 vấn đề

- Bảo vệ dữ liệu: nhằm tránh những người không được quyền hiểu được nội dung vật lý của dữ liệu.. Phương pháp sử dụng thông dụng nhất là mã hoá dữ liệu. Mã khoá bí mật và mã khoá công khai.
- Biện pháp kiểm soát cấp quyền nhằm đảm bảo rằng chỉ những người sử dụng được phép mới có thể được thực hiện các thao tác được phép trên cơ sở dữ liệu. Cấp quyền truy xuất cơ sở dữ liệu cho người sử dụng là người quản trị cơ sở dữ liệu. Người sử dụng khác nhau được cấp các quyền khác nhau dưới sự kiểm soát của hệ thống. Từ các giải pháp kiểm soát cấp quyền.trong các hệ thống tập trung có thể đề xuất các giải pháp kiểm soát cấp quyền phân tán.

2.9.1 Kiểm soát cấp quyền tập trung

Ba tác nhân chính có ảnh hưởng đến việc kiểm soát cấp quyền truy xuất cơ sở dữ liệu của người sử dụng là: Người sử dụng, người quản trị cơ sở dữ liệu kích hoạt các trình ứng dụng, các thao tác gắn với các ứng dụng và tác nhân cuối cùng là các đối tượng cơ sở dữ liệu được các thao tác tác động. Kiểm soát cấp quyền bao gồm việc kiểm soát người sử dụng, các thao tác, đối tượng có được phép thực hiện hay không. Nghĩa là người sử dụng có thể thực hiện thao tác trên các đối tượng đó hay không. Một quyền được cấp (Authorization) được xem là một bộ ba thành phần: Người sử dụng, loại thao tác và định nghĩa đối tượng. Nghĩa là người sử dụng được quyền thao tác gì trên các đối tượng nào.

Khai báo người sử dụng, hay nhóm người sử dụng với hệ thống thường được thực hiện bằng một cặp: tên người sử dụng (User name), mật khẩu (Password).Tên người sử dụng xác định duy nhất một người sử dụng có tên trong hệ thống. Mật khẩu xác nhận người sử dụng được quyền truy nhập vào cơ sở dữ liệu. Tên và mật khẩu phải khai báo khi đăng nhập vào hệ thống, nhằm ngăn chặn truy nhập vào hệ thống bất hợp pháp

Đối tượng cần bảo vệ là các tập con của cơ sở dữ liệu. Trong hệ thống quan hệ, đối tượng cần được bảo vệ có thể là định nghĩa của quan hệ, khung nhìn, bộ, thuộc tính... và nội dữ liệu. Hơn nữa, cơ chế khung nhìn cho phép bảo vệ các đối tượng làm ẩn đi các tập con của quan hệ (thuộc tính hoặc bộ) đối với người sử dụng không được phép truy xuất..

Quyền hạn (Right) biểu thị mối liên hệ giữa người sử dụng và một đối tượng ứng với một tập các thao tác cụ thể. Trong các hệ quản trị cơ sở dữ liệu dựa trên SQL, một thao tác là một câu lệnh bậc cao như SELECT, INSERT, UPDATE hoặc DELETE, và các quyền được định nghĩa hoặc trao quyền hoặc thu hồi quyền bằng các câu lệnh:

GRANT <kiểu thao tác> ON <đối tượng> TO <người sử dụng>
REVOKE <kiểu thao tác> FROM <đối tượng> TO <người sử dụng>

Từ khóa Public để chỉ tất cả mọi người sử dụng. Điều khiển cấp quyền có thể được đặc trưng dựa vào người cấp quyền (Grantor). Ở dạng đơn giản nhất, việc điều khiển có thể được tập trung vào một người hoặc một nhóm người đóng vai trò là nhà quản trị dữ liệu sẽ có mọi quyền hạn trên các đối tượng CSDL và có quyền sử dụng các câu lệnh cấp quyền GRANT và thu hồi quyền đã cấp REVOKE. Phức tạp nhưng linh hoạt hơn là điều khiển không tập trung.

Người tạo ra đối tượng là chủ của đối tượng và được trao tất cả mọi quyền trên đối tượng đó. GRANT là trao cho quyền mọi quyền cho người người sử dụng đã được mô tả. Người nhận quyền có thể tiếp tục trao quyền cho người sử dụng khác trên các đối tượng đó. Quá trình thu hồi quyền phải thực hiện đệ quy, gặp nhiều khó khăn. Hệ thống phải duy trì một cây phân cấp chứa các hoạt động trao quyền cho mỗi đối tượng, trong đó chủ của đối tượng chính là gốc.

Quyền hạn của các chủ thể trên các đối tượng là những người sử dụng nhận được quyền trên các đối tượng được lưu trữ dưới dạng các qui tắc cấp quyền. Thuận tiện nhất là xem các quyền như là một ma trận cấp quyền (Authorization Matrix), trong đó hàng là chủ thể, và cột là đối tượng, và phần tử ma trận là các thao tác được phép được xác định bằng kiểu thao tác (ví dụ SELECT, UPDATE). Thường kèm với mỗi kiểu thao tác có một vị từ hạn chế thêm khả năng truy nhập đến đối tượng. Tùy chọn này được cung cấp với các đối tượng là các quan hệ cơ sở, không dành cho các khung nhìn. Ví dụ, một thao tác được phép cho cặp <Jones, quan hệ EMP> có thể là:

SELECT WHERE TITLE = “Syst. Anal.”

cho phép Jones chỉ được phép truy nhập đến các bộ của các phân tích viên hệ thống. Hình 2.22 là ví dụ mẫu về ma trận cấp quyền, trong đó đối tượng là các quan hệ EMP và ASG hoặc là các thuộc tính ENAME..

	EMP	ENAME	ASG
Casey	UPDATE	UPDATE	UPDATE
Jones	SELECT	SELECT	SELECT WHERE RESP ≠ ‘Manager’
Smith	NONE	SELECT	NONE

Hình 2.21 Ví dụ về ma trận cấp quyền

Ma trận cấp quyền có thể được lưu trữ theo ba cách: theo cột, theo hàng hoặc theo phần tử. Khi ma trận được lưu theo hàng, mỗi chủ thể được liên kết với một danh sách các đối tượng được phép truy nhập cùng với các quyền truy nhập tương ứng. Cách tiếp cận này cho phép duy trì các cấp quyền một cách hiệu quả, vì tất cả các quyền của một người sử dụng khi truy nhập vào hệ thống đều được lưu trữ cùng nhau trong hồ sơ cá nhân (Profile) của người sử dụng. Tuy nhiên, việc thao tác trên các quyền truy nhập (ví dụ cho phép mọi người truy nhập đến đối tượng) sẽ không hiệu quả vì phải truy nhập đến tất cả các hồ sơ cá nhân. Nếu ma trận được lưu theo cột, mỗi đối tượng được liên kết với một danh sách người sử dụng được phép truy nhập. Ưu và nhược điểm cũng như lưu theo hàng.

Ưu điểm của hai cách tiếp cận trên được tổ hợp trong cách tiếp cận thứ ba, trong đó ma trận được lưu theo phần tử, nghĩa là theo quan hệ (chủ thể, đối tượng, quyền). Quan hệ này có thể có chỉ mục trên cả chủ thể và đối tượng, qua đó cho phép truy nhập nhanh đến các quyền của mỗi chủ thể và đối tượng.

2.9.2 Kiểm soát cấp quyền phân tán

Các vấn đề kiểm soát cấp quyền trong môi trường phân tán bao gồm: cấp quyền cho người sử dụng ở xa, quản lý các quy tắc cấp quyền phân tán và việc xử lý khung nhìn và nhóm người sử dụng.

Cấp quyền cho người sử dụng ở xa nhằm ngăn chặn truy nhập từ xa trái phép, nghĩa là từ một vị trí không nằm trong hệ quản trị CSDL phân tán. Người sử dụng cũng cần phải được nhận diện và xác nhận tại vị trí được truy nhập. Có hai giải pháp cho vấn đề này:

1. Thông tin xác nhận người sử dụng bao gồm tên truy nhập và mật khẩu được nhân bản tại tất cả các vị trí. Các chương trình cục bộ, được khởi hoạt từ một vị trí ở xa cũng phải chỉ rõ tên và mật khẩu của người sử dụng.
2. Tất cả các vị trí trong hệ thống phân tán phải nhận diện và xác nhận nhau tương tự như người sử dụng. Các vị trí giao tiếp với nhau bằng tên và mật khẩu.

Giải pháp (1) có chi phí cao hơn tính theo công việc quản lý thư mục nếu việc đưa thêm một người sử dụng mới vào là một thao tán phân tán. Tuy nhiên, người sử dụng có thể truy nhập CSDL phân tán từ bất kỳ một vị trí nào. Giải pháp (2) là cần thiết khi thông tin người sử dụng không được nhân bản. Tuy vậy, nó cũng có thể sử dụng cấp quyền từ xa có hiệu quả. Nếu tên và mật khẩu của người sử dụng không được nhân bản, nhưng phải được lưu tại vị trí người sử dụng truy nhập vào hệ thống.

Các quy tắc cấp quyền phân tán cũng như các quy tắc cấp quyền trong các hệ tập trung. Các định nghĩa khung nhìn phải được lưu trữ. Có thể nhân bản hoàn toàn tại mỗi vị trí hoặc lưu trữ tại các vị trí của các đối tượng cần truy xuất. Ưu điểm của phương pháp tiếp cận nhân bản hoàn toàn là cấp quyền có thể được xử lý bằng kỹ thuật hiệu chỉnh truy vấn. Tuy nhiên việc quản lý thư mục sẽ tốn kém. Giải pháp lưu trữ tại các vị trí của các đối tượng cần truy xuất tốt hơn trong trường hợp tính chất cục bộ của tham chiếu rất cao.Tuy nhiên việc cấp quyền phân tán không thể kiểm soát được vào thời điểm biên dịch.

Khung nhìn có thể được coi mhư các đối tượng qua cơ chế cấp quyền. Khung nhìn được cấu tạo bởi các đối tượng cơ sở khác. Vì vậy khi trao quyền truy xuất đến một khung nhìn được dịch thành trao quyền truy xuất đến các đối tượng cơ sở. Nếu định nghĩa khung nhìn và các quy tắc cấp quyền được nhân bản hoàn toàn, thì việc phiên dịch khá đơn giản và được thực hiện tại chỗ. Phức tạp hơn khi định nghĩa khung nhìn và các đối tượng cơ sở của nó được lưu riêng thì phiên dịch sẽ là một thao tác hoàn toàn phân tán. cấp quyền được trao trên khung nhìn phụ thuộc vào quyền truy xuất của chủ nhân khung nhìn trên các đối tượng cơ sở. giải pháp ghi nhận thông tin liên kết tại các vị trí của mỗi đối tượng cơ sở.

Nhóm người sử dụng nghĩa là cấp quyền truy xuất chung cho nhiều người, với mục đích làm đơn giản hoá công việc quản lý cơ sở dữ liệu. Trong các hệ quản trị cơ sở dữ liệu tập trung, khái niệm mọi người sử dụng có thể đồng nhất với nhóm người sử dụng. Trong môi trường phân tán, mhóm người sử dụng biểu thị cho tất cả người sử dụng tại một vị trí cụ thể, được biểu thị public@site_s, là nhóm đặc biệt, được định nghĩa bởi lệnh sau:.

DEFINE GROUP <group_id> AS <Danh sách các id chủ thể>

Vì trong môi trường phân tán, các chủ thể, các đối tượng phân tán tại nhiều vị trí khác nhau và quyền truy xuất thông tin đến một đối tượng có thểuto cho nhiều nhóm phân tán khác

nhau. Vì vậy vấn đề quản lý nhóm trong môi trường phân tán có một số vấn đề cần giải quyết. Nếu thông tin của nhóm và các quy tắc cấp quyền được nhân bản hoàn toàn tại tất cả các vị trí, thì việc duy trì quyền truy xuất tương tự như trong các hệ thống tập trung. Tuy nhiên việc duy trì các bản sao là tốn kém. Việc kiểm soát phi tập trung, tức là duy trì sự hoạt động tự trị vị trí sẽ khó khăn và phức tạp hơn rất nhiều.

Nhân bản hoàn toàn cho các thông tin cấp quyền có ưu điểm là kiểm soát cấp quyên đơn giản hơn và có thể thực hiện vào lúc biên dịch. Tuy nhiên, chi phí cho việc quản lý phân tán sẽ quá cao, nếu có rất nhiều vị trí trong hệ thống.

2.10 KIỂM SOÁT TÍNH TOÀN VẸN NGỮ NGHĨA

Một vấn đề quan trọng và khó khăn cho một hệ CSDL là bảo đảm được tính nhất quán cơ sở dữ liệu (Databasse Consistency). Một trạng thái CSDL được gọi là nhất quán nếu nó thỏa một tập các ràng buộc, được gọi là ràng buộc toàn vẹn ngữ nghĩa (Semantic Integrity Constrsint). Đảm bảo tính nhất quán của CSDL, kiểm soát toàn vẹn ngữ nghĩa bằng cách loại bỏ hoặc hoá giải các trình cập nhật làm cho CSDL không nhất quán. CSDL đã cập nhật nghĩa là đã thỏa tập các ràng buộc toàn vẹn.

Có hai loại ràng buộc toàn vẹn: ràng buộc cấu trúc (Structural Constraint) và ràng buộc hành vi (Behavioral Constraint). Ràng buộc cấu trúc mô tả những đặc tính ngữ nghĩa cơ bản của mô hình. Ví dụ như ràng buộc khóa trong mô hình quan hệ, hoặc các liên kết một-nhiều giữa các đối tượng trong mô hình mạng. Ngược lại, ràng buộc hành vi mô tả mối liên kết giữa các đối tượng, như khái niệm phụ thuộc hàm trong mô hình quan hệ.

2.10.1 Kiểm soát toàn vẹn ngữ nghĩa tập trung

Một tiểu hệ thống kiểm soát toàn vẹn ngữ nghĩa có hai thành phần chính: một ngôn ngữ cho phép diễn tả và thao tác các phán đoán toàn vẹn, và một định chế chịu trách nhiệm thực hiện các hành động cụ thể nhằm ép buộc tính toàn ven khi có cập nhật.

*a) Các loại ràng buộc:*Bằng một ngôn ngữ cấp cao, người quản trị cơ sở dữ liệu có thể thiết lập được các ràng buộc toàn vẹn, khi tạo quan hệ hoặc khi quan hệ đã có dữ liệu, có chung một cú pháp. Ngôn ngữ này cho phép mô tả, ghi nhận hoặc loại bỏ ràng buộc toàn vẹn. Trong các hệ CSDL quan hệ, ràng buộc toàn vẹn được định nghĩa như là các phán đoán. Một phán đoán (Assertion) là một biểu thức đặc biệt được mô tả bằng phép tính quan hệ bộ, trong đó mỗi biến được lượng từ hóa với mọi ($\forall$) hoặc tồn tại ($\exists$). Vì vậy, một phán đoán có thể được xem như là một lượng từ hóa câu truy vấn, mang giá trị đúng hoặc sai cho mỗi bộ trong tích Đề các của các quan hệ được xác định bởi các biến bộ. Có ba loại ràng buộc toàn vẹn: ràng buộc tiền định, ràng buộc tiền dịch hoặc ràng buộc tổng quát.

Xét các qun hệ:

EMP(ENO, ENAME, TITLE)
PROJ(PNO, PNAME, BUDGET)
ASG(ENO, PNO, RESP, DUR)

Ràng buộc tiền định (Predefined Constraint): Ràng buộc tiền định định nghĩa dựa trên các từ khóa đơn giản để diễn tả chính xác các ràng buộc trong mô hình quan hệ, chẳng hạn như

ràng buộc giá trị của thuộc tính không nhận giá trị null, hoặc khóa duy nhất, khóa ngoại hoặc phụ thuộc hàm....

Ví dụ 2.4: Ràng buộc tiền định

- Thuộc tính không nhận giá trị null: Mã số nhân viên trong quan hệ EMP không được nhận giá trị null.

  ENO NOT NULL IN EMP
- Khóa duy nhất: Cặp (ENO, PNO) là khóa duy nhất của quan hệ ASG.

  (ENO, PNO) UNIQUE IN ASG
- Khóa ngoại: Mã dự án PNO trong quan hệ ASG là khóa ngoại tương ứng với khóa chính PNO của quan hệ PROJ. Nói cách khác, một dự án được tham chiếu trong quan hệ ASG phải tồn tại trong quan hệ PROJ.

  PNO IN ASG REFERENCES PNO IN PROJ
- Phụ thuộc hàm: Tên nhân viên phụ thuộc hàm Mã nhân viên trong quan hệ EMP

  ENO IN EMP DETERMINES ENAME

Ràng buộc tiền dịch (Precompiled Constraint): Ràng buộc tiền dịch mô tả các điều kiện mà các bộ trong một quan hệ phải được thỏa bởi một kiểu cập nhật đã cho. Kiểu cập nhật có thể là INSERT, DELETE hoặc MODIFY cho phép giới hạn kiểm soát ràng buộc. Để xác định các bộ cần cập nhật trong định nghĩa ràng buộc, người ta đưa ra hai biến NEW và OLD, tương ứng biến thiên trên các bộ mới (được chèn vào) và các bộ cũ (được xóa bỏ). Cú pháp ràng buộc tiền dịch được biểu diễn bởi câu lệnh sau:

CHECK ON <quan hệ> WHEN <kiểu cập nhật> (<lượng từ hóa trên quan hệ>)

- Ràng buộc miền: Ngân sách của một dự án trong khoảng từ 500000 đến 1000000.
  CHECK ON PROJ (BUDGET ≥ 500000 AND BUDGET ≤ 1000000)
- Ràng buộc miền khi xóa: Xoá các bộ có ngân sách không:
  CHECK ON PROJ WHEN DELETE (BUDGET = 0)
- Ràng buộc thay đổi: Ngân sách của dự án chỉ được tăng khi:
  CHECK ON PROJ (NEW. BUDGET > OLD. BUDGET
  AND NEW. PNO = OLD. PNO)

Ràng buộc tổng quát (General Constraint): Ràng buộc tổng quát là các công thức của phép tính quan hệ bộ trong đó tất cả các biến đều được lượng từ hóa. Hệ thống CSDL phải đảm bảo rằng những công thức này phải luôn đúng. Ràng buộc tổng quát chuẩn xác hơn ràng buộc tiền dịch vì có thể liên quan đến nhiều quan hệ. Có thể sử dụng ít nhất ba ràng buộc tiền dịch để diễn tả một ràng buộc tổng quát trên ba quan hệ. Một ràng buộc tổng quát có thể được diễn tả với cú pháp như sau:

CHECK ON danh sách <tên biến> : <tên quan hệ>, (<lượng từ hóa>)

Ví dụ 2.5: Ràng buộc tổng quát.

- Phụ thuộc hàm: Tên nhân viên phụ thuộc hàm Mã nhân viên trong quan hệ EMP:
  CHECK ON e1:EMP, e2:EMP
  (e1.ENAME = e2.ENAME IF e1.ENO = e2.ENO)
- Ràng buộc có kèm hàm gộp nhóm: Tổng thời gian của các nhân viên trong dự án CAD/CAM phải nhỏ hơn 100
  CHECK ON g:ASG, j:PROJ (SUM(g.DUR WHERE g.PNO = j.PNO) < 100

IF j.PNAME = “CAD/CAM”).

b) Ép buộc thực thi ràng buộc: Ép buộc thực thi ràng buộc toàn vẹn nghĩa là thực hiện việc loại bỏ những chương trình cập nhật vi phạm một số ràng buộc nào đó. Một ràng buộc bị vi phạm do các hành động cập nhật hoặc do các phán đoán ràng buộc bị sai. Có hai phương pháp cơ bản cho phép loại bỏ các trình cập nhật phát sinh mâu thuẫn.

Phương pháp phát hiện mâu thuẫn (không nhất quán): Một thao tác cập nhật *u* được thi hành sẽ biến đổi CSDL từ trạng thái D sang trạng thái D_u..Thuật toán ép buộc phải khẳng định rằng mọi ràng buộc liên quan vẫn đúng trong D_u bằng cách áp dụng các ràng buộc này để kiểm tra. Nếu trạng thái D_u không nhất quán, hệ quản trị CSDL sẽ chuyển sang một trạng thái $D_{u'}$ khác bằng cách hiệu chỉnh lại D_u hoặc phải khôi phục lại trạng thái D. Vì những kiểm tra này được áp dụng sau khi trạng thái của CSDL đã thay đổi nên được gọi là kiểm tra sau (Posttest). Phương pháp này sẽ không hiệu quả nếu hệ thống phản hồi lại rất nhiều thao tác khi ràng buộc bị vi phạm.

Phương pháp ngăn chặn mâu thuẫn: Một thao tác chỉ được thực hiện nếu nó chuyển CSDL sang một trạng thái nhất quán khác. Các bộ cần cập nhật đã có sẵn (trong trường hợp chèn) hoặc phải truy nhập trong CSDL (trường hợp xóa hoặc hiệu chỉnh). Thuật toán ép buộc xác nhận rằng tất cả các ràng buộc có liên đới đều đúng sau khi cập nhật các bộ. Nói chung, nó được thực hiện bằng cách áp dụng các kiểm tra có được từ các ràng buộc cho các bộ. Như vậy, các kiểm tra được áp dụng trước khi trạng thái của CSDL bị thay đổi, nên gọi là các kiểm tra trước (Pretest). Phương pháp ngăn chặn này hiệu quả hơn phương pháp phát hiện vì không bao giờ phải hồi lại các thao tác cập nhật do ràng buộc bị vi phạm.

Thuật toán hiệu chỉnh truy vấn là một ví dụ về phương pháp ngăn chặn, có hiệu quả đặc biệt trong việc ép buộc các ràng buộc miền biến thiên. Nó đưa thêm lượng từ hóa phán đoán vào lượng từ hóa truy vấn bằng toán tử AND, vì thể câu truy vấn được hiệu chỉnh có thể được ép buộc toàn vẹn.

Ví dụ 2.6: Tăng ngân sách dự án CAD/CAM lên 10%:

```
UPDATE    PROJ
SET       BUDGET = BUDGET * 1.1
WHERE     PNAME = “CAD/CAM”
```

Câu truy vấn trên sẽ được biến đổi thành câu truy vấn sau, có thêm điều kiện nhằm ép buộc miền giá trị

```
UPDATE    PROJ
SET       BUDGET = BUDGET * 1.1
WHERE     PNAME = “CAD/CAM”
AND       NEW. BUDGET ≥ 500000
AND       NEW. BUDGET ≤ 1000000
```

Thuật toán hiệu chỉnh truy vấn đơn giản, tạo ra các kiểm tra trước vào thời điểm thi hành bằng cách lấy hội các vị từ phán đoán với các vị từ cập nhật của mỗi chỉ thị của giao dịch. Tuy nhiên, thuật toán chỉ được áp dụng cho các công thức phép tính bộ và có thể mô tả như sau: Xét phán đoán $(\forall x \in R)\ F(x)$ trong đó R là quan hệ, x là biến bộ của R. Thao tác cập nhật của R có thể viết như sau: $(\forall x \in R)\ Q(x) \Rightarrow$ Update(x), Trong đó Q(x) là biểu thức phép tính bộ có biến là x.. Nói ngắn gọn hiệu chỉnh truy vấn cần sinh ra thao tác cập nhật $(\forall x \in R)\ ((Q(x)$ and $F(x)) \Rightarrow$ Update(x). Vì vậy, x cần phải được lượng từ hoá phổ dụng.

Ví dụ 2.7: Phán đoán cho khoá ngoại

$$\forall r \in ASG,\ \exists j \in PROJ:\ g.PNO = j.PNO$$

Không thể được xử lý bởi phép hiệu chỉnh truy vấn vì biến j không được lượng từ hoá phổ dụng.

Để có thể xử lý các phán đoán tổng quát, sự kiểm tra trước cần phải đượqọc xây dựng khi định nghĩa phán đoán và nó được thực hiện khi cập nhật. Phương pháp này có thể định nghĩa phán đoán đa quan hệ, đơn biến và có thể gộp nhóm bằng cách thay thế các biến bộ trong phán đoán bằng các từ một bộ được cập nhật. Tuy nhiên phương pháp này khó được sử dụng trong thực tế, vì .có nhiều hạn chế khi thực hiện các cập nhật.

Định nghĩa các phán đoán biên dịch dựa trên khái niệm quan hệ vi phân (Differential Relation). Goi u là một cập nhật trên quan hệ R. R^+ và R^- là các quan hệ vi phân của R do u, với R^+ chứa các bộ được chèn vào R và R^- là quan hệ chứa các bộ bị xoá khỏi R. Nếu u là thao tác chèn, khi đó R^- sẽ rỗng. Nếu u là thao tác xoá thì quan hệ R^+ cũng là rỗng. Nếu u là thao tác sửa đổi, quan hệ R sau khi sửa đổi sẽ là $R^+ \cup (R – R^-)$. Một phán đoán biên dịch là bộ ba (R, T, C). Trong đó R là một quan hệ, T là kiểu cập nhật và C là một phán đoán biến thiên trên các quan hệ vi phân có mặt trong kiểu cập nhật T. Khi một ràng buộc I được định nghĩa, một tập phán đoán biên dịch có thể được sinh ra cho các quan hệ được I tác động. Mỗi khi một quan hệ có trong I được cập nhật bởi một chương trình u, các phán đoán biên dịch được định nghĩa trên I cho kiểu cập nhật u cần phải được thẩm tra. Có hai ưu điểm về hiệu năng. Một là số lượng các phán đoán cần cưỡng chế thấp tối đa, vì chỉ có các phán đoán thuộc kiểu u mới được kiểm tra. Ưu điểm thứ hai là chi phí cưỡng chế thi hành một phán đoán biên dịch nhỏ hơn chi phí cưỡng chế I, vì các quan hệ vi phân nhỏ hơn nhiều so với các quan hệ cơ sở.

Phán đoán biên dịch có thể thu được bằng cách áp dụng các quy tắc biến đổi cho phán đoán gốc. Các quy tắc này dựa trên phân tích ngữ nghĩa của phấn đoán và các hoán vị lượng từ. Cho pháp thay thế các quan hệ cơ sở bằng các quan hệ vi phân, vì các phán đoán biên dịch đơn giản hơn phán đoán gốc. Quá trình tạo ra phán đoán biên dịch gọi là quá trình đơn giản hoá (Simplìtication).

Ví dụ 2.8: Xét biểu thức ràng buộc khoá ngoại:

$$\forall r \in ASG,\ \exists j \in PROJ:\ g.PNO = j.PNO$$

Phán đoán biên dịch đi kèm với ràng buộc này sẽ là:

(ASG, INSERT, C_1), (PROJ, DELETE, C_2) và (PROJ, MODIFY, C_3).

Trong đó: C_1 là: $\forall NEW \in ASG,\ \exists j \in PROJ:\ NEW.PNO = j.PNO$

C_2 là: $\forall g \in ASG,\ \forall OLD \in PROJ^-:\ NEW.PNO \neq j.PNO$

C_3 là: $\forall g \in ASG,\ \forall OLD \in PROJ, \exists NEW \in PROJ^+.:$

$g.PNO \neq OLD.PNO\ OR\ OLD = NEW.PNO.$

Thao tác xoá trên quan hệ ASG không gây ra bất kỳ một kiểm tra nào.

Có 3 loại phán đoán: phán đoán đơn quan hệ, phán đoán đa quan hệ và phán đoán có các hàm gộp nhóm. Thuật toán cưỡng chế sử dụng các phán đoán biên dịch và được chuyên biệt hoá theo từng loại phán đoán. Một chương trình cập nhật sẽ hiệu chỉnh tất cả các bộ của quan

hệ R, thoả một lượng từ hoá nào đó. Thuật toán cưỡng chế được thực hiện trong hai bước như sau:

- Bước 1: Tạo các quan hệ vi phân R^+ và R^-. từ quan hệ R.
- Bước 2: Truy xuất các bộ trong R^+ và R^-, lấy các bộ không thoả các phấn đoán biên dịch. Nếu không truy xuất được bộ nào, phán đoán sẽ có giá trị.

Ví dụ 2.9: Giả sử có thao tác xoá trên PROJ. Cưỡng chế (PRO, DELETE, C_2) tạo ra các câu lệnh sau:

Result ← Truy xuất tất cả các bộ của PROJ trong đó ¬ (C_2) đúng. Nếu Result rỗng, phán đoán đã được xác nhận bởi phép cập nhật khác..

2.10.2 Kiểm soát toàn vẹn ngữ nghĩa phân tán

Phần này trình bày các thuật toán đảm bảo tính toàn vẹn ngữ nghĩa của các CSDL phân tán, được mở rộng từ các phương pháp đơn giản trong phần kiểm soát toàn vẹn ngữ nghĩa tập trung. Giả thiết các thuật toán có tính đến yếu tố tự trị vị trí, nghĩa là mỗi vị trí có thể xử lý các câu truy vấn cục bộ và thực hiện việc kiểm soát dữ liệu như một hệ quản trị CSDL tập trung. Giả thiết này làm đơn giản việc mô tả phương pháp. Hai vấn đề chính của thiết kế một tiểu hệ thống kiểm soát toàn vẹn trong một hệ quả trị CSDL phân tán, đó là vấn đề định nghĩa và lưu trữ các phán đoán và vấn đề ép buộc thi hành các phán đoán này..

a) Định nghĩa các phán đoán toàn vẹn phân tán: Giả sử một phán đoán toàn vẹn được diễn tả bằng phép tính quan hệ bộ. Mỗi phán đoán được coi là một lượng từ hóa truy vấn, trong đó nó nhận giá trị đúng hoặc sai với mỗi bộ trong tích Đề các của các quan hệ được xác định bằng các biến bộ. Các phán đoán có thể liên quan đến dữ liệu lưu trên nhiều vị trí khác nhau. Vì vậy chọn vị trí lưu trữ phán đoán sao cho giảm thiểu chi phí kiểm tra toàn vẹn. Một chiến lược phán đoán toàn vẹn chia ra ba lớp:

1. *Phán đoán riêng*: là các phán đoán đơn biến đơn quan hệ. Chỉ đề cập đến các bộ được cập nhật, độc lập với phần còn lại của CSDL. Ràng buộc miền giá trị của ví dụ "Ngân sách của một dự án trong khoảng 500000 đến 1000000" là một phán đoán riêng.
2. *Phán đoán hướng tập hợp*: bao gồm các ràng buộc đa biến đơn quan hệ như phụ thuộc hàm (mã số nhân viên xác định tên nhân viên) và đa biến đa quan hệ như các ràng buộc khóa ngoại.
3. *Phán đoán có các hàm gộp*: đòi hỏi phải được xử lý đặc biệt do chi phí ước lượng hàm gộp. Phán đoán trong ví dụ 2.5 là một đại diện cho lớp phán đoán này.

Định nghĩa một phán đoán toàn vẹn có thể được bắt đầu tại một vị trí có lưu các quan hệ trong phán đoán. Quan hệ có thể bị phân mảnh, một vị từ phân mảnh là một trường hợp đặc biệt của phán đoán thuộc lớp 1. Các mảnh khác nhau của một quan hệ có thể có trên nhiều vị trí khác nhau. Vì vậy, định nghĩa một phán đoán toàn vẹn sẽ là một thao tác phân tán và được thực hiện qua hai bước. Bước đầu tiên biến đổi các phán đoán ở cấp cao thành các phán đoán biên dịch. Bước tiếp theo là lưu trữ các phán đoán này tùy theo lớp phán đoán. Các phán đoán thuộc lớp 3 được xử lý giống như các phán đoán thuộc lớp 1 hoặc 2, tùy thuộc vào đặc tính của chúng là riêng hay theo tập hợp.

Phán đoán riêng: Định nghĩa phán đoán được gửi đến tất cả vị trí lưu trữ mảnh của quan hệ có mặt trong phán đoán. Phán đoán phải tương thích với dữ liệu của quan hệ tại mỗi vị trí. Tính tương thích có thể được kiểm tra ở hai cấp: cấp vị từ và cấp dữ liệu. Trước tiên, tương

thích vị từ được xác nhận bằng cách so sánh vị từ phán đoán với vị từ mảnh. Một phán đoán C được coi là không tương thích với vị từ mảnh p nếu "C đúng" dẫn đến "p sai", và ngược lại thì được coi là tương thích. Nếu không tương thích tại một vị trí, định nghĩa phán đoán phải bị loại bỏ ở mức toàn cục vì các bộ của mảnh đó không thỏa mãn ràng buộc toàn vẹn này. Nếu có tương thích vị từ, phán đoán sẽ được kiểm tra ứng với thể hiện của mảnh. Nếu thể hiện đó không thỏa mãn phán đoán thì nó cũng bị loại bỏ ở mức toàn cục. Nếu tương thích, phán đoán sẽ được lưu lại tại mỗi vị trí. Việc kiểm tra tính tương thích được thực hiện cho các phán đoán biên dịch với kiểu cập nhật là "chèn" (các bộ trong các mảnh coi như là "được chèn vào").

Ví dụ 2.10: Xét quan hệ EMP phân mảnh ngang trên ba vị trí bởi các vị từ

P_1: 0 ≤ ENO ≤ "E3"

P_2: "E3" ≤ ENO ≤ "E6"

P_3: ENO > "E6"

Phán đoán miền biến thiên C: ENO < "E4". Như vậy phán đoán C tương thích với P_1 và P_2, nghĩa là nếu C đúng thì P_1 đúng.và nếu C đúng thì P_2 chưa chắc đã sai, nhưng không tương thích với P_3, vì nếu C đúng thì P_3 sai. Vì vậy phán đoán C buộc phải bị loại bỏ bởi vì các bộ tại vị trí 3 không thoả C. Vì vậy quan hệ EMP không thoả C.

Phán đoán hướng tập hợp: Phán đoán hướng tập hợp thuộc loại đa quan hệ, nghĩa là có các vị từ kết nối. Tuy nhiên mỗi phán đoán biên dịch chỉ được liên kết với một quan hệ. Vì vậy định nghĩa phán đoán có thể được gửi đến tất cả các vị trí chứa các mảnh được các biến này tham chiếu. Việc kiểm tra tính tương thích bao gồm các mảnh của quan hệ được sử dụng trong vị từ nối. Tương thích vị từ sẽ không có tác dụng vì không thể suy một vị từ mảnh P sai nếu phán đoán C (dựa trên vị từ kết nối) là đúng. Vì vậy, cần phải kiểm tra C theo dữ liệu, đòi hỏi phải nối mỗi mảnh của quan hệ R với tất cả các mảnh của quan hệ S, là hai quan hệ có trong vị từ kết nối. Như vậy chi phí sẽ rất cao như các phép kết nối. Cần phải tối ưu hóa bằng cách xử lý truy vấn phân tán. Ba tình huống theo mức chi phí:có thể xảy ra:

1. Các mảnh của R được dẫn xuất từ các mảnh của S dựa vào một kết nối nửa trên thuộc tính được dùng trong vị từ kết nối. Trong trường hợp này kiểm tra tương thích có chi phí thấp vì bộ của S đối sánh được với một bộ của R sẽ ở cùng một vị trí.
2. S được phân mảnh trên thuộc tính kết nối. Mỗi bộ của R phải được so sánh với tối đa một mảnh của S, vì giá trị thuộc tính nối của bộ thuộc R có thể được dùng để tìm vị trí mảnh tương ứng của S.
3. S không được phân mảnh trên thuộc tính kết nối. Mỗi bộ của R phải được so sánh với tất cả các bộ của R đều được đảm bảo tương thích thì phán đoán sẽ được lưu lại tại mỗi vị trí..

Ví dụ 2.11: Xét phán đoán biên dịch tập hợp (ASG, INSERT, C_1). Trong đó: C_1 là:

$$\forall NEW \in ASG,\ \exists j \in PROJ:\ NEW.PNO = j.PNO$$

Xét ba trường hợp sau:

1.. ASG được phân mảnh theo vị từ: $ASG \rhd\!\prec_{PNO} PROJ_i$, trong đó $PROJ_i$ là một mảnh của quan hệ PROJ. Trong trường hợp này, mỗi bộ NEW của ASG đã được cài đặt tại cùng vị trí với bộ J sao cho NEW.PNO = J.NEW. Vì vị từ phân mảnh cũng như vị từ của C_1, nên việc thẩm tra tương thích không mất chi phí truyền dữ liệu.

2. PROJ được phân mảnh ngang theo hai vị từ:

P_1: PNO < "P3"

P_2: PNO ≥ "p3".

Trong trường hợp này, mỗi bộ NEW của ASG được so sánh với mảnh $PROJ_1$ nếu NEW.PNO < "P3", hoặc với mảnh $PROJ_2$ nếu NEW.PNO ≥ "P3".

3. PROJ phân mảnh theo hai vị từ:

P_1: PNAME = "CAD/CAM"

P_2: PNAME = "CAD/CAM".

Trong trường hợp này, mỗi bộ của ASG phải so sánh với cả hai mảnh PRO_1 và $PROJ_2$.

b) Ép buộc thi hành các phán đoán toàn vẹn phân tán: Ép buộc thi hành các phán đoán toàn vẹn phân tán chính là quyết định xem vị trí nào sẽ thực hiện ép buộc. Việc quyết định phụ thuộc vào lớp phán đoán, kiểu cập nhật và bản chất của vị trí đưa ra yêu cầu cập nhật (được gọi là vị trí truy vấn chính). Vị trí được lựa chọn có thể có hoặc không có quan hệ cần cập nhật hoặc một số quan hệ có mặt trong phán đoán toàn vẹn. Cần phải xem xét chi phí truyền thông. Ép buộc thi hành các phán đoán toàn vẹn phân tán phức tạp hơn nhiều so với các hệ quản trị CSDL tập trung.

Phán đoán riêng: Hai trường hợp có thể xẩy ra: Trường hợp thứ nhất, nếu cập nhật là yêu cầu chèn, phán đoán riêng có thể bị ép buộc thi hành tại vị trí đưa ra cập nhật. Trường hợp nếu cập nhật là xóa hoặc sửa đổi, yêu cầu sẽ gửi đến các vị trí có quan hệ cần cập nhật. Xử lý truy vấn sẽ thực hiện bằng cách lượng từ hóa cập nhật cho mỗi mảnh. Các bộ được tạo ra tại các vị trí sẽ hợp lại thành một quan hệ tạm thời trong trường hợp câu lệnh xóa, hoặc hiệu chỉnh (nghĩa là R^+ và R^-). Mỗi vị trí có mặt trong cập nhật đều thảm tra các phán đoán có liên quan đến vị trí đó (ví dụ ràng buộc miền khi xóa).

Phán đoán hướng tập hợp: Xét phụ thuộc hàm ràng buộc đơn quan hệ Mã nhân viên xác định tên nhân viên: ENO IN EMP DETERMINES ENAME

Phán đoán biên dịch kèm với kiểu cập nhật INSERT là

(EMP, INSERT, C)

trong đó C là

$(\forall e \in EMP)\ (\forall NEW1 \in EMP)\ (\forall NEW2 \in EMP)$

$(NEW1.ENO = e.ENO \Rightarrow NEW1.ENAME = e.ENAME) \wedge$

$(NEW1.ENO = NEW2.ENO \Rightarrow NEW1.ENAME = NEW2.ENAME)$

Dòng thứ hai trong định nghĩa của C kiểm tra ràng buộc giữa các bộ được chèn (NEW1) và các bộ hiện có (e). Dòng thứ ba kiểm tra giữa các bộ được chèn. Vì vậy trong dòng thứ nhất cần thiết phải định nghĩa hai biến NEW1 và NEW2.

Xét một thao tác cập nhật EMP. Lượng từ hóa cập nhật được thực hiện bởi cách xử lý truy vấn và trả về một hoặc hai quan hệ tạm thời được gửi đến tất cả các vị trí có lưu các mảnh EMP. Giả sử cập nhật là một câu lệnh INSERT. Khi đó các vị trí có chứa mảnh EMP sẽ thi hành phán đoán C mô tả ở trên. Vì e trong C được lượng từ hóa phổ dụng nên dữ liệu cục bộ tại mỗi vị trí phải thỏa C. Điều này do sự kiện

$\forall x \in \{a_1, \ldots, a_n\} f(x)$ tương đương với $[f(a_1 \wedge \ldots \wedge f(a_n)]$.

Vì vậy, vị trí đưa ra yêu cầu cập nhật phải nhận được các thông báo từ các vị trí , cho biết phán đoán này thỏa mãn và là một điều kiện cho tất cả mọi vị trí . Nếu phán đoán không thỏa mãn tại một vị trí , thì vị trí này sẽ gửi thông báo lỗi cho biết phán đoán bị vi phạm. Vì vậy,

cập nhật sẽ không có giá trị, tiểu hệ thống sẽ kiểm tra toàn vẹn, quyết định toàn bộ chương trình phải loại bỏ hay không loại bỏ.

Xét các phán đoán đa quan hệ. Để đơn giản, giả sử các phán đoán toàn vẹn không quá một biến bộ biến thiên trên cùng một quan hệ. Đây là trường hợp thường xẩy rat. Cũng như với các phán đoán đơn quan hệ, cập nhật được tính toán tại vị trí đưa ra yêu cầu. Ép buộc thi hành được thực hiện tại vị trí truy vấn nhờ thuật toán ENFORCE dưới đây:

.

```
Thuật toán: ENFORCE
    Input:  T: kiểu cập nhật; R: quan hệ
    begin
        Truy nhập tất cả các phán đoán biên dịch (R, T, Ci)
        inconsistency ← false
        for  mỗi phán đoán biên dịch  do
        begin
            result ← truy nhập tất cả các bộ mới (tương ứng là bộ cũ) của R với ¬(C)
            if  card(result) ≠ 0  then
            begin
                inconsistency ← true
                exit
            end-if
        end-for
        if not (inconsistency) then
            Gửi các bộ cập nhật đến tất cả các vị trí  đang có các mảnh của R
        else  loại bỏ cập nhật
        end-if
    end.  {ENFORCE}
```

Ví dụ 2.12: Một ví dụ dựa trên phán đoán khoá ngoại : Mã số dự án PNO trong quan hệ ASG là khoá ngoại tương ứng với khoá chính PNO của quan hệ PROJ.Nói cách khác , một dự án được tham chiếu trong quan hệ ASG phải tồn tại trong quan hệ PROJ.

PNO IN ASG REFERENCES PNO IN PROJ

Gọi u là một thao tác chèn một bộ mới vào ASG. Thuật toán trên dùng phán đoán biên dịch (ASG , INSERT ,C). Trong đó C là:

$$\forall NEW \in ASG\ ,\ \exists J \in PROJ:\ NEW.PNO = J.PNO$$

Với phán đoán này, câu lệnh truy xuất tất cả các bộ mới trong ASG^+ không thoả C.

Câu lệnh truy xuất có thể biểu diễn bằng SQL như sau :

```
SELECT    NEW.*
FROM      ASG+  NEW ,  PROJ
WHERE     COUNT(PROJ.PNO  WHERE NEW.PNO = PROJ.PNO) = 0
```

NEW . * là biểu thị tất cả các bộ của ASG^+.

Như vậy , chiến lược là gửi các bộ mới đến các vị trí có lưu quan hệ PROJ để thực hiện nối rồi tập trung tất cả các kết qủa về vị trí vấn tin chính. Vị trí có lưu mảnh của PROJ, kết nối mảnh đó với ASG^+ và gửi kết quả về vị trí vấn tin chính. Tại vị trí chính, hợp tất cả các kết quả lại với nhau..Nếu hợp rỗng nghĩa là CSDL nhất quán, ngược lại cập nhật đã dẫn đến trạng

thái không nhất quán. Loại bỏ chương trình phụ thuộc vào chiến lược được chọn bởi bộ quản lý chương trình của DBMS phân tán.

*Phán đoán có hàm gộp:*Loại phán đoán này có chi phí kiểm tra cao nhất vì chúng đòi hỏi phải tính các hàm gộp. Hàm gộp thường được dùng là MIN, MAX, SUM và COUNT. Mỗi hàm gộp chứa một phần chiếu và một phần chọn. Để ép buộc các phán đóan này một cách hiệu quả, chúng ta có thể tạo ra các phán đoán biên dịch nhằm cô lập dữ liệu thừa được lưu tại mỗi vị trí có chứa quan hệ đi kèm. Dữ liệu này được gọi là các khung nhìn cụ thể.

Vấn đề chủ yếu của kiểm soát toàn vẹn phân tán là chi phí truyền và chi phí xử lý ép buộc thi hành các phán đoán phân tán. Hai vấn đề chính trong thiết kế một tiểu hệ thống toàn vẹn phân tán là định nghĩa các phán đoán phân tán và các thuật toán ép buộc nhằm hạ thấp tối đa chi phí kiểm tra toàn vẹn phân tán. Việc kiểm soát ràng buộc phân tán có thể đạt được bằng cách mở rộng một phương pháp ngăn cản dựa trên quá trình biên dịch các phán đoán toàn vẹn ngữ nghĩa. Đây là phương pháp tổng quát vì tất cả mọi kiểu phán đoán diễn tả bằng logic vị từ bậc nhất đều có thể xử lý được. Nó cũng tương thích với định nghĩa phân mảnh và hạ thấp tối đa việc truyền thông qua lại giữa các vị trí . Có thể có được một hiệu năng ép buộc toàn vẹn phân tán tốt hơn nếu các mảnh được định nghĩa một cách cẩn thận. Vì thế đặc tả ràng buộc toàn vẹn phân tán là một vấn đề quan trọng trong quá trình thiết kế CSDL phân tán.

2.10.3 So sánh việc kiểm soát toàn vẹn ngữ nghĩa tập trung và phân tán

Những khác nhau cơ bản của việc kiểm soát ngữ nghiã tập trung và phân như sau:

Kiểm soát toàn vẹn ngữ nghĩa tập trung	Kiểm soát toàn vẹn ngữ nghĩa phân tán
Vì cơ sở dữ liệu tập trung trên một vị trí, nên không cần xét tới tính tự trị vị trí của cơ sở dữ liệu. Cơ chế ép buộc thi hành việc kiểm soát là thực hiện dựa trên các ràng buộc toàn vẹn • Ràng buộc cấu trúc • Ràng buộc hành vi	Xem xét tính tự trị vị trí , nghĩa là trên mỗi vị trí có thể xử lý vấn tin cục bộ và thực hiện kiểm soát dữ liệu như một hệ CSDLtập trung. Cơ chế cưỡng chế dựa trên các phán đoán: • Phán đoán riêng • Phán đoán hướng tập hợp • Phán đoán có các hàm gộp

CÂU HỎI VÀ BÀI TẬP

1. Lý do phân mảnh dữ liệu
   D. Truy xuất thông tin tập con là hợp lý, .thực hiện nhiều giao dịch đồng thời,
   E. Các thao tác trên các quan hệ, thực hiện song song một câu truy vấn.
   F. Các ứng dụng truy xuất trên khung nhìn
2. Hạn chế khi phân mảnh:
   A. Khả năng xung đột, tăng chi phí truy xuất dữ liệu
   B. Phân tán dữ liệu, không an toàn.
   C. Giảm hiệu suất hoạt động của hệ thống
3. Các quy tắc phân mảnh
   A. Tính đầy đủ, tính phục hồi, tính tách biệt.
   B. Tính toàn vẹn, tính độc lập và tính tách biệt.

C. Các mục dữ liệu được ánh xạ hoàn toàn vào các mảnh và không bị mất.

4. Nguyên tác cấp phát
   A. Chi phí nhỏ nhất, hiệu năng lớn nhất.
   B. Giảm thiểu thời gian đáp ứng và tăng tối đa lưu lượng hệ thống.
   C. Chi phí nhỏ nhất
5. Thông tin phân mảnh
   A. Thông tin về cơ sở dữ liệu, ứng dụng, mạng,
   B. Thông tin về mạng và thông tin về hệ thống máy tính
   C. Thông tin về tổ chức logic cơ sở dữ liệu, vị trí và đặc tính các ứng dụng
6. Thông tin cần thiết phân mảnh ngang
   A. Thông tin về cơ sở dữ liệu và ứng dụng
   B. Thông tin về các phụ thuộc hàm
   C. Thông tin về cơ sở dữ liệu
7. Phân mảnh ngang cơ sở quan hệ R
   A. $R_i = \sigma_{Fi}(R)$, i=1...n
   B. $R_i = \pi(R)$, i=1...n
   C. $R = R_1 \triangleright\triangleleft R_2 \triangleright\triangleleft .....\triangleright\triangleleft .R_n$
8. Phân mảnh ngang dẫn xuất của R:
   A. $R_i = R \triangleright\prec S_i$, $S_i = \sigma_{Fi}(S)$
   B. $R_i = R \triangleright\triangleleft S_i$, $S_i = \sigma_{Fi}(S)$
   C. $R_i = R \times S_i$, $S_i = \sigma_{Fi}(S)$
9. Phân mảnh dọc quan hệ R
   A. Ri , i=1,2..n chứa một tập con các thuộc tính và khoá.
   B. $R_i = \sigma_{Fi}(R)$, i=1...n
   C. $R_i = \pi(R)$, i=1...n
10. Thông tin cần thiết để phân mảnh dọc
   A. Ma trận giá trị sử dụng thuộc tính, ma trận hấp dẫn của thuộc tính.
   B. Ma trận giá trị sử dụng thuộc tính,
   C. Ma trận hấp dẫn của thuộc tính.
11. Thuật toán phân mảnh dọc cho kết quả đúng nếu:
   A. Tính đầy đủ, khôi phục lại, tính tách biệt.
   B. Tính đầy đủ, đơn giản, tính liên kết
   C. Tính toàn vẹn, tính độc lập và tính tách biệt.
12. Bài toán cấp phát phát biểu như sau:
   A. Phân bố các mảnh F trên các node S có các ứng dụng Q sao cho tối ưu
   B. Phân bố các mảnh F trên các node S có các ứng dụng Q
   C. Phân bố các mảnh F trên các node S có các ứng dụng Q sao cho chi phí nhỏ nhất
13. Tính tối ưu cấp phát được định nghĩa::
   A. Chi phí nhỏ nhất:, hiệu năng cao
   B. Thời gian truy xuất thấp nhất.

C. Hiệu năng cao.

14. Thông tin cần thiết cho bài toán cấp phát
    A. Thông tin về CSDL, ứng dụng, vị trí và thông tin về mạng.
    B. Mối liên kết giữa các CSDL, ứng dụng, vị trí và thông tin về mạng.
    C. Thông tin về CSD và thông tin về mạng truyền thông.
15. Hàm tổng chi phí gồm
    A. Chi phí lưu trữ và chi phí xử lý truy vấn
    B. Chi phí xử lý truy vấn và chi phí trao đổi thông tin
    C. Chi phí xử lý tại các node mạng và chi phí trao đổi thông tin
16. Kiểm soát dữ liệu ngữ nghĩa bao gồm:
    A. Quản lý khung nhìn, an toàn, bảo mật dữ liệu và kiểm soát tính toàn vẹn ngữ nghĩa.
    B. Quản lý khung nhìn và kiểm soát tính toàn vẹn ngữ nghĩa.
    C. Kiểm soát tính toàn vẹn ngữ nghĩa.
17. Một khung nhìn dữ liệu là:
    A. Một một quan hệ ảo được định nghĩa bởi một kết quả truy vấn.
    B. Một một quan hệ trong cơ sở dữ liệu.
    C. Một một quan hệ chung của các vị trí khác nhau
18. Người sử dụng:
    A. Chỉ được phép truy nhập CSDL qua khung nhìn.
    B. Xử lý thông tin qua khung nhìn
    C. Truy vấn thông tin qua khung nhìn
19. Quản lý khung nhìn có tác dụng:
    A. Bảo đảm được tính an toàn dữ liệu.
    B. Bảo đảm độ tin cậy của truy vấn dữ liệu.
    C. Bảo đảm tính độc lập của dữ liệu.
20. Khung nhìn cập nhật được:
    A. Khi thực hiện các phép cập nhật trên nó sẽ lan truyền chính xác đến các quan hệ cơ sở.
    B. Có thể thực hiện các phép cập nhật trên nó.
    C. Khi thực hiện các phép cập nhật trên nó che dấu các chi tiết cập nhật.
21. Khung nhìn không cập nhật được:
    A. Khi thực hiện các phép cập nhật các thuộc tính bị che khuất khung nhìn, chúng có thể nhận giá trị không hoặc null.
    B. Khi thực hiện các phép cập nhật chúng được dẫn xuất từ một quan hệ duy nhất bằng phép chọn hoặc phép chiếu.
    C. Được định nghĩa bởi phép chọn hay phép chiếu.
22. Khung nhìn trong các hệ QTCSDL phân tán
    A. Được dẫn xuất từ các quan hệ phân tán.
    B. Được dẫn xuất từ các phép chiếu và chọn
    C. Được dẫn xuất từ một quan hệ duy nhất bằng phép chọn hoặc phép chiếu.
23. An toàn dữ liệu bao gồm các vấn đề
    A. Bảo vệ dữ liệu và các biện pháp kiểm soát cấp/thu hồi quyền

B. Cấp quyền truy xuất cơ sở dữ liệu cho người sử dụng.
C. Các giải pháp kiểm soát cấp quyền phân tán.

24. Các yếu tố ảnh hưởng đến kiểm soát cấp quyền truy xuất CSDL:
A. Người sử dụng, người quản trị cơ sở dữ liệu và các đối tượng cơ sở dữ liệu.
B. Các thao tác kiểm soát người sử dụng, các thao tác trên đối tượng CSDL
C. Người sử dụng và người quản trị cơ sở dữ liệu

25. Một quyền được cấp gồm các thành phần:
A. Người sử dụng, loại thao tác
B. Người sử dụng được quyền thao tác gì trên các đối tượng nào.
C. Người sử dụng được thao tác trên các đối tượng nào.

26. Kiểm soát cấp quyền phân tán bao gồm:
A. Cấp quyền cho người sử dụng ở xa, quản lý các quy tắc cấp quyền, xử lý khung nhìn và nhóm người sử dụng.
B. Cấp quyền cho người sử dụng ở xa , ngăn chặn truy nhập trái phép.
C. Nhận diện người sử dụngvà xác nhận vị trí được truy nhập.

27. Kiểm soát tính toàn vẹn ngữ nghĩa
A. Bảo đảm được tính nhất quán cơ sở dữ liệu
B. Bảo đảm được tính độc lập cơ sở dữ liệu.
C. Bảo đảm an toàn, bảo mật cơ sở dữ liệu.

28. Một trạng thái CSDL được gọi là nhất quán:
A. Nếu nó thỏa một tập các ràng buộc toàn vẹn ngữ nghĩa
B. Nêu no đảm bảo tính nhất quán của CSDL,
C. Nếu nó thỏa một tập các phụ thuộc hàm.

29. Kiểm soát toàn vẹn ngữ nghĩa tập trung gồm:
A. Các loại ràng buộc và cơ chế ép buộc thực thi.
B. Một ngôn ngữ cho phép diễn tả và thao tác các phán đoán toàn vẹn
C. Một cơ chế chịu trách nhiệm thực hiện các hành động cụ thể nhằm ép buộc tính toàn ven khi có cập nhật.

30. Các loại ràng buộc:
A. Ràng buộc tiền định, ràng buộc tiền dịch và ràng buộc tổng quát.
B. Ràng buộc phụ thuộc hàm ràng buộc miền
C. Ràng buộc phụ thuộc hàm, ràng buộc miền khi xoá và di chuyển.

31. Ép buộc thực thi ràng buộc, nghĩa là:
A. Thực hiện việc loại bỏ những chương trình cập nhật vi phạm ràng buộc.
B. Ràng buộc các hành động cập nhật hoặc các phán đoán ràng buộc sai.
C. Thực hiện việc loại bỏ những ràng buộc.hoặc các phán đoán ràng buộc sai.

32. Các phương pháp loại bỏ các trình cập nhật phát sinh mâu thuẫn.
A. Phát hiện mâu thuẫn , phương pháp ngăn chặn mâu thuẫn
B. Phương pháp chuyển CSDL sang một trạng thái nhất quán khác.
C. Phương pháp cập nhật và truy nhập trong CSDL

33. Kiểm soát toàn vẹn ngữ nghĩa phân tán
A. Gồm các thuật toán đảm bảo tính toàn vẹn ngữ nghĩa CSDL phân tán.

B. Gồm các thuật toán xử lý các câu truy vấn cục bộ và thực hiện việc kiểm soát dữ liệu như một hệ quản trị CSDL tập trung.
C. các phương pháp kiểm soát toàn vẹn ngữ nghĩa tập trung.

34. Một tiểu hệ thống kiểm soát toàn vẹn trong một hệ quả trị CSDL phân tán bao gồm:
A. Định nghĩa và lưu trữ các phán đoán và ép buộc thi hành các phán đoán này.
B. Định nghĩa các ràng buộc và ép buộc thi hành các phán đoán
C. Định nghĩa và lưu trữ các phán đoán

35. Phán đoán toàn vẹn bao gồm:
A. Phán đoán riêng, phán đoán hướng tập hợp và phán đoán có các hàm gộp.
B. Phán đoán vị trí lưu các quan hệ trong phán đoán.
C. Phán đoán phải tương thích với dữ liệu của quan hệ tại mỗi vị trí.

36. Ép buộc thi hành các phán đoán toàn vẹn phân tán:
A. Là quyết định xem vị trí nào sẽ thực hiện ép buộc.
B. Là quyết định kiểu cập nhật và vị trí đưa ra yêu cầu cập nhật
C. Là quyết định chi phí truyền thông.

37. Trong cơ sở dữ liệu tập trung:
A. Không cần xét tới tính tự trị vị trí của cơ sở dữ liệu.
B. Cần xét tới tính tự trị vị trí của cơ sở dữ liệu.
C. Tính tự trị vị trí của cơ sở dữ liệu phụ thuộc vào nhu cầu truy vấn dữ liệu.

38. Trong cơ sở dữ liệu phân tán:
A. Không cần xét tới tính tự trị vị trí của cơ sở dữ liệu.
B. Cần xét tới tính tự trị vị trí của cơ sở dữ liệu.
C. Tính tự trị vị trí của cơ sở dữ liệu phụ thuộc vào nhu cầu truy vấn dữ liệu.

39. Cơ chế ép buộc thi hành việc kiểm soát trong CSDL tập trung:
A. Là thực hiện các ràng buộc cấu trúc và ràng buộc hành vi
B. Là thực hiện các ràng buộc cấu trúc
C. Là thực hiện các ràng buộc hành vi

40. Cơ chế ép buộc thi hành việc kiểm soát trong CSDL phán đoán:
A. Là dựa trên các phán đoán riêng và phán đoán hướng tập hợp
B. Là thực hiện các ràng buộc cấu trúc và ràng buộc hành vi
C. Là dựa trên các phán đoán riêng, hướng tập hợp và phán đoán các hàm gộp

CHƯƠNG III: XỬ LÝ TRUY VẤN TRONG CƠ SỞ DỮ LIỆU QUAN HỆ PHÂN TÁN

Nội dung của chương bao gồm các phần sau:

- Khái niệm truy vấn
- Các phép toán đại số quan hệ
- Đặc trưng của xử lý truy vấn
- Phân lớp xử lý truy vấn
- Phân rã truy vấn
- Cục bộ hóa dữ liệu phân tán
- Tối ưu hóa truy vấn phân tán.
- Các thuật toán tối ưu hóa truy vấn phân tán

3.1 GIỚI THIỆU

Chương này sẽ giới thiệu tổng quan về xử lý truy vấn trong các hệ cơ sở dữ liệu quan hệ phân tán. Bộ xử lý truy vấn cung cấp các phương tiện cho người sử dụng có thể xây dựng các câu truy vấn và thực hiện tối ưu hoá truy vấn trong các hệ cơ sở dữ liệu phân tán DBMS. Quá trình tối ưu hoá truy vấn có thể thực hiện khi DBMS tập trung và dữ liệu phân tán. Trong truy vấn phân tán, chi phí truyền thông và thời gian đáp ứng truy vấn là những vấn đề cần quan tâm.

Truy vấn phân tán là một chuỗi các thao tác dữ liệu được thực hiện trên các mảnh quan hệ phân rã. Cụ thể như sau:

- Câu truy vấn phải được phân rã thành một chuỗi các thao tác đại số quan hệ.
- Dữ liệu được truy nhập bởi truy vấn là những mảnh dữ liệu được phân rã, gọi là dữ liệu cục bộ.
- Phép truy vấn đại số trên các mảnh phải được mở rộng với các thao tác truyền thông và tối ưu hoá chức năng tham chiếu các nguồn tài nguyên.

3.2 VẤN ĐỀ XỬ LÝ TRUY VẤN

3.2.1 Đặt vấn đề

Chức năng chính của bộ xử lý truy vấn là chuyển đổi một truy vấn mức cao (phép tính quan hệ) sang một truy vấn mức thấp tương đương (đại số quan hệ). Quá trình chuyển đổi cùng cho một kết quả như nhau.

Có nhiều giải pháp chuyển đổi, mỗi giải pháp khác nhau có thể tiêu thụ tài nguyên của mạng máy tính khác nhau. Vì vậy, cần phải lựa chọn một giải pháp khi thực hiện, nó tiêu thụ tài nguyênt là tối thiểu.

Có 2 phương pháp tối ưu cơ bản: phương pháp biến đổi một câu vấn tin có mức cao thành câu vấn tin tương đương ở mức thấp hơn dưới dạng biểu thức đại số quan hệ và phương pháp chọn lựa trong số các câu vấn tin dạng biểu thức đại số quan hệ tương đương, một biểu thức có chi phí thời gian thực hiện và chi phí sử dụng tài nguyên là ít nhất.

Ví dụ 3.1: Xét 2 quan hệ:

EMP (ENO, ENAME, TITLE) và ASG (ENO, PNO, RESP, DUR).

ENO	ENAME	TITLE
E1	J.Doe	Elect.Eng
E2	M.Smith	Analyst
E2	M.Smith	Analyst
E3	A.Lee	Mech.Eng
E3	A.Lee	Mech.Eng
E4	J.Miller	Programmer
E5	B.Casey	Syst.Anal
E6	L.Chu	Elect.Eng
E7	R.David	Mech.Eng
E8	J.Jones	Syst.Anal

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

Và câu truy vấn : ” Tên của các nhân viên là giám đốc (Manager) dự án”.

Câu truy vấn dưới dạng phép tính quan hệ theo cú pháp SQL là:

```
SELECT    ENAME
FROM      EMP, ASG
WHERE     EMP.ENO = ASG.ENO
          AND RESP = “Manager”.
```

Câu truy vấn trên được biểu diễn dưới dạng biểu thức đại số quan hệ sau

$$\pi_{ENAME}(\sigma_{RESP=\text{"Manager"} \wedge EMP.ENO = ASG.ENO}(EMP \times ASG))$$

Phép toán trên được chuyển đổi tương đương, cho cùng một kết quả nhưng có chi phí thời gian và sử dụng tài nguyên ít hơn:

$$\pi_{ENAME}(EMP \triangleright\!\triangleleft_{ENO} (\sigma_{RESP = \text{“Manager”}}(ASG))) .$$

Trong trường hợp cơ sở dữ liệu tập trung, việc chuyển đổi câu truy vấn sang các phép đại số quan hệ được tiến hành một cách thuận lợi. Chức năng chính của bộ xử lý tập trung là lựa chọn phép truy vấn đại số quan hệ tối ưu trong các phép đại số tương đương.

Trong môi trường phân tán, các phép đại số quan hệ không đủ để mô tả các giải pháp thực hiện. Nó phải được cung cấp thêm các các thao tác để chuyển đổi dữ liệu giữa các vị trí. Việc lựa chọn thứ tự các thao tác đại số quan hệ, bộ xử lý truy vấn cũng phải lực chọn các vị trí tốt nhất để xử lý dữ liệu và đường truyền thông mà dữ liệu sẽ lưu chuyển. Việc này sẽ làm tăng số các giải pháp lựa chọn cần lựa chọn trong số đó một giải pháp thực hiện. Vì vậy việc xử lý truy vấn phân tán càng trở nên khó khăn hơn.

Ví dụ 3.2 Ví dụ này mô tả tầm quan trọng của việc lựa chọn vị trí và việc truyền thông từ truy vấn đại số quan hệ được lựa chọn với CSDL được phân mảnh. Xét câu vấn tin với biểu thức đại số quan hệ ví dụ trên:

$$\pi_{ENAME}(EMP \triangleright\!\triangleleft_{ENO} (\sigma_{RESP = \text{“Manager”}}(ASG)))$$

Giả sử hai quan hệ EMP và ASG được phân mảnh theo chiều ngang như sau:

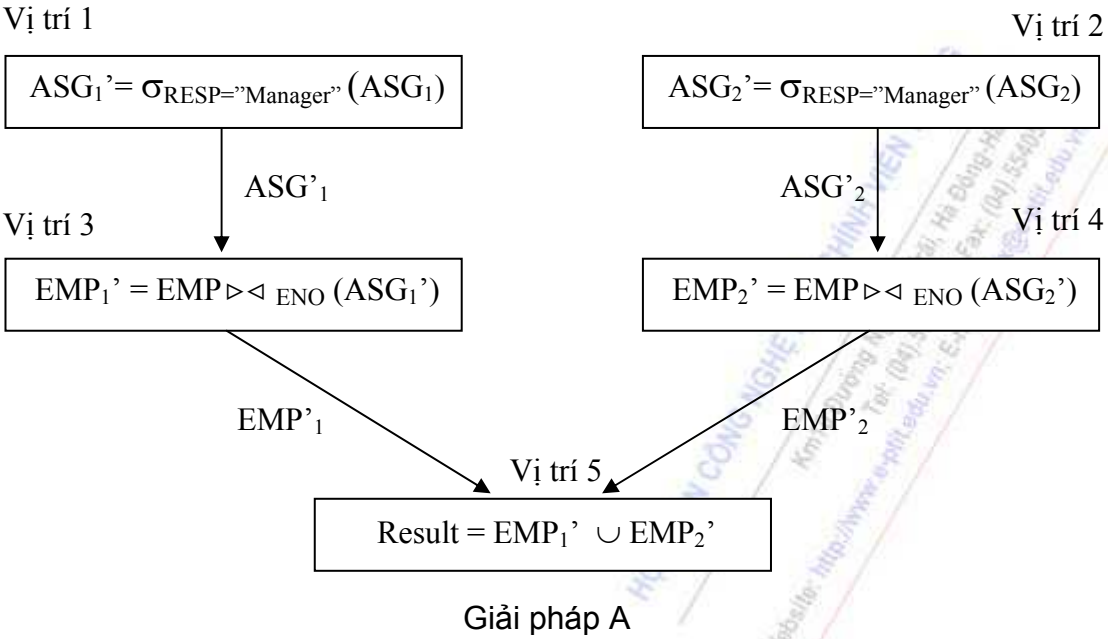
$EMP_1 = \sigma_{ENO \leq \text{“E3”}}(EMP)$.

$EMP_2 = \sigma_{ENO > \text{“E3”}}(EMP)$.

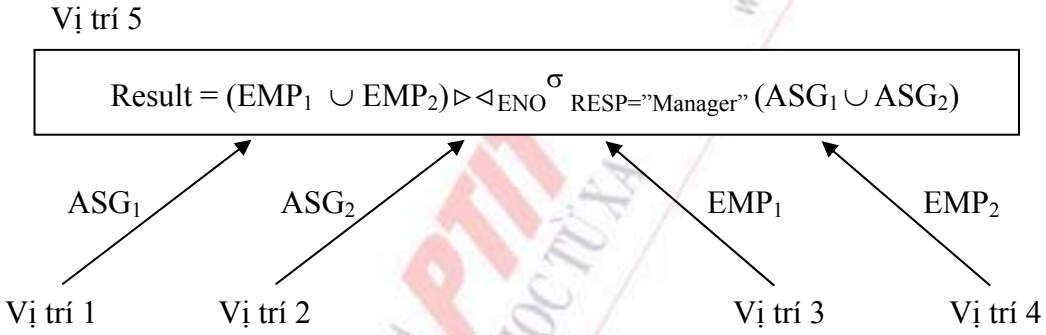
$ASG_1 = \sigma_{ENO \leq \text{“E3”}} (ASG)$.

$ASG_2 = \sigma_{ENO > \text{“E3”}} (ASG)$.

Các mảnh EMP_1, EMP_2, ASG_1, ASG_2 được lưu trữ tại các vị trí 1, 2, 3, 4 và các kết quả được lưu trữ tại vị trí 3.



Giải pháp A



Giải pháp B

Hình 3.1 Các giải pháp truy vấn tương đương

Mũi tên đi kèm ký hiệu quan hệ để chỉ hệ thống sẽ truyền quan hệ kết quả từ vị trí i đến vị trí j. Ví dụ trong giải pháp A, tại vị trí 1 sau khi thực hiện phép chọn trên quan hệ ASG_1 thoả biểu thức RESP="Manager", kết quả ASG'_1 sẽ được truyền từ vị trí 1 sang vị trí 3 để tham gia phép kết nối với quan hệ EMP_1.

Hai giải pháp A và B thực hiện truy vấn là tương đương trong đó:

- Giải pháp A: Sử dụng hai quan hệ EMP và ASG được phân mảnh theo cùng một cách để thực hiện lựa chọn và kết nối các thao tác một cách song song các phép chọn và kết nối.
- Giải pháp B: Tập trung tất cả các dữ liệu trong toán hạng tại vị trí kết quả trước khi xử lý truy vấn.

Để có thể đánh giá chi phí nguồn tài nguyên của hai giải pháp trên, ký hiệu

- Chi phí để thao tác truy xuất một bộ (tuple access) là một đơn vị, ký hiệu tupacc.
- Chi phí để truyền một bộ (tuple transfer) là 10 đơn vị, ký hiệu tuptrans.
- Giả sử lực lượng của quan hệ EMP và ASG sẽ là 400 và 1000 bản ghi.
- Giả sử có 20 Manager trong quan hệ ASG.

Khi đó, tổng chi phí của giải pháp A sẽ là:

1. Tạo ASG' bằng phép chọn ASG cần (10+10)**tupacc* = 20.
2. Truyền ASG'đến các vị trí của EMP cần 10+10)**tuptrans* = 200
3. Tạo EMP' bằng phép kết nối ASG' và EMP cần (10+10)* *tupacc*2* = 40
4. Truyền EMP' đến vị trí kết quả cần (10+10)**tuptrans* = 200

Tổng chi phí *460*

Tổng chi phí của giải pháp B được tính như sau:

1. Truyền EMP đến vị trí 5 cần 400* tuptrans = 4 000
2. Truyền ASG đến vị trí 5 cần 400*tuptrans = 10 000
3. Tạo ASG' bằng phép chọn ASG cần 1000*tupacc = 1 000
4. Kết nối EMP và ASG' cần 400*20*tupacc = 8000

Tổng chi phí *23 000*

Như vậy chi phí để thực hiện giải pháp A thấp hơn chi phí cho giải pháp B.

3.2.2 Mục đích của việc xử lý truy vấn

Mục đích của việc xử lý truy vấn trong môi trường phân tán là biến đổi một truy vấn ở mức cao trên một cơ sở dữ liệu phân tán thành một giải pháp thực hiện hiệu quả được xác định dưới dạng ngôn ngữ mức thấp trên các cơ sở dữ liệu cục bộ. Ngôn ngữ mức cao có thể hiểu là các phép tính quan hệ, các ngôn ngữ mức thấp là sự mở rộng của đại số quan hệ và các thao tác truyền dữ liệu giữa các vị trí dữ liệu.

Tối ưu hoá truy vấn là một vấn đề quan trọng trong việc xử lý truy vấn. Có nhiều phép biến đổi một truy vấn mức cao trên cơ sở dữ liệu phân tán thành nhiều giải pháp thực hiện dưới dạng ngôn ngữ mức thấp, nhưng trong đó chỉ có một giải pháp thực hiện có hiệu quả, tối ưu về chi phí sử dụng tài nguyên của mạng bao gồm chi phí sử dụng bộ nhớ, thời gian xử lý và thời gian truyền dữ liệu.

Chỉ số đánh giá chi phi sử dụng tài nguyên mạng hoặc là tổng thời gian xử lý các thao tác truy vấn tại các vị trí khác nhau và việc truyền dữ liệu giữa các vị trí [Sacco and Yao,1982]. Hoặc đánh giá theo chỉ số theo thời gian đáp ứng của truy vấn [Epstein et al, 1978] là tổng thời gian thực hiện truy vấn. Các thao tác có thể được thực hiện đồng thời song song tại các vị trí khác nhau, vì vậy thời gian đáp ứng có thể nhỏ hơn tổng chi phí.

Trong môi trường cơ sở dữ liệu phân tán, chỉ số đánh giá tối ưu hoá truy vấn có thể dựa vào tổng chi phí giảm thiểu sử dụng bộ nhớ, chi phí thời gian cần thiết khi thực hiện các thao tác vào/ra dữ liệu trong bộ nhớ và chi phí cần thiết để trao đổi dữ liệu giữa các bên tham gia vào trong quá trình xử lý truy vấn. Chi phí truyền thông là một trong các nhân tố quan trọng, được quan tâm trong cơ sở dữ liệu phân tán.

3.2.3 Độ phức tạp của các thao tác đại số quan hệ

Độ phức tạp của các phép toán đại số quan hệ có ảnh hưởng trực tiếp đến thời gian thực hiện các thao tác truy vấn. Độ phức tạp của các phép toán phụ thuộc vào lực lượng của quan

hệ (Cardinality), phương pháp phân mảnh, cấp phát dữ liệu cũng như các vấn đề về cấu trúc lưu trữ các mảnh. Độ phức tạp của các thao tác đơn ngôi và hai ngôi theo thứ tự tăng dần, vì thế cũng tăng dần thời gian thực hiện. Độ phức tạp O(n) là tuyến tính đối với các phép toán đơn ngôi, trong đó n là lực lượng của quan hệ, với điều kiện các bộ độc lập với nhau. Độ phức tạp O(n*logn) đối với các phép toán hai ngôi nếu các bộ tồn tại phép so sánh dựa trên giá trị của một thuộc tính nào đó. Độ phức tạp $O(n^2)$ đối với các phép nhân, tích Đề các của hai quan hệ, vì mỗi bộ trong quan hệ này phải tổ hợp với tất cả các bộ trong quan hệ kia.. Độ phức tạp của các thao tác đại số quan hệ có thể được xác định như sau:

Phép toán	Độ phức tạp
Phép chọn (Select) Project (Không loại bỏ trùng lặp)	O(n)
Project (có loại bỏ trùng lặp) Các phép gộp nhóm	O(n*log n)
Phép nối (Join) Phép nửa nối (Semijoin) Phép chia (Division) Các phép tập hợp (Setoperators)	O(n*log n)
Tích Đề các (Cartesian Product)	O(n2)

Hình 3.2 Độ phức tạp của các phép toán quan hệ

3.3 ĐẶC TRƯNG CỦA BỘ XỬ LÝ TRUY VẤN

Khó có thể đánh giá và so sánh độ phức tạp xử lý truy vấn trong cả hai trường hợp cơ sở dữ liệu tập trung [Jarke and Koch, 1984] và phân tán [Sacco and Yao, 1982], [Aper et al, 1983] bởi vì chúng có nhiều điểm khác nhau. Sau đây là một số đặc trưng của bộ xử lý truy vấn. Bốn đặc điểm đầu tiên chung cho cả trường hợp tập trung và trường hợp phân tán. Bốn đặc điểm tiếp theo là cho bộ xử lý truy vấn phân tán.

3.3.1 Ngôn ngữ (Languages)

Nói chung, các thao tác xử lý truy vấn được thực hiện trong hệ quản trị cơ sở dữ liệu quan hệ (DBMS) bằng ngôn ngữ bậc cao, hệ thống có khả năng thực hiện tối ưu hoá câu hỏi truy vấn. Ngôn ngữ xử lý truy vấn dựa trên phép tính đại số quan hệ hay đại số quan hệ, tức là chuyển đổi câu truy vấn dưới dạng phép tính quan hệ thành đại số quan hệ. Trong môi trường phân tán, ngôn ngữ đầu ra là tổ hợp của các phép đại số quan hệ cơ bản được mở rộng thêm các nguyên thuỷ truyền thông. Các phép toán của ngôn ngữ này được cài đặt trực tiếp trong hệ thống. Quá trình xử lý truy vấn phải thực hiện việc ánh xạ một cách hiệu quả từ ngôn ngữ đầu vào đến ngôn ngữ đầu ra.

3.3.2 Các kiểu tối ưu hoá (Types of Optimization)

Tối ưu hoá truy vấn là nhằm mục đích lựa chọn một giải pháp tốt nhất trong toàn bộ các giải pháp có thể thực hiện. Một phương pháp tối ưu hoá truy vấn có hiệu quả là tìm kiếm trong tập các giải pháp và dự đoán chi phí của mỗi giải pháp, lựa chọn giải pháp có chi phí là nhỏ nhất. Tuy nhiên phương pháp này phải chi phí cho chính quá trình tối ưu hoá. Không gian sử dụng thực hiện giải pháp tăng lên khi số các mảnh quan hệ hay số quan hệ tăng lên. Vì vậy phương pháp tìm kiếm vét cạn được sử dụng "Exhaustive" cho với tất cả các giải pháp có thể thực hiện được.

Để tránh những tìm kiếm vét cạn có chi phí cao, có các giải pháp ngẫu nhiên như Iterative Improvement [Swami, 1989] và Simulated Annealing [Ioannidis and Wong, 1987] được đề xuất. Cố gắng tìm kiếm một giải pháp tốt, không nhất thiết là tốt nhất nhưng tránh được chi phí tối ưu hoá quá cao về mặt sử dụng bộ nhớ và thời gian thực hiện.

Một giải pháp làm giảm chi phí tìm kiếm vét cạn là sử dụng các giải thuật Heuristics thu hẹp phạm vi tìm kiếm, chỉ có một số giải pháp được xem xét. Trong môi trường tập trung và phân tán, một giải thuật Heuristics làm giảm kích thước các quan hệ trung gian bằng cách thực hiện các phép toán đơn ngôi trước và sắp xếp thứ tự thực hiến các phép toán hai ngôi theo kích thước tăng dần của các quan hệ trung gian do chúng tạo ra. Một cách đánh giá quan trọng trong các hệ phân tán là thay thế phép kết nối bằng các tổ hợp các nối nửa nhằm giảm thiểu chi phí truyền thông dữ liệu.

3.3.3 Thời điểm tối ưu hoá (Optimization timing)

Một câu truy vấn có thể được tối ưu hoá tại các thời điểm khác nhau liên quan đến thời gian thực hiện truy vấn. Việc tối ưu hoá có thể được thực hiện theo kiểu tĩnh (Statically) trước khi thực hiện truy vấn hoặc theo kiểu động (Dynamically) khi truy vấn được thực hiện. Việc tối ưu hoá truy vấn tĩnh được thực hiện tại thời điểm biên dịch truy vấn. Vì vậy, chi phí của truy vấn có thể được giảm dần qua nhiều lần thực hiện truy vấn, vì đây là các thời điểm thích hợp cho phương pháp tìm kiếm vét cạn (Exhaustive). Kích thước của các quan hệ trung gian không được biết trước, trước khi thực hiện. Vì vậy cần phải được đánh giá ước lượng theo số liệu thống kê cơ sở dữ liệu. Các sai sót trong cách đánh giá này là có thể dẫn đến việc lựa chọn một giải pháp gần tối ưu.

Việc tối ưu hoá động được tiến hành vào thời gian thực hiện truy vấn. Trong các thời điểm thực hiện tối ưu hoá truy vấn, việc lựa chọn thao tác tiếp theo tốt nhất cho tối ưu hoá truy vấn dựa trên những thông tin chính xác về kết quả của các thao tác thực hiện trước đó. Vì vậy, các số liệu thống kê không cần thiết cho việc đánh giá kích thước của các quan hệ trung gian (nhưng có ích trong việc lựa chọn các thao tác đầu tiên).

Ưu điểm của phương pháp động so với phương pháp tĩnh là kích thước thực sự của các quan hệ trung gian là phù hợp cho bộ xử lý truy vấn. Vì vậy sẽ giảm thiểu xác xuất cho việc lựa chọn một giải pháp tồi. Nhược điểm của phương pháp động là các thao tác tối ưu hoá có chi phí cao. Lặp lại nhiều lần cho mỗi thao tác. Vì vậy phương pháp này sẽ rất phù hợp cho một số câu truy vấn đặc biệt.

Các phương pháp tối ưu hoá truy vấn hỗn hợp có các ưu điểm của tối ưu hoá truy vấn tĩnh, tránh được các vấn đề được tạo ra bởi các đánh giá không chính xác gây ra. Về cơ bản

phương pháp này là tĩnh nhưng quá trình truy vấn động có thể ra lúc chạy khi phát hiện có sự khác biệt lớn giữa kích thước dự đoán và kích thước thực tế của các quan hệ trung gian.

3.3.4 Số liệu thống kê (Statistics)

Tính hiệu quả của việc tối ưu hoá truy vấn là dựa trên các số liệu thống kê về cơ sở dữ liệu. Tối ưu hoá truy vấn động cần đến các số liệu thống kê nhằm chọn các thao tác cần phải thực hiện trước tiên. Tối ưu hoá truy vấn tĩnh có nhiều yêu cầu hơn, vì kích thước của các quan hệ trung gian phải được đánh giá dựa trên các số liệu thống kê về cơ sở dữ liệu.

Trong cơ sở dữ liệu phân tán, các số liệu thống kê dành cho việc tối ưu hoá truy vấn có liên quan đến các mảnh, lực lượng và kích thước của mảnh, cũng như kích thước và số lượng các giá trị phân biệt của mỗi thuộc tính. Để giảm thiểu xác xuất lỗi, cần nhiều thông tin chi tiết hơn như các biểu đồ hình cột cho các giá trị thuộc tính... Độ chính xác của số liệu thống kê phụ thuộc vào việc cập nhật theo chu kỳ. Với phương pháp tối ưu hoá tĩnh, các thay đổi lớn về số liệu thống kê được sử dụng để tối ưu hoá truy vấn có thể buộc phải tạo lại quá trình tối ưu hoá truy vấn.

3.3.5 Vị trí quyết định (Decision sites)

Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp tĩnh, có thể sử dụng một vị trí hay nhiều vị trí tham gia vào việc chọn một giải pháp trả lời câu truy vấn. Hầu hết các hệ thống sử dụng phương pháp quyết định tập trung, trong đó có một vị trí đưa ra giải pháp. Quá trình quyết định có thể được phân tán cho nhiều vị trí tham gia vào việc tạo một giải pháp tốt nhất. Phương pháp tập trung đơn giản, nhưng đòi hỏi phải biết toàn bộ về các thông tin cục bộ. Các phương pháp lai, hỗn hợp thì có một vị trí đưa ra quyết định chính và các vị trí khác đưa ra các quyết định cục bộ. Chẳng hạn như System R* sử dụng cách tiếp cận hỗn hợp.

3.3.6 Khai thác cấu hình mạng (Exploitation of Network topology)

Trong các mạng diện rộng WAN, hàm chi phí nhỏ nhất có thể bị giới hạn bởi chi phí truyền thông. Giả thiết này làm đơn giản hoá tối ưu hoá truy vấn phân tán. Vì vậy có thể tách vấn đề này thành 2 vấn đề riêng biệt: lựa chọn giải pháp thực hiện toàn cục dựa trên truyền thông giữa các vị trí và lựa chọn mỗi giải pháp thực hiện cục bộ dựa trên thuật toán xử lý truy vấn tập trung.

Trong các mạng cục bộ LAN, chi phí truyền thông có thể so sánh với chi phí vào/ra. Vì vậy bộ xử lý truy vấn phân tán tăng quá trình thực hiện song song (tất nhiên chi phí truyền thông sẽ nhiều hơn). Khả năng quảng bá của các mạng cục bộ được sử dụng có hiệu quả tối ưu hoá việc xử lý các phép kết nối. Trong môi trường Client/Server, máy khách có thể được khai thác để thực hiện thao tác chuyển đổi cơ sở dữ liệu. Tối ưu hoá truy vấn để ra quyết định một phần của truy vấn sẽ được thực hiện trên máy khách.

3.3.7 Khai thác các mảnh nhân bản (Exploitation of Replicated Fragments)

Một quan hệ phân tán thường được chia thành các mảnh quan hệ. Các câu truy vấn phân tán được mô tả trên các quan hệ toàn cục sẽ được ánh xạ thành các câu truy vấn trên các mảnh vật lý của quan hệ bằng việc chuyển đổi các quan hệ thành các mảnh. Quá trình này được gọi là quá trình cục bộ hoá. Thường người ta nhân bản các mảnh ở nhiều vị trí khác nhau. Phần lớn các thuật toán tối ưu hoá đều xem xét quá trình cục bộ hoá một cách độc lập với quá trình

tối ưu hoá. Một số thuật toán khác tận dụng sự tồn tại của việc nhân bản trong lúc chạy để giảm số lần truyền thông, thuật toán tối ưu hoá sẽ phức tạp hơn.

3.3.8 Sử dụng nửa kết nối (Use of Semijoint)

Thao tác nửa kết nối làm giảm kích thước của các quan hệ. Khi chi phí truyền thông được quan tâm, thì thao tác nửa kết nối đặc biệt sẽ làm giảm dữ liệu cần trao đổi giữa các vị trí. Tuy nhiên, việc sử dụng thao tác nửa kết nối có thể dẫn đến việc tăng số lượng các thông điệp và thời gian xử lý cục bộ. Các thao tác nửa kết nối hiệu quả trong các mạng tốc độ cao, cho phép giảm các thao tác kết nối. Một số thuật toán tối ưu hoá truy vấn tổ hợp tối ưu kết nối với nửa kết nối.

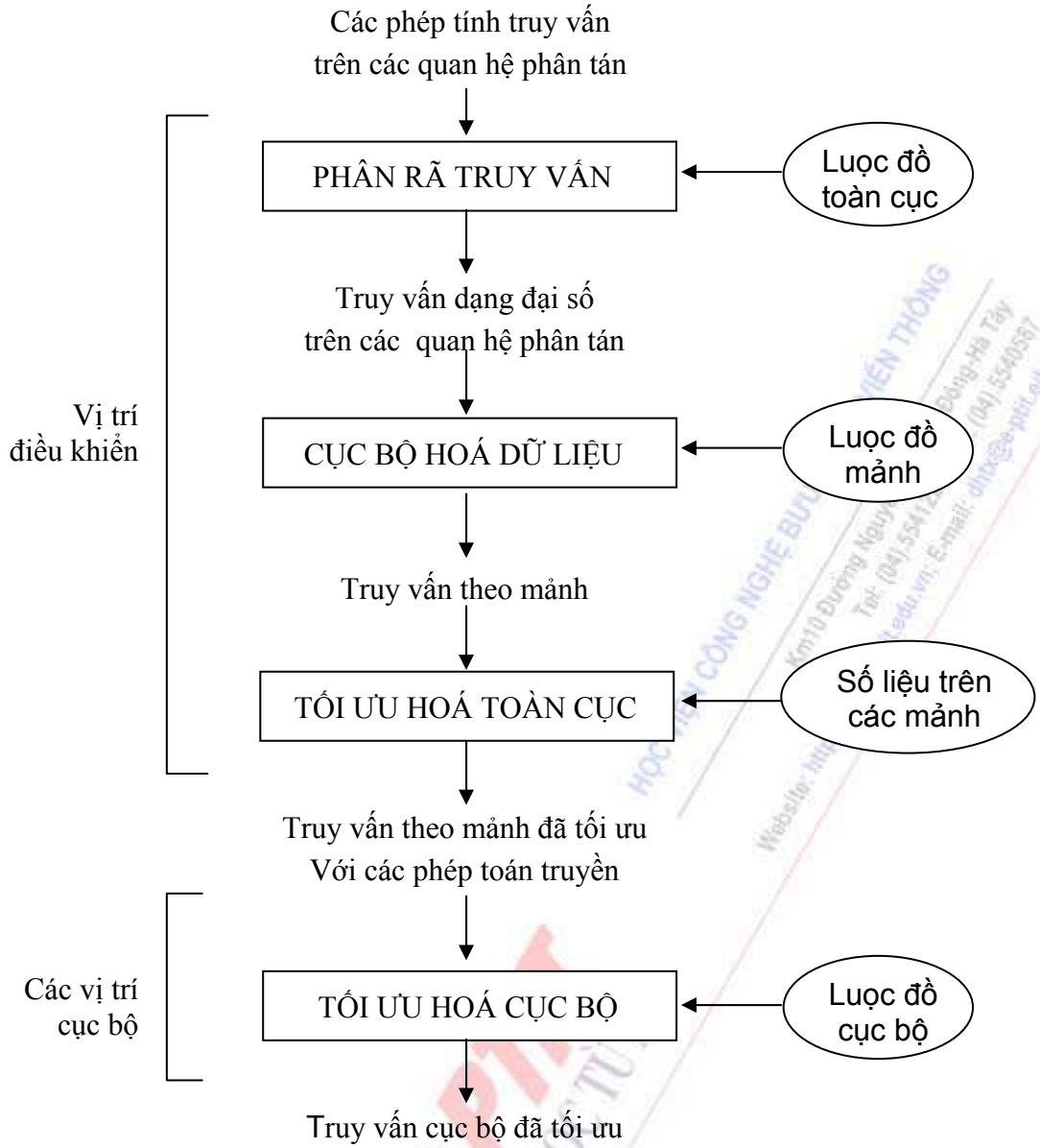
3.4 CÁC TẦNG CỦA QUÁ TRÌNH XỬ LÝ TRUY VẤN

Câu truy vấn phải được phân rã thành một chuỗi các phép toán quan hệ được gọi là truy vấn đại số. Dữ liệu cần truy vấn phải được cục bộ hoá để các thao tác trên các quan hệ được chuyển thành các thao tác trên dữ liệu cục bộ (các mảnh). Cuối cùng câu truy vấn đại số trên các mảnh phải được mở rộng bao gồm các thao tác trao đổi thông tin giữa các vị trí và được tối ưu hoá câu truy vấn. Như vậy có thể mô tả việc xử lý truy vấn trong nhiều vấn đề nhỏ tương ứng với các tầng khác nhau. Lược đồ phân tầng chung của quá trình xử lý truy vấn như hình 3.3. Đầu vào là một câu truy vấn trên dữ liệu phân tán được xác định dưới dạng các phép tính quan hệ. Câu truy vấn phân tán được đặt trên các quan hệ toàn cục, nghĩa là dữ liệu phân tán được che dấu. Bốn tầng lược đồ ánh xạ truy vấn phân tán thành một chuỗi các thao tác cục bộ được tối ưu hoá, hoạt động trên cơ sở dữ liệu cục bộ.

Chức năng các tầng bao gồm: phân rã truy vấn, tập trung hoá dữ liệu, tối ưu hoá truy vấn toàn cục và tối ưu hoá truy vấn cục bộ.

Phân rã truy vấn và tập trung hoá dữ liệu tương ứng với việc viết lại truy vấn. Chức năng của ba tầng đầu tiên được thực hiện tại một vị trí tập trung và sử dụng các thông tin toàn cục còn chức năng của tầng thứ tư được thực hiện ở vị trí cục bộ.

Phân rã truy vấn là giai đoạn đầu tiên của quá trình xử lý câu truy vấn, thực hiện việc biến đổi câu truy vấn ở dạng ngôn ngữ bậc cao thành câu truy vấn ngôn ngữ bậc thấp thực thi cho kết quả tương đương. đặc trưng của giai đoạn này, khi biến đổi không sử dụng các thông tin về dữ liệu đã được phân tán trên các vị trí.



Hình 3.3 Lược đồ phân tầng tổng quát để xử lý truy vấn phân tán

3.5 PHÂN RÃ TRUY VẤN

Một lược đồ tổng quát để xử lý truy vấn dữ liệu phân tán gồm 4 tầng, trong đó 2 tầng đầu có chức năng phân rã câu truy vấn (Query Decomposition) và cục bộ hoá dữ liệu (Data Location). Các chức năng này được thực hiện liên tiếp, nhằm biến đổi câu truy vấn dạng phép tính quan hệ được đặc tả trên các quan hệ toàn cục thành câu truy vấn dưới dạng đại số quan hệ được định nghĩa trên các mảnh.

Tầng phân rã câu truy vấn có chức năng ánh xạ câu truy vấn phân tán ở dạng phép tính quan hệ thành câu truy vấn đại số trên quan hệ toàn cục. Thông tin cần thiết cho việc biến đổi phân rã truy vấn phân tán được tìm thấy trong mô tả lược đồ khái niệm toàn cục và trong mô tả các quan hệ toàn cục. Thông tin về phân tán các quan hệ không được sử dụngểtong tầng này, nhưng sẽ được sử dụng trong các tầng kế tiếp. Vì vậy các kỹ thuật phân rã được áp dụng trong tầng này là những kỹ thuật của các hệ quản trị cơ sở dữ liệu quan hệ tập trung.

Phân rã câu truy vấn có thể thực hiện theo 4 bước liên tiếp nhau:

- *Bước chuẩn hoá*: Các câu truy vấn bằng các phép tính quan hệ được viết lại dưới dạng chuẩn tắc thích hợp cho những bước tiếp theo. Sự chuẩn hoá một câu truy vấn bao gồm đặt các lượng tử và lượng tử hoá truy vấn bằng cách áp dụng độ ưu tiên các toán tử logic.
- *Bước phân tích:* Câu truy vấn đã chuẩn hoá được phân tích về mặt ngữ nghĩa nhằm loại bỏ các câu vấn tin sai càng sớm càng tốt. Tìm ra truy vấn sai chỉ tồn tại với một tập con các phép tính quan hệ. Thông thường sử dụng một loại đồ thị để nắm bắt ngữ nghĩa của câu truy vấn.
- *Bước loại bỏ dư thừa:* Câu truy vấn đúng được đơn giản hoá bằng cách loại bỏ các phụ thuộc dư thừa. Truy vấn dư thừa chỉ xuất hiện khi các một truy vấn là kết quả của việc biến đổi hệ thống được áp dụng cho truy vấn của người sử dụng.
- *Bước xây dựng lại câu truy vấn:* Câu truy vấn phép tính quan hệ được xây dựng lại dưới dạng truy vấn đại số quan hệ bằng các quy tắc biến đổi.

3.3.1 Bước chuẩn hoá câu truy vấn

Mức độ phức tạp của câu truy vấn đầu vào phụ thuộc vào ngôn ngữ cung cấp các phương tiện. Mục đích của việc chuẩn hoá là biến đổi một câu truy vấn sang một dạng chuẩn để xử lý tiếp. Với các ngôn ngữ quan hệ như SQL, việc chuyển đổi quan trọng nhất là lượng từ hoá truy vấn (Query qualification) (mệnh đề WHERE. Có 2 dạng chuẩn có thể cho việc dự đoán: một là đưa ra mức độ ưu tiên cho phép AND và cái còn lại cho phép OR.

- *Dạng chuẩn hội (Conjunctive Normal Form)* là hội của các tuyển như sau:

  $(p_{11} \vee p_{12} \vee \ldots \vee p_{1n}) \wedge \ldots \wedge (p_{m1} \vee p_{m2} \vee \ldots \vee p_{mn})$

  Trong đó, p_{ij} là một vị từ đơn giản.
- *Ngược lại một lượng từ hoá ở dạng tuyển (Disjunctive Normal Form)*

  $(p_{11} \wedge p_{12} \wedge \ldots \wedge p_{1n} \vee \ldots \vee (p_{m1} \wedge p_{m2} \wedge \ldots \wedge p_{mn})$

Biến đổi các vị từ phi lượng từ bằng cách sử dụng các quy tắc tương đương như sau:

1. $p_1 \wedge p_2 \Leftrightarrow p_2 \wedge p_1$.
2. $p_1 \vee p_2 \Leftrightarrow p_2 \vee p_1$
3. $p_1 \wedge (p_2 \wedge p_3) \Leftrightarrow (p_1 \wedge p_2) \wedge p_3$
4. $p_1 \vee (p_2 \vee p_3) \Leftrightarrow (p_1 \vee p_2) \vee p_3$
5. $p_1 \wedge (p_2 \vee p_3) \Leftrightarrow (p_1 \wedge p_2) \vee (p_1 \wedge p_3)$
6. $p_1 \vee (p_2 \wedge p_3) \Leftrightarrow (p_1 \vee p_2) \wedge (p_1 \vee p_3)$
7. $\neg (p_1 \wedge p_2) \Leftrightarrow \neg p_1 \vee (\neg p_2)$
8. $\neg (p_1 \vee p_2) \Leftrightarrow \neg p_1 \wedge (\neg p_2)$
9. $\neg (\neg p) \Leftrightarrow p$

Ví dụ 3.3: "Tìm tên các nhân viên làm việc trong dự án P1 trong 12 hoặc 24 tháng".

```
SELECT   ENAME
FROM     EMP, ASG
WHERE    EMP.ENO = ASG.ENO
         AND  ASG.PNO = "P1"
         AND   DUR = 12  OR  DUR  = 24
```

Lượng từ hoá dạng chuẩn hội là:

EMP.ENO = ASG.ENO ∧ ASG.PNO = "P_1" ∧ (DUR = 12 ∨ DUR = 24)

Lượng từ hoá dạng chuẩn tuyển là:

(EMP.ENO = ASG.ENO ∧ ASG.PNO = "P_1" ∧ DUR = 12)

∨ (EMP.ENO = ASG.ENO ∧ ASG.PNO = "P_1" ∧ DUR = 24)

3.3.2 Bước phân tích

Bước phân tích câu truy vấn cho phép loại bỏ câu truy vấn đã được chuẩn hoá nhưng sai kiểu hoặc không đúng ngữ nghĩa.

a) Sai kiểu: Một câu truy vấn sai kiểu (Semantically Incorrect) nếu có một thuộc tính trong quan hệ hay tên một quan hệ̂cha được khai báo trong lược đồ toàn cục, hoặc các thao tác áp dụng cho các thuộc tính có kiểu không thích hợp. Tìm các truy vấn có kiểu không đúng tương tự như kiểm tra khai báo kiểu trong các ngôn ngữ lập trình.

Ví dụ 3.4:

```
SELECT   E#
FROM     EMP
WHERE    ENAME > 200.
```

Truy vấn này sai kiểu, vì

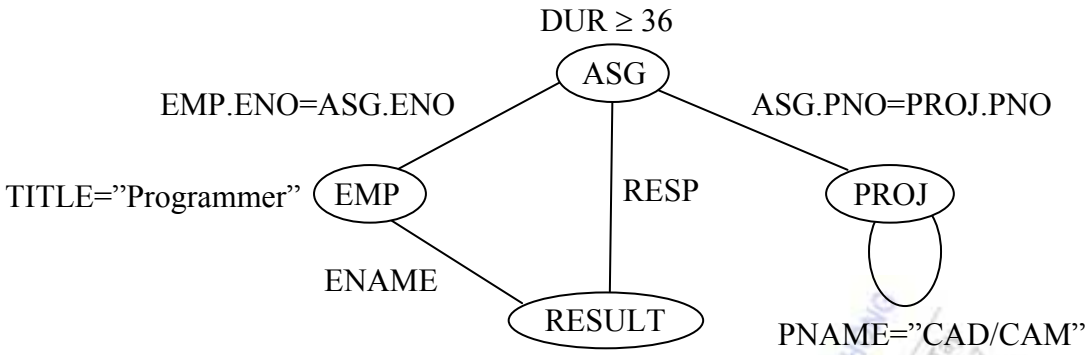
- Thuộc tính E# không phải là thuộc tính của lược đồ quan hệ toàn cục.
- ENAME có kiểu chuỗi (String) không thể so sánh với 200 kiểu số.

b. Sai ngữ nghĩa: Một câu truy vấn được gọi là sai nghĩa (Semantically Incorrect) nếu các thành phần của nó không tham gia vào việc tạo ra kết quả. Để có thể xác định tính đúng đắn ngữ nghĩa câu truy vấn tổng quát, bằng cách biểu diễn đồ thị truy vấn (Query Graph) hay đồ thị kết nối [Ulhman, 1982] gồm các phép chọn, phép chiếu và các phép kết nối, không chứa các phép tuyển và phép phủ định. Trong một đồ thị truy vấn, có một node biểu thị cho một quan hệ kết quả và các node khác biểu thị cho các quan hệ toán hạng. Đường nối giữa 2 node không phải là quan hệ kết quả biểu thị cho phép kết nối. đường nối có node đích là quan hệ kết quả biểu thị cho phép chiếu. Các node quan hệ toán hạng có thể được gán nhãn là một vị từ chọn hoặc vị từ tự kết nối. Đồ thị kết nối nối (Join Graph) là đồ thị con của đồ thị truy vấn, được sử dụng nhiều trong kỹ thuật tối ưu hoá truy vấn.

Ví dụ 3.5: "Tên và nhiệm vụ của các lập trình viên đã làm việc cho dự án CAD/CAM hơn 3 năm"

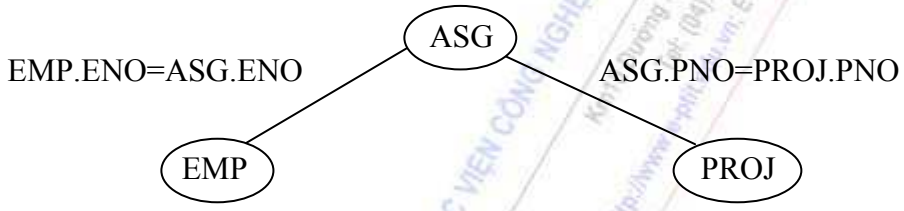
```
SELECT    ENAME, RESP
FROM      EMP, ASG, PROJ
WHERE     EMP.ENO = ASG.ENO
          AND   ASG.PNO = PROJ.PNO
          AND   PNAME = "CAD/CAM"
          AND   DUR ≥ 36
          AND   TITLE = "Programmer"
```

Đồ thị truy vấn như sau:



Hình 3.4: Ví dụ đồ thị truy vấn

Đồ thị kết nối tương ứng:



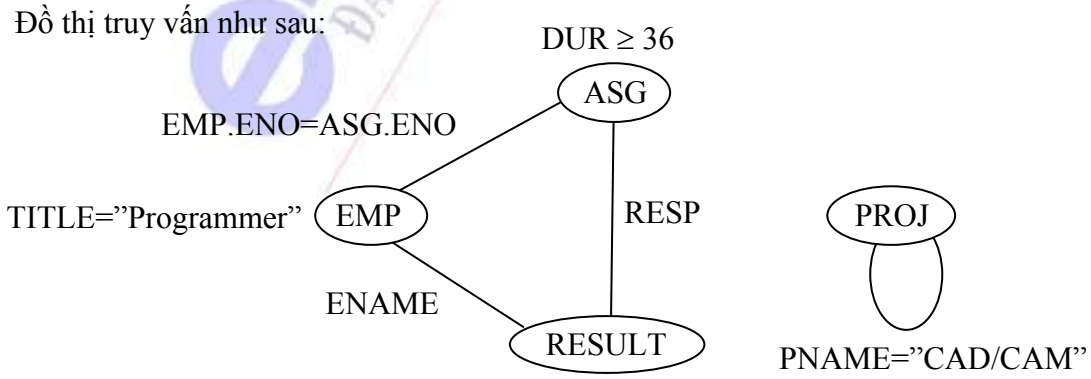
Hình 3.5: Đồ thị kết nối tương ứng

Đồ thị truy vấn sử dụng nhằm xác định tính đúng đắn về mặt ngữ nghĩa của truy vấn hội đa biến không có phủ định (¬). Một câu truy vấn không đúng về mặt ngữ nghĩa nếu đồ thị truy vấn của nó không liên thông. Nghĩa là có ít nhất một đồ thị con, tương ứng câu với truy vấn con, tách ra khỏi đồ thị truy vấn có chứa quan hệ kết quả. Thường khi có các vị từ kết nối bị thiếu, câu truy vấn đó cần được loại bỏ.

Ví dụ 3.6: Xét câu truy vấn sau:

```
SELECT    ENAME, RESP
FROM      EMP, ASG, PROJ
WHERE     EMP.ENO = ASG.ENO
          AND  PNAME = “CAD/CAM”
          AND  DUR ≥ 36
          AND  TITLE = “PROGRAMMER”
```

Đồ thị truy vấn như sau:



Hình 3.6: Đồ thị truy vấn không liên thông

Đồ thị truy vấn hình 3.6 không liên thông, có 2 đồ thị tách biệt nhau. Điều này có nghĩa là câu truy vấn này sai nghĩa. Có 3 giải pháp khắc phục (1) loại bỏ câu vấn tin, (2) giả thiết một tích Đề các giữa quan hệ ASG và PROJ, hoặc (3) có thể đoán nhận vị từ kết nối bị thiếu ASG.PNO = PROJ.PNO.

3.3.3 Bước loại bỏ dư thừa

Trong một câu truy vấn, có thể loại bỏ các vị từ dư thừa và các thao tác dư thừa bằng cách giản ước các lượng từ hoá bằng các quy tắc luỹ đẳng sau đây:

1. $p \wedge p \Leftrightarrow p$
2. $p \vee p \Leftrightarrow p$
3. $p \wedge true \Leftrightarrow p$
4. $p \vee false \Leftrightarrow p$
5. $p \wedge false \Leftrightarrow false$
6. $p \vee true \Leftrightarrow true$
7. $p \wedge \neg p \Leftrightarrow false$
8. $p \vee \neg p \Leftrightarrow true$
9. $p_1 \wedge (p_1 \vee p_2) \Leftrightarrow p_1$
10. $p_1 \vee (p_1 \wedge p_2) \Leftrightarrow p_1$

Ví dụ 3.7: Xét câu truy vấn sau:

```
SELECT      TITLE
FROM        EMP
WHERE       (NOT  (TITLE = “Programmer”)
            AND   (TITLE = “Programmer”
            OR      TITLE = “Elect.Eng” )
            AND   NOT  (TITLE = “Elect.Eng” ))
            OR      ENAME = :J. Doe”
```

Đặt p_1 là TITLE = “Programmer”
P_2 là TITLE = “Elect.Eng”
P_3 là TENAME = :J. Doe”

Lượng từ hoá câu truy vấn trên được là:

Áp dụng quy tắc: $p \wedge (s \vee q) \Leftrightarrow (p \wedge s) \vee (p \wedge q)$ và $p \wedge (s \wedge q) \Leftrightarrow (p \wedge s) \wedge q$

khi đó: $(\neg p_1 \wedge (p_1 \vee p_2) \wedge \neg p_2) \vee p_3 \Leftrightarrow ((\neg p_1 \wedge p_1) \vee (\neg p_1 \wedge p_2) \wedge \neg p_2) \vee p_3$

$\Leftrightarrow ((\neg p_1 \wedge (p_2 \wedge \neg p_2)) \vee p_3 \Leftrightarrow \neg p_1 \wedge false \vee p_3 \Leftrightarrow p_3$

Tức là:

```
SELECT      TITLE
FROM        EMP
WHERE       ENAME = :J. Doe”
```

3.3.3 Bước viết lại truy vấn

Bước cuối cùng của việc phân rã truy vấn là viết lại truy vấn dưới dạng đại số quan hệ. Bước này lại chia thành các bước nhỏ sau:

1. Chuyển đổi câu truy vấn từ phép tính quan hệ sang đại số quan hệ.

2. Xây dựng lại truy vấn đại số quan hệ để cải thiện khả năng thực hiện.

Có thể biểu diễn câu truy vấn đại số quan hệ bằng một cây đại số quan hệ. Các node lá của nó biểu thị cho một quan hệ được lưu trữ trong cơ sở dữ liệu và các node không phải là node lá biểu thị cho một quan hệ trung gian sinh ra bởi các phép toán đại số quan hệ. Chuỗi các thao tác từ các node lá đến các node gốc biểu diễn cho kết quả câu truy vấn.

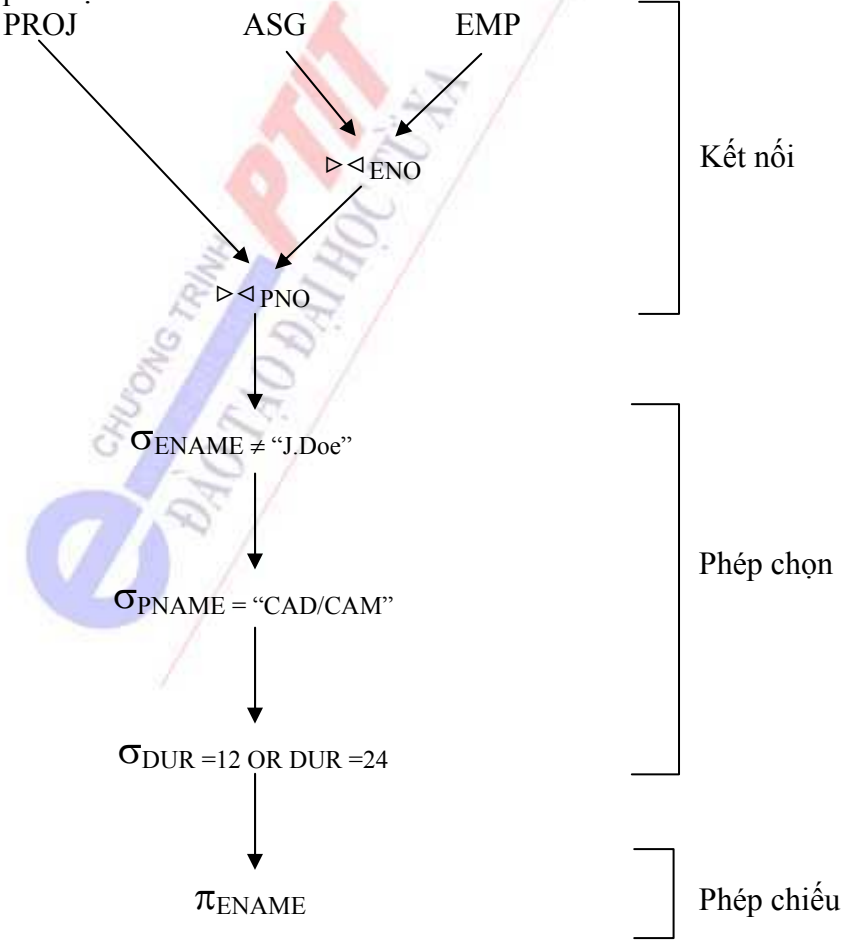
Biến đổi một câu truy vấn phép tính quan hệ thành một cây đại số quan hệ, thực hiện như sau: các node lá tương ứng với các quan hệ có sau mệnh đề FROM. Node gốc được tạo ra như một phép chiếu chứa các thuộc tính kết quả. Các thuộc tính này nằm sau mệnh đề SELECT của câu truy vấn. Lượng từ hóa sau mệnh đề WHERE được biểu thị bằng một chuỗi các phép toán chọn, kết nối, phép hợp...đi từ các node lá đến node gốc. Chuỗi này có thể được cho trực tiếp qua thứ tự xuất hiện của các vị từ và các toán tử.

Ví dụ 3.8: Xét câu truy vấn sau: " Xác định tên các nhân viên trừ J. Doe đã làm cho dự án CAD/CAM trong một hoặc hai năm"

Câu lệnh SQL như sau:

```
SELECT   ENAME
FROM     EMP, ASG, PROJ
WHERE    EMP.ENO = ASG.ENO
         AND   PROJ.PNO = ASG.PNO
         AND   ENAME ≠ "J.Doe"
         AND   PROJ.NAME = "CAD/CAM"
         AND   (DUR =12  OR  DUR =24)
```

Cây đại số quan hệ như sau:



Hình 3.7: Ví dụ về cây đại số quan hệ

Các quy tắc biến đổi cây đại số quan hệ :

Cho các quan hệ R(A), A = {A_1, A_2, .., A_n }, S(B), B = {B_1, B_2, ... , B_n } và T. Có thể nhận được các cây đại số quan hệ tương đương, áp dụng các qui tắc biến đổi (Transformation Rule) sau:

1. *Tính chất giao hoán của các phép toán hai ngôi:*
   - Tích Đề các: R x S $\Leftrightarrow$ S x R
   - Kết nối : R $\triangleright\triangleleft$ S $\Leftrightarrow$ S $\triangleright\triangleleft$ R.
   - Hợp của hai quan hệ: R $\cup$ S $\Leftrightarrow$ S $\cup$ R
   - Qui tắc này không được áp dụng cho hiệu và kết nối nửa.
2. *Tính kết hợp của các phép toán hai ngôi.*
   - Tích Đề các: (R x S) x T $\Leftrightarrow$ R x (S x T)
   - Kết nối : R $\triangleright\triangleleft$ (S $\triangleright\triangleleft$ T) $\Leftrightarrow$ (R $\triangleright\triangleleft$ R) $\triangleright\triangleleft$ T
3. *Tính lũy đẳng của các phép toán đơn ngôi*:
   - Các phép chiếu liên tiếp trên cùng một quan hệ có thể nhóm lại với nhau. Ngược lại một phép chiếu trên nhiều thuộc tính có thể tách ra nhiều phép chiếu liên tiếp nhau. Nếu A' $\subseteq$ A và A'' $\subseteq$ A', khi đó $\pi_{A'}(\pi_{A''}(R)) \Leftrightarrow \pi_{A'}(R)$
   - Các phép chọn liên tiếp $\sigma_{pi(Ai)}$ trên cùng một quan hệ, trong đó p_i là một vị từ được áp dụng cho thuộc tính A_i có thể được nhóm lại. Ngược lại một phép chọn qua một hội các vị từ có thể được tách ra thành nhiều phép chọn liên tiếp.

     $$\sigma_{p1(A1)}(\sigma_{p2(A2)}(R)) \Leftrightarrow \sigma_{p1(A1) \wedge \sigma p2(A2)}(R)$$
3. *Giao hoán phép chọn với phép chiếu:*

   Phép chọn và chiếu trên cùng một quan hệ có thể được giao hoán như sau:

   $$\pi_{A1,...,An}(\sigma_{p(Ap)}(R)) \Leftrightarrow \pi_{A1,...,An}(\sigma_{p(Ap)}(\pi_{A1,...,An,Ap}(R)))$$

5 *Giao hoán phép chọn với các phép toán hai ngôi:*
   - Phép chọn và tích Đề các: $\sigma_{p(Ai)}(R \times S) \Leftrightarrow (\sigma_{p(Ai)}(R)) \times S$
   - Phép chọn và kết nối: $\sigma_{p(Ai)}(R \triangleright\triangleleft_{p(Aj,Bk)} S) \Leftrightarrow \sigma_{p(Ai)}(R) \triangleright\triangleleft_{p(Aj,Bk)} S$
   - Phép chọn và hợp: $\sigma_{p(Ai)}(R \cup T) \Leftrightarrow \sigma_{p(Ai)}(R) \cup \sigma_{p(Ai)}(T)$
6. *Giao hoán phép chiếu với phép toán hai ngôi.*
   - Phép chiếu và tích Đề Các: Nếu C = A' $\cup$ B' trong đó A' $\subseteq$ A và B' $\subseteq$ B, thì $\pi_C(R \times S) \Leftrightarrow \pi_{A'}(R) \times \pi_{B'}(S)$
   - Phép chiếu và kết nối: $\pi_C(R \triangleright\triangleleft_{Aj,Bk} S) \Leftrightarrow \pi_{A'}(R) \triangleright\triangleleft_{Aj,Bk} \pi_{B'}(S)$

     Trong đó $A_i \in A'$, $B_k \in B'$. Vì C = A' $\cup$ B' suy ra $A_i, B_k \in C$.
   - Phép chiếu và hợp $\pi_C(R \cup S) \Leftrightarrow \pi_C(R) \cup \pi_C(S)$.
   - Phép chiếu và hiệu :$\pi_C(R - S) \Leftrightarrow \pi_C(R) - \pi_C(S)$.

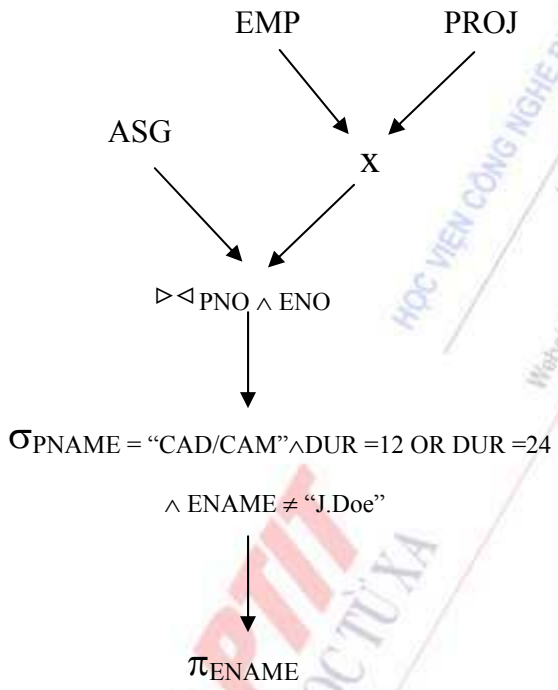
Quy tắc 6 được sử dụng nhiều trong kỹ thuật tối ưu hoá truy vấn.

Áp dụng sáu quy tắc trên có thể tạo ra nhiều cây quan hệ tương đương. Cây trong hình 3.7 tương đương với cây trong hình 3.8. Cây trong hình 3.8 thực hiện phép tích Đề Các với các quan hệ EMP và PROJ, điều này làm cho chi phí thực hiện cao hơn so với cây 3.7. Tuy nhiên

từ một cây quan hệ ban đầu chẳng hạn như cây 3.7 có thể biến đổi thành nhiều cây quan hệ tương đương , trong đó có nhiều cây có cấu trúc "xấu" và cũng có nhiều cây có cấu trúc tốt. Những cây có cấu trúc xấu như cây trong hình 3.8 cần phải loại bỏ.

Để tìm ra một cây đại số quan hệ "tốt hơn", có thể sử dụng quy tắc trên bằng bốn cách:

- Tách các phép toán đơn ngôi, làm đơn giản biểu thức truy vấn.
- Các phép toán đơn ngôi trên cùng một quan hệ có thể nhóm lại thành một phép toán truy xuất trên quan hệ một lần.
- Các phép toán đơn ngôi có thể giao hoán với các phép toán hai ngôi sao cho một số phép toán như chiếu, chọn được thực hiện trước.
- Các phép toán hai ngôi có thể được sắp xếp lại.



Hình 3.8: Cây đại số quan hệ tương đương

Viết lại câu truy vấn nghĩa là tìm một cây có cấu trúc tốt hơn, có chất lượng hơn so với cây gốc. Cây đại số quan hệ trong hình 3.9 được tái tạo từ cây đại số quan hệ trong hình 3.7. Nó được xem là có cấu trúc đạt chất lượng theo nghĩa là nó tránh truy xuất nhiều lần đến cùng một quan hệ. Các phép chọn $\sigma_{ENAME \neq "J.Doe"}(EMP)$, $\sigma_{DUR =12 \text{ OR } DUR =24}(ASG)$ và $\sigma_{PNAME = "CAD/CAM"}(PROJ)$ được thực hiện sớm nhất làm giảm đáng kể kích thước của các quan hệ trung gian. Tiếp theo phép chọn là các phép chiếu được thực hiện:

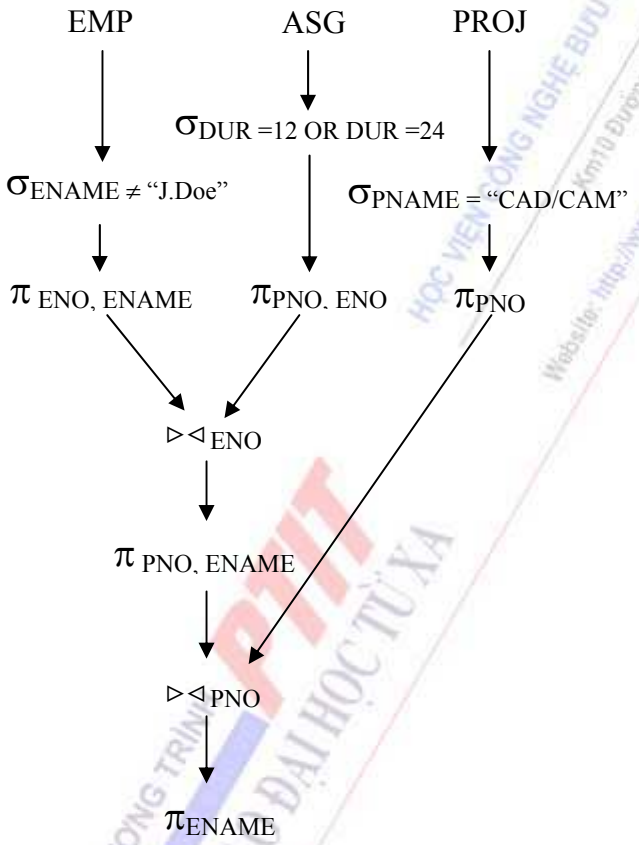
$\pi_{ENO, ENAME}(\sigma_{ENAME \neq "J.Doe"}(EMP))$, $\pi_{PNO, ENO}(\sigma_{DUR =12 \text{ OR } DUR =24}(ASG))$ và $\pi_{PNO}(\sigma_{PNAME = "CAD/CAM"}(PROJ))$. Tiếp theo là các phép kết nối với chi phí thấp. Tuy nhiên, cây đại số quan hệ hình 3.9 còn xa mới đạt được tối ưu. Vì phép chọn trên EMP với vị từ ENAME = "J. Doe" sẽ không hiệu quả, kích thước quan hệ kết quả giảm không đáng kể.

Trên đây là bốn giai đoạn kế liên tiếp trong quá trình phân rã câu truy vấn, bao gồm:

- Chuẩn hoá

- Phân tích.
- Loại bỏ dư thừa
- Viết lại truy vấn

nhằm ánh xạ một câu truy vấn dạng phép tính quan hệ thành câu truy vấn đại số quan hệ. Có thể có nhiều câu truy vấn đại số tương đương với cùng một câu truy vấn ban đầu. Để phân rã có hiệu quả, các biểu thức truy vấn gốc sẽ được tái cấu trúc bằng một số qui tắc biến đổi và các Heuristic. Các qui tắc cho phép tách các phép toán đơn ngôi, nhóm các phép toán đơn ngôi trên cùng một quan hệ, hoán vị các phép toán đơn ngôi với hai ngô và hoán vị các phép toán hai ngôi. Thí dụ về các heuristic là đẩy các phép chọn xuống và thực hiện các phép chiếu càng sớm càng tốt...



Hình 3.9: Cây đại số quan hệ đã được viết lại

3.6 CỤC BỘ HÓA DỮ LIỆU PHÂN TÁN

Đầu vào của tầng hai là một truy vấn đại số trên quan hệ phân tán. Chức năng chủ yếu của tầng này là cục bộ hoá dữ liệu truy vấn sử dụng các thông tin dữ liệu phân tán, nghĩa là tầng cục bộ hoá dữ liệu chịu trách nhiệm dịch câu truy vấn đại số quan hệ trên quan hệ toàn cục sang câu truy vấn đại số quan hệ trên các mảnh vật lý có sử dụng các thông tin được lưu trữ trong lược đồ phân mảnh. Nó xác định mảnh quan hệ nào sẽ được sử dụng trong truy vấn và chuyển đổi câu truy vấn phân tán thành một truy vấn trên mảnh cụ thể. Để tạo một truy vấn trên mảnh được thực hiện bởi hai bước:

Bước 1: Truy vấn phân tán được ánh xạ sang một truy vấn trên mảnh bằng việc thay thế mỗi quan hệ phân tán bằng chương trình xây dựng lại có chứa các phép toán đại số quan hệ thao tác trên mảnh, gọi là chương trình cục bộ hoá (Localization Program)
Bước 2: Truy vấn trên mảnh được đơn giản hoá và xây dựng lại để tạo ra một truy vấn khác tốt hơn. Quá trình đơn giản hoá và xây dựng lại có thể được thực hiện dựa theo cùng một quy tắc được sử dụng trong tầng phân rã.

Phân mảnh được định nghĩa bằng các quy tắc phân mảnh. Các mảnh được biểu diễn dưới dạng quan hệ. Một quan hệ toàn cục có thể được xây dựng lại bằng cách áp dụng các quy tắc phân mảnh đảo và dẫn xuất một chương trình cục bộ hoá mà các toán hạng quan hệ là các mảnh. Để đơn giản, giả thiết không xét các trường hợp các mảnh nhân bản.

3.6.1 Rút gọn cho phân mảnh ngang nguyên thuỷ

Phân mảnh ngang phân tán một quan hệ dựa trên các vị từ chọn (Select Predicate). Ví dụ quan hệ EMP(ENO, ENAME, TITLE) có thể được phân mảnh ngang thành:

$EMP_1 = \sigma_{ENO \leq \text{“E3”}}(EMP)$
$EMP_2 = \sigma_{\text{“E3”} < ENO < \text{“E6”}}(EMP)$
$EMP_3 = \sigma_{ENO > \text{“E6”}}(EMP)$.

Khi đó chương trình cục bộ hoá cho quan hệ phân mảnh ngang là hợp các mảnh:

$EMP = EMP_1 \cup EMP_2 \cup EMP_3$

Vì vậy dạng truy vấn gốc được xác định trên EMP sẽ thu được bằng cách thay EMP bởi ($EMP_1 \cup EMP_2 \cup EMP_3$). Như vậy, để giảm các thao tác truy vấn trên quan hệ đã được phân mảnh theo chiều ngang, trước hết phải xác định rõ cần thao tác trên mảnh nào và sau đó xây dựng lại cây con, xem xét loại bỏ các quan hệ rỗng. Phân mảnh ngang sẽ được sử dụng để làm đơn giản hoá các phép chọn và phép kết nối.

a. Rút gọn phép chọn: Phép chọn thực hiện trên các mảnh có lượng từ hoá mâu thuẫn với lượng từ hoá của quy tắc phân mảnh sinh ra các quan hệ rỗng. Cho một quan hệ R được phân mảnh theo chiều ngang là $R_j = \sigma_{pj}(R)$, j=1..n, quy tắc này có thể được biểu diễn một cách hình thức như sau:

Quy tắc 1: $\sigma_{pi}(R_j) = \varnothing$ nếu $\forall x \in R: \neg (pi(x) \wedge pj(x))$,

trong đó p_i , p_j là các vị từ chọn, x là một bộ.

Ví dụ vị từ ENO = “E1” mâu thuẫn với các vị từ của mảnh EMP_2 và EMP_3, nghĩa là không có bộ nào thoả vị từ ENO = “E1”.

Ví dụ 3.9: Xét câu truy vấn sau:

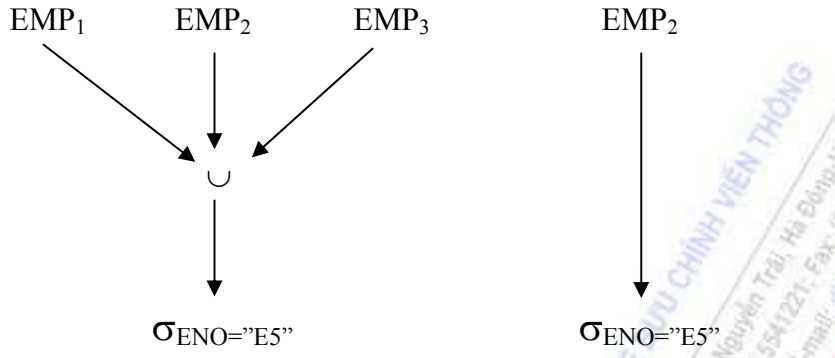
```
SELECT      *
FROM        EMP
WHERE       ENO = :E5”
```

Áp dụng cách tiếp cận thô sơ để cục bộ hoá EMP từ EMP_1, EMP_2 và EMP_3, cho câu truy vấn gốc hình 3.10[a] bằng cách hoán vị phép chọn và phép hợp. Dễ dàng nhận thấy vị từ chọn mâu thuẫn với EMP_1 và EMP_3. Câu truy vấn đã rút gọn chỉ ứng dụng có một mảnh EMP_2 , như trong hình 3.10b.

$\sigma_{ENO = :E5"} (EMP_1 \cup EMP_2 \cup EMP_3)$

$= \sigma_{ENO = :E5"} (EMP_1) \cup \sigma_{ENO = :E5"} (EMP_2) \cup \sigma_{ENO = :E5"} (EMP_3)$

$= \sigma_{ENO = :E5"} (EMP_2)$



(a) Câu truy vấn gốc (b) Câu truy vấn đã rút gọn

Hình 3.10: Rút gọn phân mảnh ngang với phép chọn

b. Rút gọn với phép kết nối: Phép kết nối thực hiện trên các phân mảnh ngang có thể đơn giản khi kết nối dựa theo các thuộc tính kết nối., bằng cách phân phối kết nối trên các hợp và sau đó loại bỏ các kết nối không tác dụng. Việc phân phối các kết nối trên phép hợp được phát biểu như sau:

$$(R_1 \cup R_2) \triangleright\triangleleft S = (R_1 \triangleright\triangleleft S) \cup (R_2 \triangleright\triangleleft S)$$

Trong đó, R_i là các mảnh của R và S là một quan hệ.

Bằng phép biến đổi này, các phép hợp có thể được di chuyển xuống dưới trong cây đại số quan hệ, nên tất cả các phép kết nối có thể của các mảnh đều được lộ ra. Các phép kết nối vô dụng được xác định khi lượng từ hoá các mảnh có mâu thuẫn. Giả sử các mảnh R_i và R_j được xác định theo các vị từ chọn p_i và p_j trên cùng một quan hệ, quy tắc đơn giản hoá có thể được phát biểu như sau

Quy tắc 2: $R_i \triangleright\triangleleft R_j = \varnothing$ nếu $\forall x \in R_i , \forall y \in R_j : \neg (p_i(x) \wedge p_j(y))$

Việc xác định các kết nối vô dụng có thể được thực hiện bằng cách chỉ xét các vị từ của các mảnh Áp dụng quy tắc này cho phép kết nối hai quan hệ được cài đặt như các nối từng phần song song các mảnh. Câu truy vấn rút gọn chưa chắc đã đơn giản hơn câu truy vấn gốc, nếu như trong câu truy vấn gốc có rất nhiều nối từng phần, vif rất ít các vị từ phân mảnh mâu thuẫn. Như vậy câu truy vấn rút gọn sẽ tốt hơn khi có ít các nối từng phần. Trường hợp xấu nhất xảy ra khi mỗi mảnh của quan hệ này được kết nối với mỗi mảnh khác của quan hệ khác, nghĩa là thực hiện phép tích Đề Các hai quan hệ. Ưu điểm của truy vấn rút gọn theo phép kết nối là các nối từng phần có thể được thực hiện song song và vì vậy nó sẽ làm giảm thời gian đáp ứng.

Ví dụ 3.10: Giả sử quan hệ EMP(ENO, ENAME, TITLE) được phân mảnh ngang thành EMP_1, EMP_2 và EMP_3.

$EMP_1 = \sigma_{ENO \leq "E3"} (EMP)$

$EMP_2 = \sigma_{"E3" < ENO < "E6"} (EMP)$

$EMP_3 = \sigma_{ENO>"E6"}(EMP)$.

Quan hệ ASG(ENO, PNO, RESP, DUR) được phân mảnh như sau:

$ASG_1 = \sigma_{ENO\leq "E3"}(ASG)$

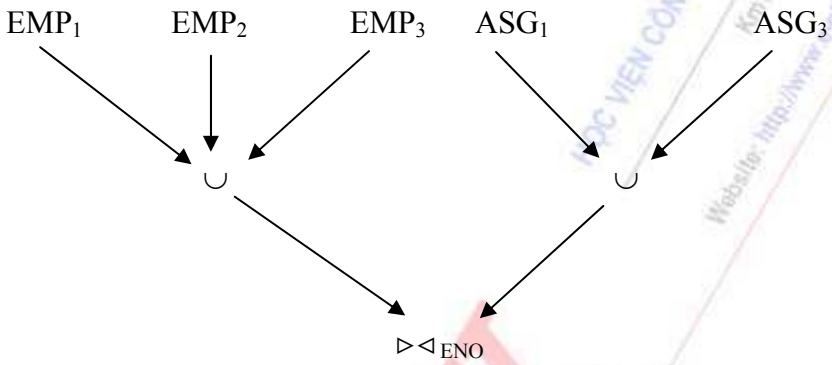
$ASG_2 = \sigma_{ENO > "E3"}(ASG)$

EMP_1 và ASG_1 cùng định nghĩa bởi vị từ ENO≤ "E3" . Vị từ định nghĩa trong ASG_2 là hợp của các vị từ được định nghĩa trong EMP_2 và EMP_3:

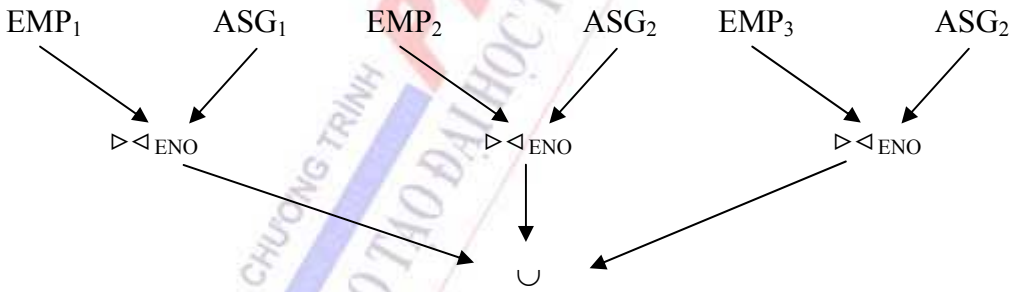
ENO > "E3" = ("E3" < ENO < "E6") ∪ (ENO>"E6"). Xét câu truy vấn sau:

```
SELECT      *
FROM        EMP, ASG
WHERE       EMP.ENO = ASG.ENO
```

Câu truy vấn được rút gọn bằng cách phân phối các nối trên các hợp và áp dụng quy tắc 2 có thể cài đặt như hợp của ba nối từng phần được thực hiện song song.



Hình 3.11a Cây đại số quan hệ truy vấn gốc



Hình 3.11b Rút gọn phân mảnh ngang với phép kết nối

3.6.2 Rút gọn cho phân mảnh dọc

Phân mảnh dọc phân tán một quan hệ dựa trên các thuộc tính chiếu. Vì vậy phép kết nối sẽ là phép toán tái xây dựng các phân mảnh dọc, Chương trình cục bộ hoá cho quan hệ phân mảnh dọc bao gồm các kết nối của các mảnh trên các thuộc tính chung.

Ví dụ 3.11: Giả sử quan hệ EMP(ENO, ENAME, TITLE) được phân mảnh như sau:

Phân mảnh ngang phân tán một quan hệ dựa trên các vị từ chọn (Select Predicate). Ví dụ quan hệ EMP(ENO, ENAME, TITLE) có thể được phân mảnh ngang thành:

$EMP_1 = \pi_{ENO, ENAME}(EMP)$

$EMP_2 = \pi_{ENO, TITLE}(EMP)$

Khi đó chương trình cục bộ hoá cho quan hệ phân mảnh dọclà:

$EMP = EMP_1 \rhd\lhd_{ENO} EMP_2$

Cũng như phân mảnh ngang, các câu truy vấn trên các mảnh dọc được rút gọn bằng cách xác định các quan hệ trung gian vô dụng và loại bỏ các cây con đã sinh ra chúng. Phép chiếu trên một mảnh dọc không có thuộc tính chung với các thuộc tính chiếu sinh ra các quan hệ vô dụng có thể không rỗng. Cho một quan hệ R định nghĩa trên tập các thuộc tính $A = \{A_1, A_2,...,A_n\}$ và được phân thành $R_i = \pi_{A'}(R)$, i=1..k, $A' \subseteq A$. Quy tắc được phát biểu một cách hình thức như sau:

Quy tắc 3: $\pi_{D, K}(R_i)$ là vô dụng nếu tập các thuộc tính chiếu D không nằm trong A'.

Ví dụ 3.12: Giả sử

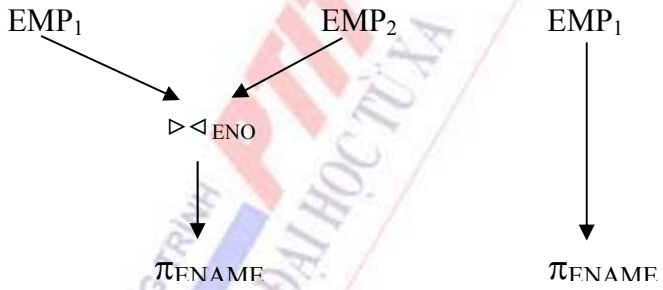
$EMP_1 = \pi_{ENO, ENAME}(EMP)$

$EMP_2 = \pi_{ENO, TITLE}(EMP)$

Xét câu truy vấn SQL như sau

```
SELECT    ENAME
FROM      EMP
```

Bằng cách hoán vị phép chiếu và phép kết nối , nghĩa là thực hiện phép chiếu trên các thuộc tính ENO và ENAME , khi đó có thể nhận thấy rằng phép chiếu trên thuộc tính ENAME trên quan hệ EMP_2 là vô dụng, vì ENAME không phải là thuộc tính của EMP_2. Vì vậy phép chiếu chỉ cần thực hiện trên EMP_1.



Hình 3.12: Rút gọn phân mảnh dọc

3.6.3 Rút gọn cho phân mảnh dẫn xuất

Phép kết nối thường xuyên xảy ra và có chi phí cao. Tối ưu hoá bằng cách sử dụng các phân mảnh ngang nguyên thuỷ khi các quan hệ nối được phân mảnh theo các thuộc tính nối. Trong trường hợp này nối của hai quan hệ được cài đặt như hợp của các nối từng phần. Tuy nhiên phương pháp này ngăn cản không cho một trong các quan hệ phân mảnh theo một phép chọn trên một thuộc tính khác. Phân mảnh ngang dẫn xuất phân phối hai quan hệ, cải thiện khả năng xử lý các điểm giao nhau giữa các phép chọn và phép kết nối. Nếu quan hệ R phân mảnh dẫn xuất theo quan hệ S, các mảnh của R và S có giá trị như nhau ở thuộc tính kết nối sẽ nằm cùng vị trí. Quan hệ S có thể phân mảnh theo một vị từ chọn.

Vì các bộ của quan hệ R được đặt tuỳ chọn theo các bộ của S. để cho đơn giản, giả sử chỉ xét phân mảnh dẫn xuất chỉ được sử dụng cho mối liên hệ một - nhiều, trong đó một bộ của S tương ứng với n bộ của R và một bộ của R chỉ khớp đúng với một bộ của S.

Ví dụ 3.13: Cho mối quan hệ một - nhiều EMP → ASG. Giả sử ASG được phân mảnh gián tiếp theo các quy tắc sau:

$ASG_1 = ASG \triangleright\prec_{ENO} EMP_1$

$ASG_2 = ASG \triangleright\prec_{ENO} EMP_2$

Trong đó $EMP_1 = \sigma_{TITLE = \text{“Programmer”}}(EMP)$

$EMP_2 = \sigma_{TITLE \neq \text{“Programmer”}}(EMP)$

Chương trình cục bộ hoá cho quan hệ phân mảnh ngang là

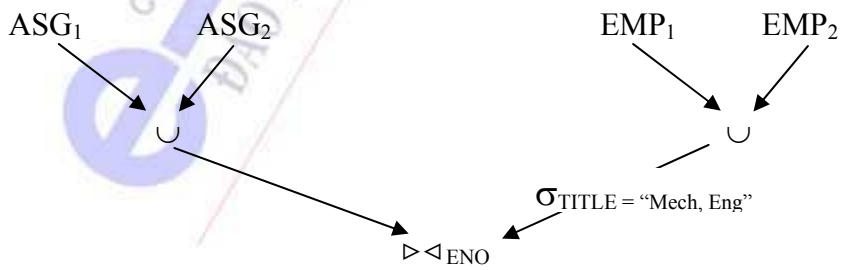
$ASG = ASG_1 \cup ASG_2$

Các câu truy vấn trên các mảnh dẫn xuất có thể được rút gọn bằng cách phân phối các nối trên các phép hợp và áp dụng quy tắc 2. Vì quy tắc phân mảnh chỉ rõ các bộ sẽ khớp với nhau, một số nối sinh ra quan hệ rỗng, các vị từ phân mảnh có mâu thuẫn. Chẳng hạn các vị từ của ASG_1 và EMP_2 có mâu thuẫn, vì vậy $ASG_1 \triangleright\triangleleft EMP_2 = \varnothing$

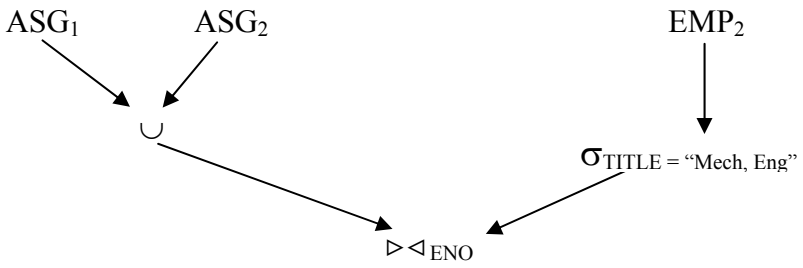
Xét câu truy vấn sau

```
SELECT    *
FROM      EMP, ASG
WHERE     EMP.ENO = ASG.ENO
          AND TITLE = “Mech, Eng”
```

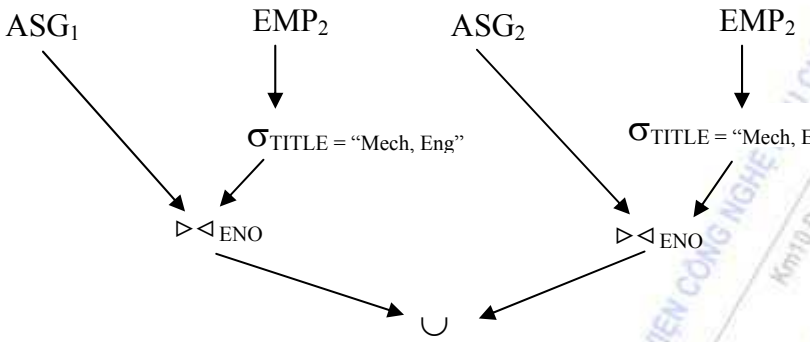
Câu truy vấn gôc được thao tác trên các mảnh EMP_1, EMP_2, ASG_1 và ASG_2. (hình 3.13a). Thực hiện phép chọn trên các mảnh EMP_1, EMP_2 , vì vị từ chọn mâu thuẫn trên mảnh EMP_1, nên kết quả câu truy vấn rút gọn thu được như trong hình 3.13b. Nhằm xác định các vị từ kết nối mâu thuẫn, cần phải phân phối các nối trên các hợp. Kết quả là cây hình 3.13c. Cây con bên trái nối hai mảnh ASG1 và EMP2 với các lượng từ hoá mâu thuẫn bởi các vị từ chọn TITLE = “Programmer: trong ASG_1 và TITLE ≠ “Programmer” trong EMP_2 . Vì vậy có thể loại bỏ cây bên trái và thu được kết quả câu truy vấn rút gọn như được chỉ ra trong hình 3.13d. Với thí dụ này muốn minh hoạ một điều là giá trị phân mảnh trong việc cải thiện hiệu năng của các câu truy vấn phân tán.



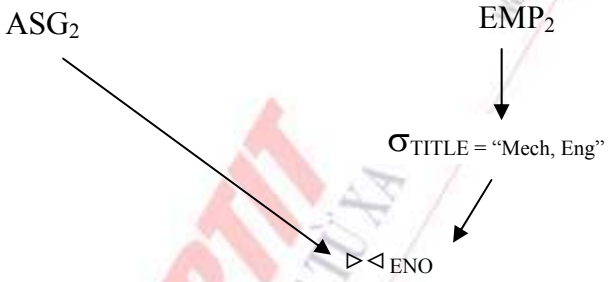
(a) Câu truy vấn gốc



(b) Câu truy vấn sau khi đẩy phép chọn xuống



(c) Truy vấn sau khi đẩy các phép hợp xuống



(d) Câu truy vấn đã rút gọn sau khi loại cây con bên trái

Hình 3.13 Rút gọn cho phân mảnh gián tiếp

3.6.4 Rút gọn cho phân mảnh hỗn hợp

Phân mảnh hỗn hợp bao gồm việc phân mảnh ngang và phân mảnh dọc. Mục đích của phân mảnh hỗn hợp là hỗ trợ một cách hiệu quả các câu truy vấn có chứa các phép chọn, phép chiếu va phép kết nối. Chương trình hoá cục bộ cho một quan hệ phân mảnh hỗn hợp có sử dụng phép hợp và kết nối các mảnh. Điều đáng quan tâm là để tối ưu hoá một phép toán hay tổ hợp các phép toán luôn luôn phải trả chi phí cao cho các phép toán khác. Ví dụ phân mảnh hỗn hợp dựa trên phép chiếu - chọn sẽ làm cho phép chiếu hoặc phép chọn kém hiệu quả hơn so với phân mảnh ngang hoặc phân mảnh dọc.

Ví dụ 3.14: Phân mảnh hỗn hợp của quan hệ EMP như sau:

EMP1 = $\sigma_{ENO \leq "E4"}(\pi_{ENO,ENAME}(EMP))$

EMP2 = $\sigma_{ENO > "E4"}(\pi_{ENO,ENAME}(EMP))$

EMP3 = $\pi_{ENO,TITLE}(EMP)$

Chương trình cục bộ hoá như sau:

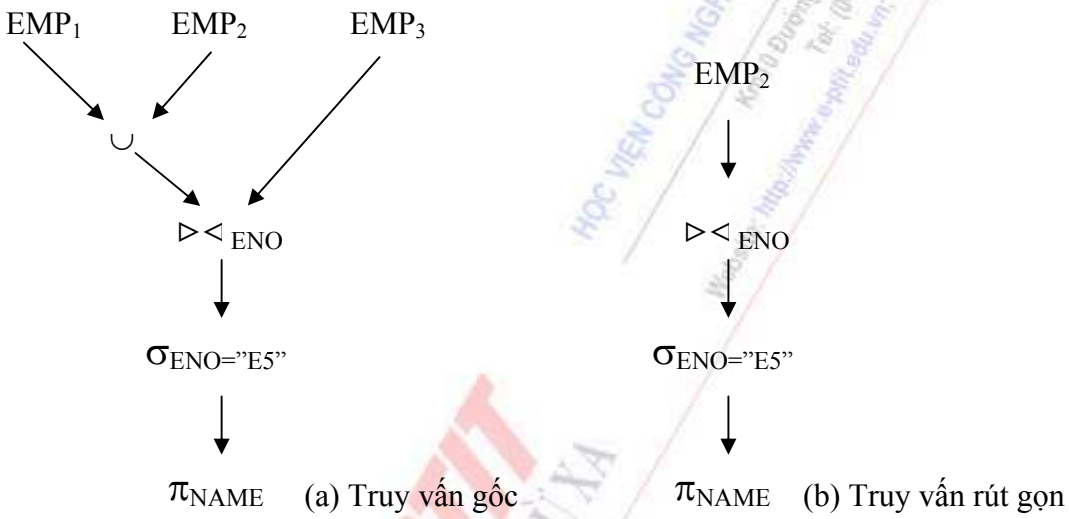
$EMP = (EMP_1 \cup EMP_2) \rhd\lhd_{ENO} EMP_3$.

Các truy vấn trên những mảnh hỗn hợp có thể được rút gọn bằng cách kết hợp lần lượt các quy tắc trong phân mảnh ngang nguyên thuỷ, phân mảnh dọc và phân mảnh ngang dẫn xuất. Các quy tắc này có thể tóm tắt như sau:

1. Loại bỏ các quan hệ rỗng được tạo ra do các phép chọn mâu thuẫn nhau trên các mảnh ngang.
2. Loại bỏ các kết nối vô dụng được tạo do các phép chiếu trên các mảnh dọc.
3. Phân phối các kết nối cho các phép hợp nhằm cô lập và loại bỏ các kết nối vô dụng.

Ví dụ 3.15: Xét câu truy vấn sau

SELECT ENAME
FROM EMP
WHERE ENO = "E5" $\rhd\lhd$



Hình 3.14: Rút gọn phân mảnh hỗn hợp

CÂU HỎI VÀ BÀI TẬP

1. Xử lý truy vấn trong các hệ cơ sở dữ liệu quan hệ phân tán. là:
   A. Cung cấp các phương tiện xây dựng các câu truy vấn và thực hiện tối ưu hoá truy vấn .
   B. Cung cấp các phương tiện thực hiện tối ưu hoá truy vấn .
   C. Cung cấp các câu truy vấn và thực hiện tối ưu hoá truy vấn .
2. Câu truy vấn phân tán là:
   A. Một chuỗi các thao tác đại số quan hệ trên CSDL cục bộ.
   B. Một chuỗi các thao tác đại số quan hệ trên CSDL cục bộ tối ưu hoá các nguồn tài nguyên. và trao đổi truyền thông.
   C. Một chuỗi các thao tác đại số quan hệ trên các mảnh dữ liệu được phân rã, được mở rộng với các thao tác truyền thông và tối ưu các nguồn tài nguyên.
3. Các phương pháp tối ưu cơ bản:
   A. Biến đổi câu truy vấn tương đương và có chi phí thấp.

B. Chọn một biểu thức có chi phí thời gian và sử dụng tài nguyên là ít nhất.
C. Biến đổi câu truy vấn tương đương

4. Mục đích của việc xử lý truy vấn trong môi trường phân tán là:
   A. Thực hiện tối ưu hoá truy vấn.
   B. Biến đổi thành câu truy vấn tương đương.
   C. Tối ưu chi phí sử dụng tài nguyên của mạng.
5. Các kiểu tối ưu hoá
   A. Lựa chọn trong các giải pháp có chi phí là nhỏ nhất.
   B. Phương pháp tìm kiếm vét cạn, giải pháp ngẫu nhiên.
   C. Giải pháp thay thế phép kết nối bằng các tổ hợp các nối nửa
6. Thời điểm tối ưu hoá
   A. Kiểu tĩnh
   B. Tại các thời điểm khác nhau phụ thuộc thời gian thực hiện truy vấn.
   C. Kiểu động
7. Ưu điểm tối ưu hoá truy vấn theo kiểu tĩnh (Statically):
   A. Thực hiện khi biên dịch, chi phí giảm dần qua nhiều lần thực hiện. Kích thước của các quan hệ trung gian không biết trước
   B. Thực hiện khi biên dịch, chi phí giảm dần qua nhiều lần thực hiện.
   C. Thực hiện khi bắt đầu truy vấn, chi phí giảm dần qua nhiều lần thực hiện. Kích thước của các quan hệ trung gian không biết trước.
8. Ưu điểm tối ưu hoá truy vấn theo kiểu động
   A. Thực hiện khi biên dịch, chi phí giảm dần qua nhiều lần thực hiện. Kích thước của các quan hệ trung gian không biết trước
   B. Được thực hiện khi truy vấn. Thao tác tiếp theo tối ưu dựa trên kết quả của các thao tác trước đó.
   C. Đánh giá kích thước của các quan hệ trung gian không cần thiết.
9. Nhược điểm của phương pháp động là:
   A. Các thao tác tối ưu hoá có chi phí cao. Lặp lại nhiều lần cho mỗi thao tác.
   B. Các thao tác có chi phí cao, chi phí tăng dần qua nhiều lần thực hiện
   C. Kích thước của các quan hệ trung gian không phù hợp cho xử lý truy vấn.
10. Tối ưu hoá truy vấn hỗn hợp có các ưu điểm:
   A. Tối ưu hoá truy vấn tĩnh, tránh được các đánh giá không chính xác gây ra.
   B. Tối ưu hoá truy vấn động , có thể phát hiện có sự khác biệt giữa kích thước dự đoán và kích thước thực tế của các quan hệ trung gian.
   C. Của tối ưu hoá truy vấn động , hạn chế các nhược của truy vấn tĩnh.
11. Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp tĩnh, có thể :
   A. Sử dụng một vị trí hay nhiều vị trí.
   B. Sử dụng một vị trí
   C. Sử dụng nhiều vị trí.
12. Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp quyết định tập trung:
   A. Có một vị trí đưa ra giải pháp.
   B. Có nhiều vị trí đưa ra giải pháp.
   C. Có một hoặc nhiều vị trí đưa ra giải pháp.

13. Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp hỗn hợp
    A. Có một vị trí quyết định chính, các vị trí khác đưa ra các quyết định cục bộ.
    B. Có thể được phân tán cho nhiều vị trí tham gia.
    C. Đòi hỏi phải biết toàn bộ về các thông tin cục bộ.
14. Quá trình cục bộ hoá là quá trình:
    A. Ánh xạ câu truy vấn phân tán mô tả trên quan hệ toàn cục thành các câu truy vấn trên các mảnh
    B. Nhân bản các mảnh ở nhiều vị trí khác nhau.
    C. Giảm số lần truyền thông
15. Thao tác nửa kết nối:
    A. Làm giảm kích thước của các quan hệ trung gian, làm giảm dữ liệu cần trao đổi giữa các vị trí.
    B. Làm giảm số lượng các thông điệp và thời gian xử lý cục bộ.
    C. Làm giảm các thao tác tối ưu hoá truy vấn
16. Thứ tự đúng caác tầng của quá trình xử lý truy vấn là:
    A. Tầng phân rã truy vấn, tập trung hoá dữ liệu, tối ưu hoá truy vấn toàn cục và tối ưu hoá truy vấn cục bộ.
    B. Tầng tập trung hoá dữ liệu, tối ưu hoá truy vấn toàn cục, tối ưu hoá truy vấn cục bộ và phân rã truy vấn,
    C. Tầng tập trung hoá dữ liệu, phân rã truy vấn, tối ưu hoá truy vấn toàn cục và tối ưu hoá truy vấn cục bộ.
17. Phân rã truy vấn và tập
    A. Có chức năng ánh xạ câu truy vấn phân tán ở dạng phép tính quan hệ thành câu truy vấn đại số trên quan hệ toàn cục.
    B. Có chức năng thực hiện tối ưu hoá truy vấn tại một vị trí tập trung và sử dụng các thông tin toàn cục.
    C. Có chức năng biến đổi phân rã truy vấn phân tán trên các quan hệ toàn cục.
18. Phân rã câu truy vấn có thể thực hiện các bước liên tiếp nhau:
    A. Bước chuẩn hoá, phân tích, loại bỏ dư thừa và xây dựng lại câu truy vấn
    B. Bước chuẩn hoá, phân tích và loại bỏ dư thừa
    C. Bước phân tích, loại bỏ dư thừa và xây dựng lại câu truy vấn
19. Chức năng chủ yếu của tầng cục bộ hoá dữ liệu phân tán:
    A. Chịu trách nhiệm chuyển câu truy vấn trên quan hệ toàn cục sang câu truy vấn trên các mảnh.
    B. Cung cấp các thông tin lưu trữ trong lược đồ phân mảnh cho quá trình cục bộ hoá phân tán.
    C. Xác định mảnh được sử dụng trong truy vấn và chuyển đổi câu truy vấn phân tán thành một truy vấn trên mảnh cụ thể.
20. Rút gọn phép chọn cho phân mảnh ngang nguyên thuỷ
    A. Bằng cách hoán vị phép chọn và phép hợp.
    B. Bằng cách hoán vị phép chọn và phép chiếu.
    C. Bằng cách hoán vị phép chọn và phép kết nối
21. Rút gọn phép kết nối cho phân mảnh ngang nguyên thuỷ

A. Bằng cách phân phối các phép kết nối trên các phép hợp
B. Bằng cách phân phối các phép kết nối dưới các phép hợp
C. Bằng cách phân phối các phép kết nối dưới các phép giao

22. Rút gọn cho phân mảnh dọc
A. Bằng cách hoán vị phép chiếu và phép kết nối.
B. Bằng cách hoán vị phép chọn và phép kết nối.
C. Bằng cách hoán vị phép chiếu và phép chọn

23. Các câu truy vấn trên các mảnh dẫn xuất có thể được rút gọn
A. Bằng cách phân phối các phép kết nối trên các phép hợp.
B. Bằng cách phân phối các phép kết nối dưới các phép hợp
C. Bằng cách phân phối các phép kết nối dưới các phép giao

24. Các truy vấn trên những mảnh hỗn hợp có thể được rút gọn bằng cách:
A. Kết hợp các quy tắc trong phân mảnh ngang và phân mảnh dọc
B. Kết hợp các quy tắc trong phân mảnh ngang nguyên thuỷ và phân mảnh dọc.
C. Kết hợp các quy tắc trong phân mảnh ngang dẫn xuất và phân mảnh dọc.

CHƯƠNG IV: XỬ LÝ TRUY VẤN TRONG CƠ SỞ DỮ LIỆU QUAN HỆ PHÂN TÁN

Nội dung của chương bao gồm các phần sau:

- Khái niệm truy vấn
- Các phép toán đại số quan hệ
- Đặc trưng của xử lý truy vấn
- Phân lớp xử lý truy vấn
- Phân rã truy vấn
- Cục bộ hóa dữ liệu phân tán
- Tối ưu hóa truy vấn phân tán.
- Các thuật toán tối ưu hóa truy vấn phân tán

4.1 GIỚI THIỆU

Chương này sẽ giới thiệu tổng quan về xử lý truy vấn trong các hệ cơ sở dữ liệu quan hệ phân tán. Bộ xử lý truy vấn cung cấp các phương tiện cho người sử dụng có thể xây dựng các câu truy vấn và thực hiện tối ưu hoá truy vấn trong các hệ cơ sở dữ liệu phân tán DBMS. Quá trình tối ưu hoá truy vấn có thể thực hiện khi DBMS tập trung và dữ liệu phân tán. Trong truy vấn phân tán, chi phí truyền thông và thời gian đáp ứng truy vấn là những vấn đề cần quan tâm.

Truy vấn phân tán là một chuỗi các thao tác dữ liệu được thực hiện trên các mảnh quan hệ phân rã. Cụ thể như sau:

- Câu truy vấn phải được phân rã thành một chuỗi các thao tác đại số quan hệ.
- Dữ liệu được truy nhập bởi truy vấn là những mảnh dữ liệu được phân rã, gọi là dữ liệu cục bộ.
- Phép truy vấn đại số trên các mảnh phải được mở rộng với các thao tác truyền thông và tối ưu hoá chức năng tham chiếu các nguồn tài nguyên.

4.2 VẤN ĐỀ XỬ LÝ TRUY VẤN

4.2.1 Đặt vấn đề

Chức năng chính của bộ xử lý truy vấn là chuyển đổi một truy vấn mức cao (phép tính quan hệ) sang một truy vấn mức thấp tương đương (đại số quan hệ). Quá trình chuyển đổi cùng cho một kết quả như nhau.

Có nhiều giải pháp chuyển đổi, mỗi giải pháp khác nhau có thể tiêu thụ tài nguyên của mạng máy tính khác nhau. Vì vậy, cần phải lựa chọn một giải pháp khi thực hiện, nó tiêu thụ tài nguyênt là tối thiểu.

Có 2 phương pháp tối ưu cơ bản: phương pháp biến đổi một câu vấn tin có mức cao thành câu vấn tin tương đương ở mức thấp hơn dưới dạng biểu thức đại số quan hệ và phương pháp chọn lựa trong số các câu vấn tin dạng biểu thức đại số quan hệ tương đương, một biểu thức có chi phí thời gian thực hiện và chi phí sử dụng tài nguyên là ít nhất.

Ví dụ 4.1: Xét 2 quan hệ:

EMP (ENO, ENAME, TITLE) và ASG (ENO, PNO, RESP, DUR).

ENO	ENAME	TITLE
E1	J.Doe	Elect.Eng
E2	M.Smith	Analyst
E2	M.Smith	Analyst
E3	A.Lee	Mech.Eng
E3	A.Lee	Mech.Eng
E4	J.Miller	Programmer
E5	B.Casey	Syst.Anal
E6	L.Chu	Elect.Eng
E7	R.David	Mech.Eng
E8	J.Jones	Syst.Anal

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

Và câu truy vấn : " Tên của các nhân viên là giám đốc (Manager) dự án".

Câu truy vấn dưới dạng phép tính quan hệ theo cú pháp SQL là:

```
SELECT    ENAME
FROM      EMP, ASG
WHERE     EMP.ENO = ASG.ENO
          AND RESP = "Manager".
```

Câu truy vấn trên được biểu diễn dưới dạng biểu thức đại số quan hệ sau

$$\pi_{\text{ENAME}} (\sigma_{\text{RESP="Manager"} \wedge \text{EMP.ENO} = \text{ASG.ENO}} (\text{EMP} \times \text{ASG}))$$

Phép toán trên được chuyển đổi tương đương, cho cùng một kết quả nhưng có chi phí thời gian và sử dụng tài nguyên ít hơn:

$$\pi_{\text{ENAME}}(\text{EMP} \triangleright\!\triangleleft_{\text{ENO}} (\sigma_{\text{RESP} = \text{"Manager"}} (\text{ASG}))) .$$

Trong trường hợp cơ sở dữ liệu tập trung, việc chuyển đổi câu truy vấn sang các phép đại số quan hệ được tiến hành một cách thuận lợi. Chức năng chính của bộ xử lý tập trung là lựa chọn phép truy vấn đại số quan hệ tối ưu trong các phép đại số tương đương.

Trong môi trường phân tán, các phép đại số quan hệ không đủ để mô tả các giải pháp thực hiện. Nó phải được cung cấp thêm các các thao tác để chuyển đổi dữ liệu giữa các vị trí. Việc lựa chọn thứ tự các thao tác đại số quan hệ, bộ xử lý truy vấn cũng phải lực chọn các vị trí tốt nhất để xử lý dữ liệu và đường truyền thông mà dữ liệu sẽ lưu chuyển. Việc này sẽ làm tăng số các giải pháp lựa chọn cần lựa chọn trong số đó một giải pháp thực hiện. Vì vậy việc xử lý truy vấn phân tán càng trở nên khó khăn hơn.

Ví dụ 4.2 Ví dụ này mô tả tầm quan trọng của việc lựa chọn vị trí và việc truyền thông từ truy vấn đại số quan hệ được lựa chọn với CSDL được phân mảnh. Xét câu vấn tin với biểu thức đại số quan hệ ví dụ trên:

$$\pi_{\text{ENAME}}(\text{EMP} \triangleright\!\triangleleft_{\text{ENO}} (\sigma_{\text{RESP} = \text{"Manager"}} (\text{ASG})))$$

Giả sử hai quan hệ EMP và ASG được phân mảnh theo chiều ngang như sau:

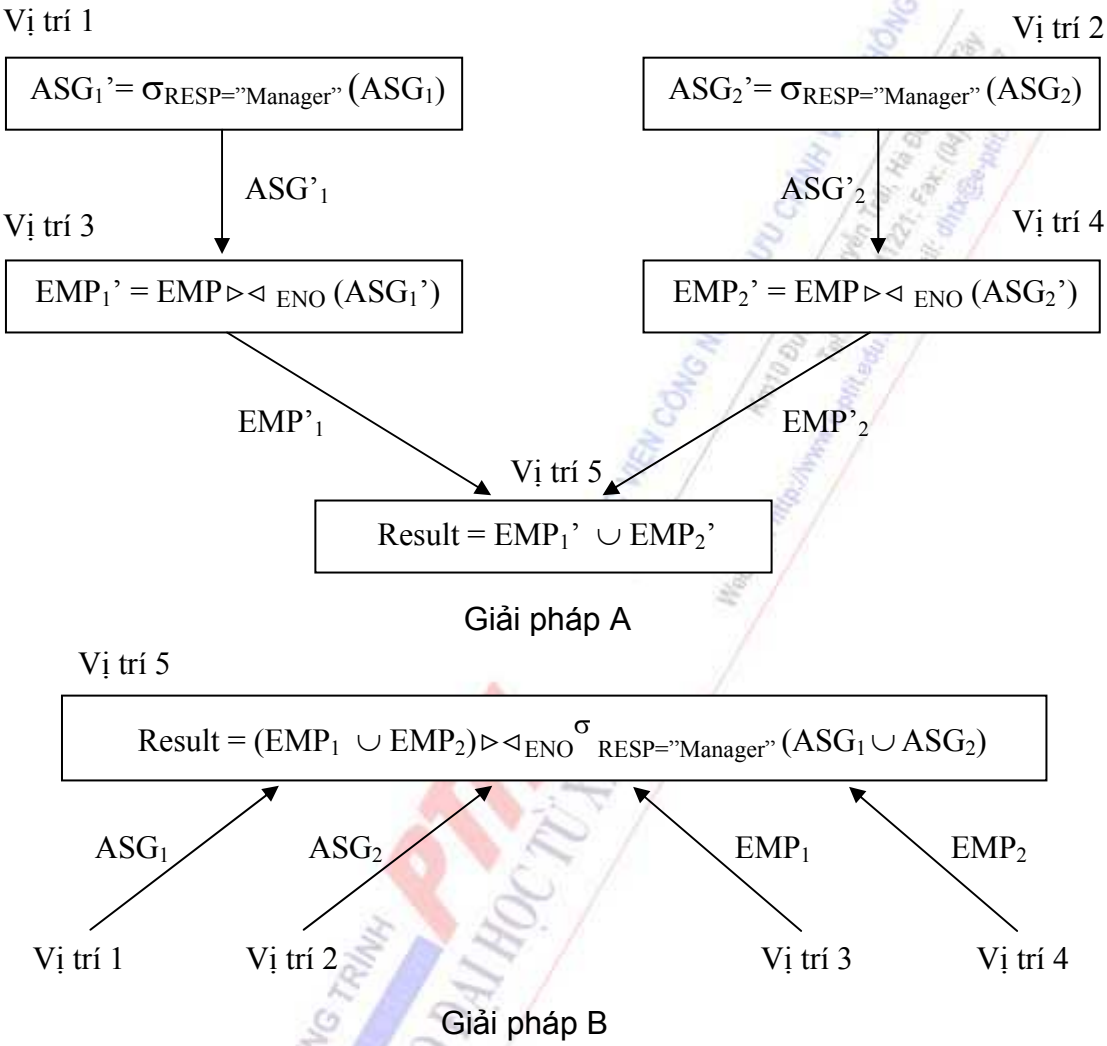
$$\text{EMP}_1 = \sigma_{\text{ENO} \le \text{"E3"}} (\text{EMP}).$$

$EMP_2 = \sigma_{ENO > \text{“E3”}} (EMP)$.

$ASG_1 = \sigma_{ENO \leq \text{“E3”}} (ASG)$.

$ASG_2 = \sigma_{ENO > \text{“E3”}} (ASG)$.

Các mảnh EMP_1, EMP_2, ASG_1, ASG_2 được lưu trữ tại các vị trí 1, 2, 3, 4 và các kết quả được lưu trữ tại vị trí 3.



Hình 4.1 Các giải pháp truy vấn tương đương

Mũi tên đi kèm ký hiệu quan hệ để chỉ hệ thống sẽ truyền quan hệ kết quả từ vị trí i đến vị trí j. Ví dụ trong giải pháp A, tại vị trí 1 sau khi thực hiện phép chọn trên quan hệ ASG_1 thoả biểu thức RESP=”Manager”, kết quả ASG'_1 sẽ được truyền từ vị trí 1 sang vị trí 3 để tham gia phép kết nối với quan hệ EMP_1.

Hai giải pháp A và B thực hiện truy vấn là tương đương trong đó:

- Giải pháp A: Sử dụng hai quan hệ EMP và ASG được phân mảnh theo cùng một cách để thực hiện lựa chọn và kết nối các thao tác một cách song song các phép chọn và kết nối.
- Giải pháp B: Tập trung tất cả các dữ liệu trong toán hạng tại vị trí kết quả trước khi xử lý truy vấn.

Để có thể đánh giá chi phí nguồn tài nguyên của hai giải pháp trên, ký hiệu

- Chi phí để thao tác truy xuất một bộ (tuple access) là một đơn vị, ký hiệu tupacc.
- Chi phí để truyền một bộ (tuple transfer) là 10 đơn vị, ký hiệu tuptrans.
- Giả sử lực lượng của quan hệ EMP và ASG sẽ là 400 và 1000 bản ghi.
- Giả sử có 20 Manager trong quan hệ ASG.

Khi đó, tổng chi phí của giải pháp A sẽ là:

1.	Tạo ASG' bằng phép chọn ASG cần (10+10)**tupacc*	= 20.
2.	Truyền ASG'đến các vị trí của EMP cần 10+10)**tuptrans*	= 200
3.	Tạo EMP' bằng phép kết nối ASG' và EMP cần (10+10)* *tupacc*2*	= 40
4.	Truyền EMP' đến vị trí kết quả cần (10+10)**tuptrans*	= 200
	Tổng chi phí	*460*

Tổng chi phí của giải pháp B được tính như sau:

1.	Truyền EMP đến vị trí 5 cần 400* tuptrans	= 4 000
2.	Truyền ASG đến vị trí 5 cần 400*tuptrans	= 10 000
3.	Tạo ASG' bằng phép chọn ASG cần 1000*tupacc	= 1 000
4.	Kết nối EMP và ASG' cần 400*20*tupacc	= 8000
	Tổng chi phí	*23 000*

Như vậy chi phí để thực hiện giải pháp A thấp hơn chi phí cho giải pháp B.

4.2.2 Mục đích của việc xử lý truy vấn

Mục đích của việc xử lý truy vấn trong môi trường phân tán là biến đổi một truy vấn ở mức cao trên một cơ sở dữ liệu phân tán thành một giải pháp thực hiện hiệu quả được xác định dưới dạng ngôn ngữ mức thấp trên các cơ sở dữ liệu cục bộ. Ngôn ngữ mức cao có thể hiểu là các phép tính quan hệ, các ngôn ngữ mức thấp là sự mở rộng của đại số quan hệ và các thao tác truyền dữ liệu giữa các vị trí dữ liệu.

Tối ưu hoá truy vấn là một vấn đề quan trọng trong việc xử lý truy vấn. Có nhiều phép biến đổi một truy vấn mức cao trên cơ sở dữ liệu phân tán thành nhiều giải pháp thực hiện dưới dạng ngôn ngữ mức thấp, nhưng trong đó chỉ có một giải pháp thực hiện có hiệu quả, tối ưu về chi phí sử dụng tài nguyên của mạng bao gồm chi phí sử dụng bộ nhớ, thời gian xử lý và thời gian truyền dữ liệu.

Chỉ số đánh giá chi phi sử dụng tài nguyên mạng hoặc là tổng thời gian xử lý các thao tác truy vấn tại các vị trí khác nhau và việc truyền dữ liệu giữa các vị trí [Sacco and Yao,1982]. Hoặc đánh giá theo chỉ số theo thời gian đáp ứng của truy vấn [Epstein et al, 1978] là tổng thời gian thực hiện truy vấn. Các thao tác có thể được thực hiện đồng thời song song tại các vị trí khác nhau, vì vậy thời gian đáp ứng có thể nhỏ hơn tổng chi phí.

Trong môi trường cơ sở dữ liệu phân tán, chỉ số đánh giá tối ưu hoá truy vấn có thể dựa vào tổng chi phí giảm thiểu sử dụng bộ nhớ, chi phí thời gian cần thiết khi thực hiện các thao tác vào/ra dữ liệu trong bộ nhớ và chi phí cần thiết để trao đổi dữ liệu giữa các bên tham gia vào trong quá trình xử lý truy vấn. Chi phí truyền thông là một trong các nhân tố quan trọng, được quan tâm trong cơ sở dữ liệu phân tán.

4.2.3 Độ phức tạp của các thao tác đại số quan hệ

Độ phức tạp của các phép toán đại số quan hệ có ảnh hưởng trực tiếp đến thời gian thực hiện các thao tác truy vấn. Độ phức tạp của các phép toán phụ thuộc vào lực lượng của quan hệ (Cardinality), phương pháp phân mảnh, cấp phát dữ liệu cũng như các vấn đề về cấu trúc lưu trữ các mảnh. Độ phức tạp của các thao tác đơn ngôi và hai ngôi theo thứ tự tăng dần, vì thế cũng tăng dần thời gian thực hiện. Độ phức tạp O(n) là tuyến tính đối với các phép toán đơn ngôi, trong đó n là lực lượng của quan hệ, với điều kiện các bộ độc lập với nhau. Độ phức tạp O(n*logn) đối với các phép toán hai ngôi nếu các bộ tồn tại phép so sánh dựa trên giá trị của một thuộc tính nào đó. Độ phức tạp $O(n^2)$ đối với các phép nhân, tích Đề các của hai quan hệ, vì mỗi bộ trong quan hệ này phải tổ hợp với tất cả các bộ trong quan hệ kia.. Độ phức tạp của các thao tác đại số quan hệ có thể được xác định như sau:

Phép toán	Độ phức tạp
Phép chọn (Select) Project (Không loại bỏ trùng lặp)	O(n)
Project (có loại bỏ trùng lặp) Các phép gộp nhóm	O(n*log n)
Phép nối (Join) Phép nửa nối (Semijoin) Phép chia (Division) Các phép tập hợp (Setoperators)	O(n*log n)
Tích Đề các (Cartesian Product)	O(n2)

Hình 4.2 Độ phức tạp của các phép toán quan hệ

4.3 ĐẶC TRƯNG CỦA BỘ XỬ LÝ TRUY VẤN

Khó có thể đánh giá và so sánh độ phức tạp xử lý truy vấn trong cả hai trường hợp cơ sở dữ liệu tập trung [Jarke and Koch, 1984] và phân tán [Sacco and Yao, 1982], [Aper et al, 1983] bởi vì chúng có nhiều điểm khác nhau. Sau đây là một số đặc trưng của bộ xử lý truy vấn. Bốn đặc điểm đầu tiên chung cho cả trường hợp tập trung và trường hợp phân tán. Bốn đặc điểm tiếp theo là cho bộ xử lý truy vấn phân tán.

4.3.1 Ngôn ngữ (Languages)

Nói chung, các thao tác xử lý truy vấn được thực hiện trong hệ quản trị cơ sở dữ liệu quan hệ (DBMS) bằng ngôn ngữ bậc cao, hệ thống có khả năng thực hiện tối ưu hoá câu hỏi truy vấn. Ngôn ngữ xử lý truy vấn dựa trên phép tính đại số quan hệ hay đại số quan hệ, tức là chuyển đổi câu truy vấn dưới dạng phép tính quan hệ thành đại số quan hệ. Trong môi trường phân tán, ngôn ngữ đầu ra là tổ hợp của các phép đại số quan hệ cơ bản được mở rộng thêm các nguyên thuỷ truyền thông. Các phép toán của ngôn ngữ này được cài đặt trực tiếp trong hệ thống. Quá trình xử lý truy vấn phải thực hiện việc ánh xạ một cách hiệu quả từ ngôn ngữ đầu vào đến ngôn ngữ đầu ra.

4.3.2 Các kiểu tối ưu hoá (Types of Optimization)

Tối ưu hoá truy vấn là nhằm mục đích lựa chọn một giải pháp tốt nhất trong toàn bộ các giải pháp có thể thực hiện. Một phương pháp tối ưu hoá truy vấn có hiệu quả là tìm kiếm trong tập các giải pháp và dự đoán chi phí của mỗi giải pháp, lựa chọn giải pháp có chi phí là nhỏ nhất. Tuy nhiên phương pháp này phải chi phí cho chính quá trình tối ưu hoá. Không gian sử dụng thực hiện giải pháp tăng lên khi số các mảnh quan hệ hay số quan hệ tăng lên. Vì vậy phương pháp tìm kiếm vét cạn được sử dụng "Exhaustive" cho với tất cả các giải pháp có thể thực hiện được.

Để tránh những tìm kiếm vét cạn có chi phí cao, có các giải pháp ngẫu nhiên như Iterative Improvement [Swami, 1989] và Simulated Annealing [Ioannidis and Wong, 1987] được đề xuất. Cố gắng tìm kiếm một giải pháp tốt, không nhất thiết là tốt nhất nhưng tránh được chi phí tối ưu hoá quá cao về mặt sử dụng bộ nhớ và thời gian thực hiện.

Một giải pháp làm giảm chi phí tìm kiếm vét cạn là sử dụng các giải thuật Heuristics thu hẹp phạm vi tìm kiếm, chỉ có một số giải pháp được xem xét. Trong môi trường tập trung và phân tán, một giải thuật Heuristics làm giảm kích thước các quan hệ trung gian bằng cách thực hiện các phép toán đơn ngôi trước và sắp xếp thứ tự thực hiến các phép toán hai ngôi theo kích thước tăng dần của các quan hệ trung gian do chúng tạo ra. Một cách đánh giá quan trọng trong các hệ phân tán là thay thế phép kết nối bằng các tổ hợp các nối nửa nhằm giảm thiểu chi phí truyền thông dữ liệu.

3.3.3 Thời điểm tối ưu hoá (Optimization timing)

Một câu truy vấn có thể được tối ưu hoá tại các thời điểm khác nhau liên quan đến thời gian thực hiện truy vấn. Việc tối ưu hoá có thể được thực hiện theo kiểu tĩnh (Statically) trước khi thực hiện truy vấn hoặc theo kiểu động (Dynamically) khi truy vấn được thực hiện. Việc tối ưu hoá truy vấn tĩnh được thực hiện tại thời điểm biên dịch truy vấn. Vì vậy, chi phí của truy vấn có thể được giảm dần qua nhiều lần thực hiện truy vấn, vì đây là các thời điểm thích hợp cho phương pháp tìm kiếm vét cạn (Exhaustive). Kích thước của các quan hệ trung gian không được biết trước, trước khi thực hiện. Vì vậy cần phải được đánh giá ước lượng theo số liệu thống kê cơ sở dữ liệu. Các sai sót trong cách đánh giá này là có thể dẫn đến việc lựa chọn một giải pháp gần tối ưu.

Việc tối ưu hoá động được tiến hành vào thời gian thực hiện truy vấn. Trong các thời điểm thực hiện tối ưu hoá truy vấn, việc lựa chọn thao tác tiếp theo tốt nhất cho tối ưu hoá truy vấn dựa trên những thông tin chính xác về kết quả của các thao tác thực hiện trước đó. Vì vậy, các số liệu thống kê không cần thiết cho việc đánh giá kích thước của các quan hệ trung gian (nhưng có ích trong việc lựa chọn các thao tác đầu tiên).

Ưu điểm của phương pháp động so với phương pháp tĩnh là kích thước thực sự của các quan hệ trung gian là phù hợp cho bộ xử lý truy vấn. Vì vậy sẽ giảm thiểu xác xuất cho việc lựa chọn một giải pháp tồi. Nhược điểm của phương pháp động là các thao tác tối ưu hoá có chi phí cao. Lặp lại nhiều lần cho mỗi thao tác. Vì vậy phương pháp này sẽ rất phù hợp cho một số câu truy vấn đặc biệt.

Các phương pháp tối ưu hoá truy vấn hỗn hợp có các ưu điểm của tối ưu hoá truy vấn tĩnh, tránh được các vấn đề được tạo ra bởi các đánh giá không chính xác gây ra. Về cơ bản

phương pháp này là tĩnh nhưng quá trình truy vấn động có thể ra lúc chạy khi phát hiện có sự khác biệt lớn giữa kích thước dự đoán và kích thước thực tế của các quan hệ trung gian.

4.3.4 Số liệu thống kê (Statistics)

Tính hiệu quả của việc tối ưu hoá truy vấn là dựa trên các số liệu thống kê về cơ sở dữ liệu. Tối ưu hoá truy vấn động cần đến các số liệu thống kê nhằm chọn các thao tác cần phải thực hiện trước tiên. Tối ưu hoá truy vấn tĩnh có nhiều yêu cầu hơn, vì kích thước của các quan hệ trung gian phải được đánh giá dựa trên các số liệu thống kê về cơ sở dữ liệu.

Trong cơ sở dữ liệu phân tán, các số liệu thống kê dành cho việc tối ưu hoá truy vấn có liên quan đến các mảnh, lực lượng và kích thước của mảnh, cũng như kích thước và số lượng các giá trị phân biệt của mỗi thuộc tính. Để giảm thiểu xác xuất lỗi, cần nhiều thông tin chi tiết hơn như các biểu đồ hình cột cho các giá trị thuộc tính... Độ chính xác của số liệu thống kê phụ thuộc vào việc cập nhật theo chu kỳ. Với phương pháp tối ưu hoá tĩnh, các thay đổi lớn về số liệu thống kê được sử dụng để tối ưu hoá truy vấn có thể buộc phải tạo lại quá trình tối ưu hoá truy vấn.

4.3.5 Vị trí quyết định (Decision sites)

Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp tĩnh, có thể sử dụng một vị trí hay nhiều vị trí tham gia vào việc chọn một giải pháp trả lời câu truy vấn. Hầu hết các hệ thống sử dụng phương pháp quyết định tập trung, trong đó có một vị trí đưa ra giải pháp. Quá trình quyết định có thể được phân tán cho nhiều vị trí tham gia vào việc tạo một giải pháp tốt nhất. Phương pháp tập trung đơn giản, nhưng đòi hỏi phải biết toàn bộ về các thông tin cục bộ. Các phương pháp lai, hỗn hợp thì có một vị trí đưa ra quyết định chính và các vị trí khác đưa ra các quyết định cục bộ. Chẳng hạn như System R* sử dụng cách tiếp cận hỗn hợp.

4.3.6 Khai thác cấu hình mạng (Exploitation of Network topology)

Trong các mạng diện rộng WAN, hàm chi phí nhỏ nhất có thể bị giới hạn bởi chi phí truyền thông. Giả thiết này làm đơn giản hoá tối ưu hoá truy vấn phân tán. Vì vậy có thể tách vấn đề này thành 2 vấn đề riêng biệt: lựa chọn giải pháp thực hiện toàn cục dựa trên truyền thông giữa các vị trí và lựa chọn mỗi giải pháp thực hiện cục bộ dựa trên thuật toán xử lý truy vấn tập trung.

Trong các mạng cục bộ LAN, chi phí truyền thông có thể so sánh với chi phí vào/ra. Vì vậy bộ xử lý truy vấn phân tán tăng quá trình thực hiện song song (tất nhiên chi phí truyền thông sẽ nhiều hơn). Khả năng quảng bá của các mạng cục bộ được sử dụng có hiệu quả tối ưu hoá việc xử lý các phép kết nối. Trong môi trường Client/Server, máy khách có thể được khai thác để thực hiện thao tác chuyển đổi cơ sở dữ liệu. Tối ưu hoá truy vấn để ra quyết định một phần của truy vấn sẽ được thực hiện trên máy khách.

4.3.7 Khai thác các mảnh nhân bản (Exploitation of Replicated Fragments)

Một quan hệ phân tán thường được chia thành các mảnh quan hệ. Các câu truy vấn phân tán được mô tả trên các quan hệ toàn cục sẽ được ánh xạ thành các câu truy vấn trên các mảnh vật lý của quan hệ bằng việc chuyển đổi các quan hệ thành các mảnh. Quá trình này được gọi là quá trình cục bộ hoá. Thường người ta nhân bản các mảnh ở nhiều vị trí khác nhau. Phần lớn các thuật toán tối ưu hoá đều xem xét quá trình cục bộ hoá một cách độc lập với quá trình

tối ưu hoá. Một số thuật toán khác tận dụng sự tồn tại của việc nhân bản trong lúc chạy để giảm số lần truyền thông, thuật toán tối ưu hoá sẽ phức tạp hơn.

4.3.8 Sử dụng nửa kết nối (Use of Semijoint)

Thao tác nửa kết nối làm giảm kích thước của các quan hệ. Khi chi phí truyền thông được quan tâm, thì thao tác nửa kết nối đặc biệt sẽ làm giảm dữ liệu cần trao đổi giữa các vị trí. Tuy nhiên, việc sử dụng thao tác nửa kết nối có thể dẫn đến việc tăng số lượng các thông điệp và thời gian xử lý cục bộ. Các thao tác nửa kết nối hiệu quả trong các mạng tốc độ cao, cho phép giảm các thao tác kết nối. Một số thuật toán tối ưu hoá truy vấn tổ hợp tối ưu kết nối với nửa kết nối.

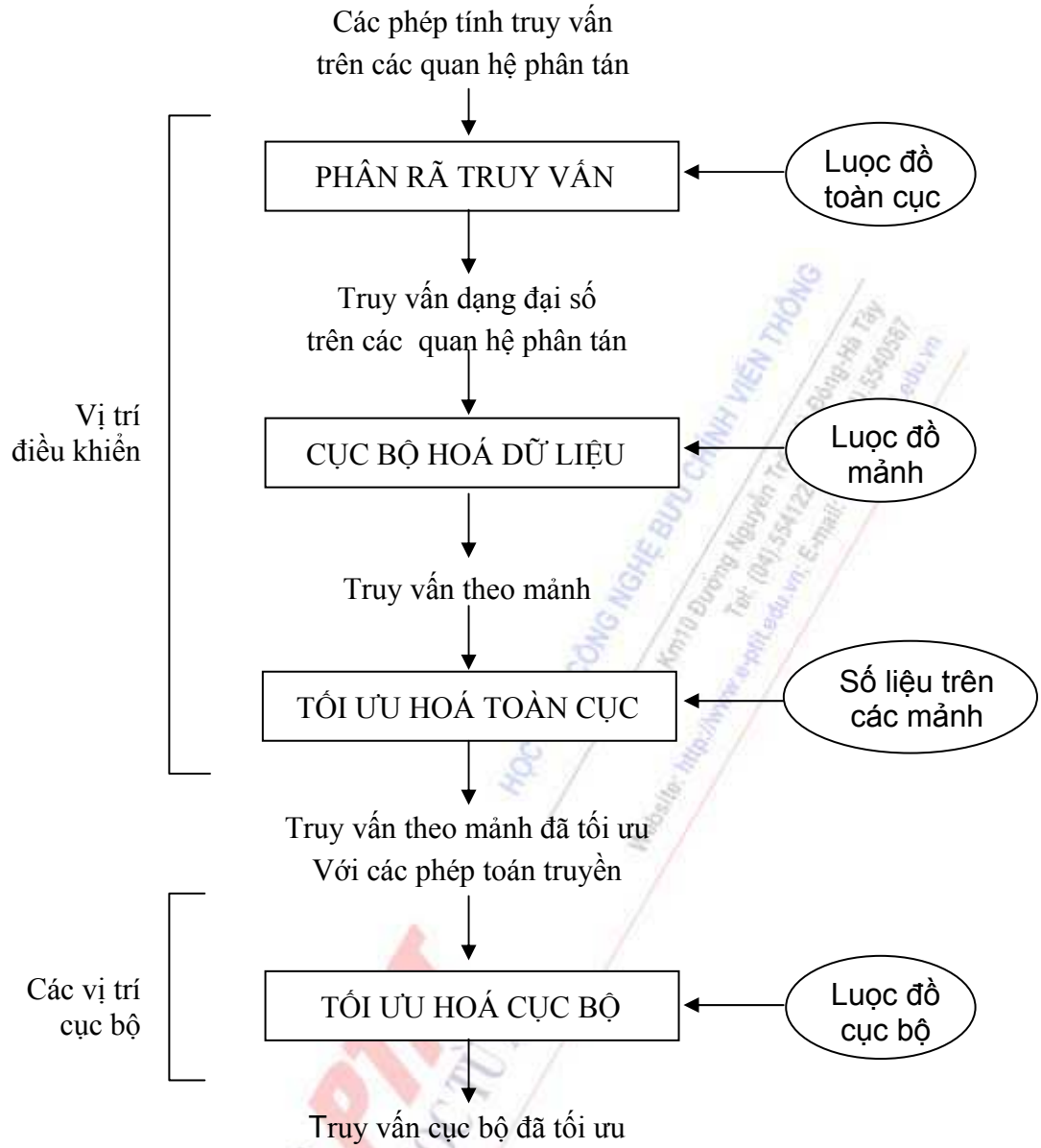
4.4 CÁC TẦNG CỦA QUÁ TRÌNH XỬ LÝ TRUY VẤN

Câu truy vấn phải được phân rã thành một chuỗi các phép toán quan hệ được gọi là truy vấn đại số. Dữ liệu cần truy vấn phải được cục bộ hoá để các thao tác trên các quan hệ được chuyển thành các thao tác trên dữ liệu cục bộ (các mảnh). Cuối cùng câu truy vấn đại số trên các mảnh phải được mở rộng bao gồm các thao tác trao đổi thông tin giữa các vị trí và được tối ưu hoá câu truy vấn. Như vậy có thể mô tả việc xử lý truy vấn trong nhiều vấn đề nhỏ tương ứng với các tầng khác nhau. Lược đồ phân tầng chung của quá trình xử lý truy vấn như Hình 4.3. Đầu vào là một câu truy vấn trên dữ liệu phân tán được xác định dưới dạng các phép tính quan hệ. Câu truy vấn phân tán được đặt trên các quan hệ toàn cục, nghĩa là dữ liệu phân tán được che dấu. Bốn tầng lược đồ ánh xạ truy vấn phân tán thành một chuỗi các thao tác cục bộ được tối ưu hoá, hoạt động trên cơ sở dữ liệu cục bộ.

Chức năng các tầng bao gồm: phân rã truy vấn, tập trung hoá dữ liệu, tối ưu hoá truy vấn toàn cục và tối ưu hoá truy vấn cục bộ.

Phân rã truy vấn và tập trung hoá dữ liệu tương ứng với việc viết lại truy vấn. Chức năng của ba tầng đầu tiên được thực hiện tại một vị trí tập trung và sử dụng các thông tin toàn cục còn chức năng của tầng thứ tư được thực hiện ở vị trí cục bộ.

Phân rã truy vấn là giai đoạn đầu tiên của quá trình xử lý câu truy vấn, thực hiện việc biến đổi câu truy vấn ở dạng ngôn ngữ bậc cao thành câu truy vấn ngôn ngữ bậc thấp thực thi cho kết quả tương đương. đặc trưng của giai đoạn này, khi biến đổi không sử dụng các thông tin về dữ liệu đã được phân tán trên các vị trí.



Hình 4.3 Lược đồ phân tầng tổng quát để xử lý truy vấn phân tán

4.5 PHÂN RÃ TRUY VẤN

Một lược đồ tổng quát để xử lý truy vấn dữ liệu phân tán gồm 4 tầng, trong đó 2 tầng đầu có chức năng phân rã câu truy vấn (Query Decomposition) và cục bộ hoá dữ liệu (Data Location). Các chức năng này được thực hiện liên tiếp, nhằm biến đổi câu truy vấn dạng phép tính quan hệ được đặc tả trên các quan hệ toàn cục thành câu truy vấn dưới dạng đại số quan hệ được định nghĩa trên các mảnh.

Tầng phân rã câu truy vấn có chức năng ánh xạ câu truy vấn phân tán ở dạng phép tính quan hệ thành câu truy vấn đại số trên quan hệ toàn cục. Thông tin cần thiết cho việc biến đổi phân rã truy vấn phân tán được tìm thấy trong mô tả lược đồ khái niệm toàn cục và trong mô tả các quan hệ toàn cục. Thông tin về phân tán các quan hệ không được sử dụngểtong tầng này, nhưng sẽ được sử dụng trong các tầng kế tiếp. Vì vậy các kỹ thuật phân rã được áp dụng trong tầng này là những kỹ thuật của các hệ quản trị cơ sở dữ liệu quan hệ tập trung.

Phân rã câu truy vấn có thể thực hiện theo 4 bước liên tiếp nhau:

- *Bước chuẩn hoá*: Các câu truy vấn bằng các phép tính quan hệ được viết lại dưới dạng chuẩn tắc thích hợp cho những bước tiếp theo. Sự chuẩn hoá một câu truy vấn bao gồm đặt các lượng tử và lượng tử hoá truy vấn bằng cách áp dụng độ ưu tiên các toán tử logic.
- *Bước phân tích:* Câu truy vấn đã chuẩn hoá được phân tích về mặt ngữ nghĩa nhằm loại bỏ các câu vấn tin sai càng sớm càng tốt. Tìm ra truy vấn sai chỉ tồn tại với một tập con các phép tính quan hệ. Thông thường sử dụng một loại đồ thị để nắm bắt ngữ nghĩa của câu truy vấn.
- *Bước loại bỏ dư thừa:* Câu truy vấn đúng được đơn giản hoá bằng cách loại bỏ các phụ thuộc dư thừa. Truy vấn dư thừa chỉ xuất hiện khi các một truy vấn là kết quả của việc biến đổi hệ thống được áp dụng cho truy vấn của người sử dụng.
- *Bước xây dựng lại câu truy vấn:* Câu truy vấn phép tính quan hệ được xây dựng lại dưới dạng truy vấn đại số quan hệ bằng các quy tắc biến đổi.

4.5.1 Bước chuẩn hoá câu truy vấn

Mức độ phức tạp của câu truy vấn đầu vào phụ thuộc vào ngôn ngữ cung cấp các phương tiện. Mục đích của việc chuẩn hoá là biến đổi một câu truy vấn sang một dạng chuẩn để xử lý tiếp. Với các ngôn ngữ quan hệ như SQL, việc chuyển đổi quan trọng nhất là lượng từ hoá truy vấn (Query qualification) (mệnh đề WHERE. Có 2 dạng chuẩn có thể cho việc dự đoán: một là đưa ra mức độ ưu tiên cho phép AND và cái còn lại cho phép OR.

- *Dạng chuẩn hội (Conjunctive Normal Form)* là hội của các tuyển như sau:

  $(p_{11} \vee p_{12} \vee \ldots \vee p_{1n}) \wedge \ldots \wedge (p_{m1} \vee p_{m2} \vee \ldots \vee p_{mn})$

  Trong đó, p_{ij} là một vị từ đơn giản.
- *Ngược lại một lượng từ hoá ở dạng tuyển (Disjunctive Normal Form)*

  $(p_{11} \wedge p_{12} \wedge \ldots \wedge p_{1n} \vee \ldots \vee (p_{m1} \wedge p_{m2} \wedge \ldots \wedge p_{mn})$

Biến đổi các vị từ phi lượng từ bằng cách sử dụng các quy tắc tương đương như sau:

1. $p_1 \wedge p_2 \Leftrightarrow p_2 \wedge p_1$.
2. $p_1 \vee p_2 \Leftrightarrow p_2 \vee p_1$
3. $p_1 \wedge (p_2 \wedge p_3) \Leftrightarrow (p_1 \wedge p_2) \wedge p_3$
4. $p_1 \vee (p_2 \vee p_3) \Leftrightarrow (p_1 \vee p_2) \vee p_3$
5. $p_1 \wedge (p_2 \vee p_3) \Leftrightarrow (p_1 \wedge p_2) \vee (p_1 \wedge p_3)$
6. $p_1 \vee (p_2 \wedge p_3) \Leftrightarrow (p_1 \vee p_2) \wedge (p_1 \vee p_3)$
7. $\neg (p_1 \wedge p_2) \Leftrightarrow \neg p_1 \vee (\neg p_2)$
8. $\neg (p_1 \vee p_2) \Leftrightarrow \neg p_1 \wedge (\neg p_2)$
9. $\neg (\neg p) \Leftrightarrow p$

Ví dụ 4.3: "Tìm tên các nhân viên làm việc trong dự án P1 trong 12 hoặc 24 tháng".

```
SELECT   ENAME
FROM     EMP, ASG
WHERE    EMP.ENO = ASG.ENO
         AND  ASG.PNO = "P1"
         AND   DUR = 12  OR  DUR  = 24
```

Lượng từ hoá dạng chuẩn hội là:

EMP.ENO = ASG.ENO ∧ ASG.PNO = "P_1" ∧ (DUR = 12 ∨ DUR = 24)

Lượng từ hoá dạng chuẩn tuyển là:

(EMP.ENO = ASG.ENO ∧ ASG.PNO = "P_1" ∧ DUR = 12)

∨ (EMP.ENO = ASG.ENO ∧ ASG.PNO = "P_1" ∧ DUR = 24)

4.5.2 Bước phân tích

Bước phân tích câu truy vấn cho phép loại bỏ câu truy vấn đã được chuẩn hoá nhưng sai kiểu hoặc không đúng ngữ nghĩa.

a) Sai kiểu: Một câu truy vấn sai kiểu (Semantically Incorrect) nếu có một thuộc tính trong quan hệ hay tên một quan hệ̂cha được khai báo trong lược đồ toàn cục, hoặc các thao tác áp dụng cho các thuộc tính có kiểu không thích hợp. Tìm các truy vấn có kiểu không đúng tương tự như kiểm tra khai báo kiểu trong các ngôn ngữ lập trình.

Ví dụ 4.4:

```
SELECT   E#
FROM     EMP
WHERE    ENAME > 200.
```

Truy vấn này sai kiểu, vì

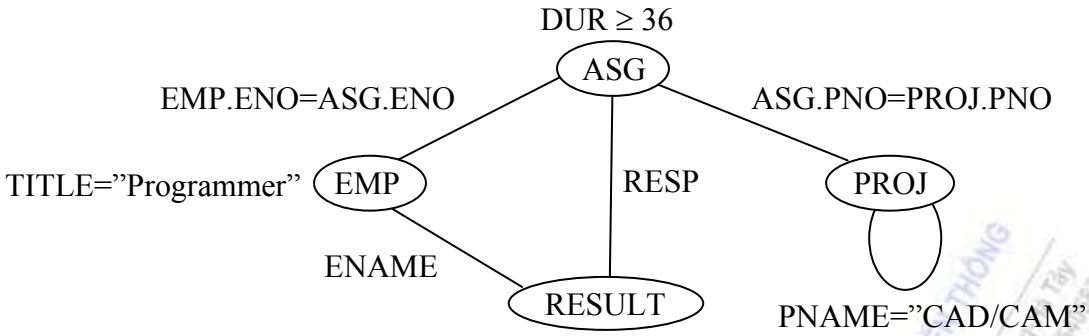
- Thuộc tính E# không phải là thuộc tính của lược đồ quan hệ toàn cục.
- ENAME có kiểu chuỗi (String) không thể so sánh với 200 kiểu số.

b. Sai ngữ nghĩa: Một câu truy vấn được gọi là sai nghĩa (Semantically Incorrect) nếu các thành phần của nó không tham gia vào việc tạo ra kết quả. Để có thể xác định tính đúng đắn ngữ nghĩa câu truy vấn tổng quát, bằng cách biểu diễn đồ thị truy vấn (Query Graph) hay đồ thị kết nối [Ulhman, 1982] gồm các phép chọn, phép chiếu và các phép kết nối, không chứa các phép tuyển và phép phủ định. Trong một đồ thị truy vấn, có một node biểu thị cho một quan hệ kết quả và các node khác biểu thị cho các quan hệ toán hạng. Đường nối giữa 2 node không phải là quan hệ kết quả biểu thị cho phép kết nối. đường nối có node đích là quan hệ kết quả biểu thị cho phép chiếu. Các node quan hệ toán hạng có thể được gán nhãn là một vị từ chọn hoặc vị từ tự kết nối. Đồ thị kết nối nối (Join Graph) là đồ thị con của đồ thị truy vấn, được sử dụng nhiều trong kỹ thuật tối ưu hoá truy vấn.

Ví dụ 4.5: "Tên và nhiệm vụ của các lập trình viên đã làm việc cho dự án CAD/CAM hơn 3 năm"

```
SELECT    ENAME, RESP
FROM      EMP, ASG, PROJ
WHERE     EMP.ENO = ASG.ENO
          AND   ASG.PNO = PROJ.PNO
          AND   PNAME = "CAD/CAM"
          AND   DUR ≥ 36
          AND   TITLE = "Programmer"
```

Đồ thị truy vấn như sau:



Hình 4.4: Ví dụ đồ thị truy vấn

Đồ thị kết nối tương ứng::

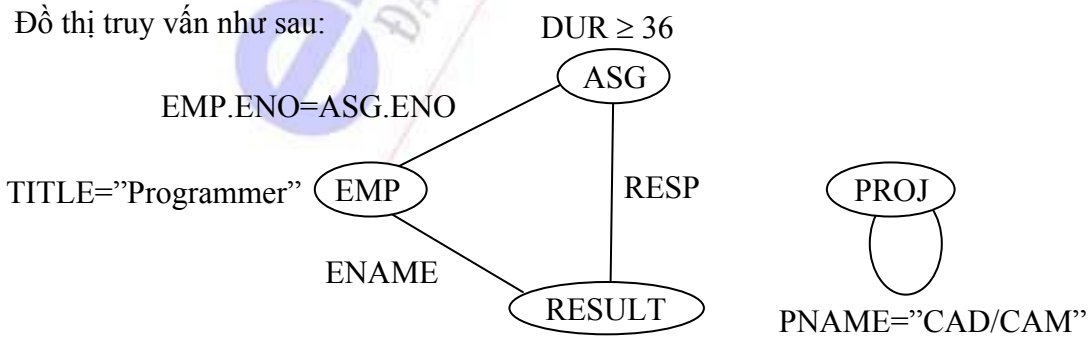
Hình 4.5: Đồ thị kết nối tương ứng

Đồ thị truy vấn sử dụng nhằm xác định tính đúng đắn về mặt ngữ nghĩa của truy vấn hội đa biến không có phủ định (¬). Một câu truy vấn không đúng về mặt ngữ nghĩa nếu đồ thị truy vấn của nó không liên thông. Nghĩa là có ít nhất một đồ thị con, tương ứng câu với truy vấn con, tách ra khỏi đồ thị truy vấn có chứa quan hệ kết quả. Thường khi có các vị từ kết nối bị thiếu, câu truy vấn đó cần được loại bỏ.

Ví dụ 4.6: Xét câu truy vấn sau:

```
SELECT    ENAME, RESP
FROM      EMP, ASG, PROJ
WHERE     EMP.ENO = ASG.ENO
          AND  PNAME = “CAD/CAM”
          AND  DUR ≥ 36
          AND  TITLE = “PROGRAMMER”
```

Đồ thị truy vấn như sau:



Hình 4.6: Đồ thị truy vấn không liên thông

Đồ thị truy vấn Hình 4.6 không liên thông, có 2 đồ thị tách biệt nhau. Điều này có nghĩa là câu truy vấn này sai nghĩa. Có 3 giải pháp khắc phục (1) loại bỏ câu vấn tin, (2) giả thiết một tích Đề các giữa quan hệ ASG và PROJ, hoặc (3) có thể đoán nhận vị từ kết nối bị thiếu ASG.PNO = PROJ.PNO.

4.5.3 Bước loại bỏ dư thừa

Trong một câu truy vấn, có thể loại bỏ các vị từ dư thừa và các thao tác dư thừa bằng cách giản ước các lượng từ hoá bằng các quy tắc luỹ đẳng sau đây:

1. $p \wedge p \Leftrightarrow p$
2. $p \vee p \Leftrightarrow p$
3. $p \wedge true \Leftrightarrow p$
4. $p \vee false \Leftrightarrow p$
5. $p \wedge false \Leftrightarrow false$
6. $p \vee true \Leftrightarrow true$
7. $p \wedge \neg p \Leftrightarrow false$
8. $p \vee \neg p \Leftrightarrow true$
9. $p_1 \wedge (p_1 \vee p_2) \Leftrightarrow p_1$
10. $p_1 \vee (p_1 \wedge p_2) \Leftrightarrow p_1$

Ví dụ 4.7: Xét câu truy vấn sau:

```
SELECT      TITLE
FROM        EMP
WHERE       (NOT  (TITLE = “Programmer”)
            AND   (TITLE = “Programmer”
            OR     TITLE = “Elect.Eng” )
            AND   NOT  (TITLE = “Elect.Eng” ))
            OR     ENAME = :J. Doe”
```

Đặt p_1 là TITLE = “Programmer”
P_2 là TITLE = “Elect.Eng”
P_3 là TENAME = :J. Doe”

Lượng từ hoá câu truy vấn trên được là:

Áp dụng quy tắc: $p \wedge (s \vee q) \Leftrightarrow (p \wedge s) \vee (p \wedge q)$ và $p \wedge (s \wedge q) \Leftrightarrow (p \wedge s) \wedge q$

khi đó: $(\neg p_1 \wedge (p_1 \vee p_2) \wedge \neg p_2) \vee p_3 \Leftrightarrow ((\neg p_1 \wedge p_1) \vee (\neg p_1 \wedge p_2) \wedge \neg p_2) \vee p_3$

$\Leftrightarrow ((\neg p_1 \wedge (p_2 \wedge \neg p_2)) \vee p_3 \Leftrightarrow \neg p_1 \wedge false \vee p_3 \Leftrightarrow p_3$

Tức là:

```
SELECT      TITLE
FROM        EMP
WHERE       ENAME = :J. Doe”
```

4.5.4 Bước viết lại truy vấn

Bước cuối cùng của việc phân rã truy vấn là viết lại truy vấn dưới dạng đại số quan hệ. Bước này lại chia thành các bước nhỏ sau:

3. Chuyển đổi câu truy vấn từ phép tính quan hệ sang đại số quan hệ.
4. Xây dựng lại truy vấn đại số quan hệ để cải thiện khả năng thực hiện.

Có thể biểu diễn câu truy vấn đại số quan hệ bằng một cây đại số quan hệ. Các node lá của nó biểu thị cho một quan hệ được lưu trữ trong cơ sở dữ liệu và các node không phải là node lá biểu thị cho một quan hệ trung gian sinh ra bởi các phép toán đại số quan hệ. Chuỗi các thao tác từ các node lá đến các node gốc biểu diễn cho kết quả câu truy vấn.

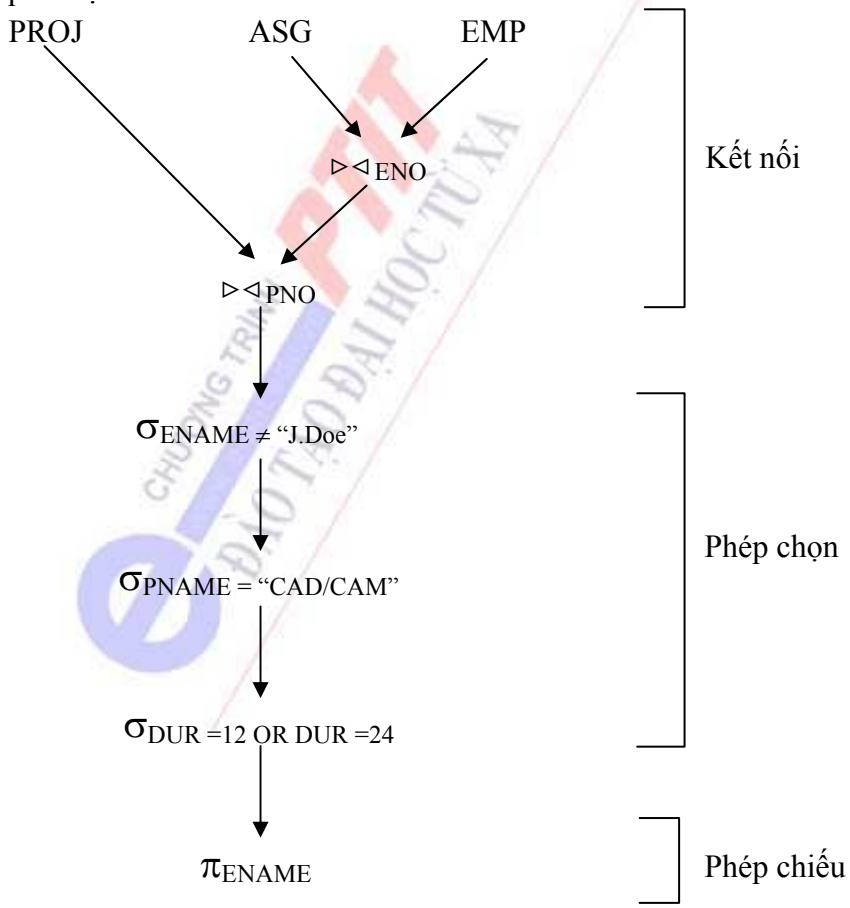
Biến đổi một câu truy vấn phép tính quan hệ thành một cây đại số quan hệ, thực hiện như sau: các node lá tương ứng với các quan hệ có sau mệnh đề FROM. Node gốc được tạo ra như một phép chiếu chứa các thuộc tính kết quả. Các thuộc tính này nằm sau mệnh đề SELECT của câu truy vấn. Lượng từ hóa sau mệnh đề WHERE được biểu thị bằng một chuỗi các phép toán chọn, kết nối, phép hợp...đi từ các node lá đến node gốc. Chuỗi này có thể được cho trực tiếp qua thứ tự xuất hiện của các vị từ và các toán tử.

Ví dụ 4.8: Xét câu truy vấn sau: " Xác định tên các nhân viên trừ J. Doe đã làm cho dự án CAD/CAM trong một hoặc hai năm"

Câu lệnh SQL như sau:

```
SELECT    ENAME
FROM      EMP, ASG, PROJ
WHERE     EMP.ENO = ASG.ENO
          AND   PROJ.PNO = ASG.PNO
          AND   ENAME ≠ "J.Doe"
          AND   PROJ.NAME = "CAD/CAM"
          AND   (DUR =12  OR  DUR =24)
```

Cây đại số quan hệ như sau:



Hình 4.7: Ví dụ về cây đại số quan hệ

Các quy tắc biến đổi cây đại số quan hệ :

Cho các quan hệ R(A), A = {A_1, A_2, .., A_n }, S(B), B = {B_1, B_2,, ... , B_n } và T. Có thể nhận được các cây đại số quan hệ tương đương, áp dụng các qui tắc biến đổi (Transformation Rule) sau:

1. *Tính chất giao hoán của các phép toán hai ngôi:*
   - Tích Đề các: R x S $\Leftrightarrow$ S x R
   - Kết nối : R$\triangleright\triangleleft$ S $\Leftrightarrow$ S$\triangleright\triangleleft$ R.
   - Hợp của hai quan hệ: R $\cup$ S $\Leftrightarrow$ S $\cup$ R
   - Qui tắc này không được áp dụng cho hiệu và kết nối nửa.
2. *Tính kết hợp của các phép toán hai ngôi.*
   - Tích Đề các: (R x S) x T $\Leftrightarrow$ R x (S x T)
   - Kết nối : R$\triangleright\triangleleft$ (S $\triangleright\triangleleft$ T) $\Leftrightarrow$ (R $\triangleright\triangleleft$ R) $\triangleright\triangleleft$ T
3. *Tính lũy đẳng của các phép toán đơn ngôi*:
   - Các phép chiếu liên tiếp trên cùng một quan hệ có thể nhóm lại với nhau. Ngược lại một phép chiếu trên nhiều thuộc tính có thể tách ra nhiều phép chiếu liên tiếp nhau. Nếu A' $\subseteq$ A và A'' $\subseteq$ A', khi đó $\pi_{A'}(\pi_{A''}(R)) \Leftrightarrow \pi_{A'}(R)$
   - Các phép chọn liên tiếp $\sigma_{pi(Ai)}$ trên cùng một quan hệ, trong đó p_i là một vị từ được áp dụng cho thuộc tính A_i có thể được nhóm lại. Ngược lại một phép chọn qua một hội các vị từ có thể được tách ra thành nhiều phép chọn liên tiếp.

     $$\sigma_{p1(A1)}(\sigma_{p2(A2)}(R)) \Leftrightarrow \sigma_{p1(A1) \wedge \sigma p2(A2)}(R)$$
4. *Giao hoán phép chọn với phép chiếu:*

   Phép chọn và chiếu trên cùng một quan hệ có thể được giao hoán như sau:

   $$\pi_{A1,...,An}(\sigma_{p(Ap)}(R)) \Leftrightarrow \pi_{A1,...,An}(\sigma_{p(Ap)}(\pi_{A1,...,An,Ap}(R)))$$
5. *Giao hoán phép chọn với các phép toán hai ngôi:*
   - Phép chọn và tích Đề các: $\sigma_{p(Ai)}(R \times S) \Leftrightarrow (\sigma_{p(Ai)}(R)) \times S$
   - Phép chọn và kết nối: $\sigma_{p(Ai)}(R \triangleright\triangleleft_{p(Aj,Bk)} S) \Leftrightarrow \sigma_{p(Ai)}(R) \triangleright\triangleleft_{p(Aj,Bk)} S$
   - Phép chọn và hợp: $\sigma_{p(Ai)}(R \cup T) \Leftrightarrow \sigma_{p(Ai)}(R) \cup \sigma_{p(Ai)}(T)$
6. *Giao hoán phép chiếu với phép toán hai ngôi.*
   - Phép chiếu và tích Đề Các: Nếu C = A´ $\cup$ B´ trong đó A´$\subseteq$ A và B´ $\subseteq$ B, thì $\pi_C(R \times S) \Leftrightarrow \pi_{A'}(R) \times \pi_{B'}(S)$
   - Phép chiếu và kết nối: $\pi_C(R \triangleright\triangleleft_{,Bk)} S) \Leftrightarrow \pi_{A'}(R) \triangleright\triangleleft_{Aj,Bk)} \pi_{B'}(S)$

     Trong đó $A_i \in A'$, $B_k \in B'$. Vì C = A´ $\cup$ B´ suy ra $A_i, B_k \in C$.
   - Phép chiếu và hợp $\pi_C(R \cup S) \Leftrightarrow \pi_C(R) \cup \pi_C(S)$.
   - Phép chiếu và hiệu :$\pi_C(R - S) \Leftrightarrow \pi_C(R) - \pi_C(S)$.

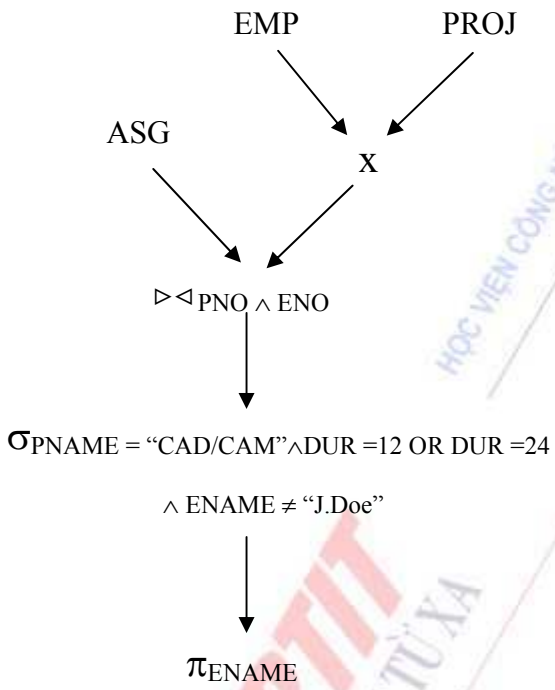
Quy tắc 6 được sử dụng nhiều trong kỹ thuật tối ưu hoá truy vấn.

Áp dụng sáu quy tắc trên có thể tạo ra nhiều cây quan hệ tương đương. Cây trong Hình 4.7 tương đương với cây trong Hình 4.8. Cây trong Hình 4.8 thực hiện phép tích Đề Các với các quan hệ EMP và PROJ, điều này làm cho chi phí thực hiện cao hơn so với cây 3.7. Tuy nhiên từ một cây quan hệ ban đầu chẳng hạn như cây 3.7 có thể biến đổi thành nhiều cây quan hệ

tương đương , trong đó có nhiều cây có cấu trúc "xấu" và cũng có nhiều cây có cấu trúc tốt. Những cây có cấu trúc xấu như cây trong Hình 4.8 cần phải loại bỏ.

Để tìm ra một cây đại số quan hệ "tốt hơn", có thể sử dụng quy tắc trên bằng bốn cách:

- Tách các phép toán đơn ngôi, làm đơn giản biểu thức truy vấn.
- Các phép toán đơn ngôi trên cùng một quan hệ có thể nhóm lại thành một phép toán truy xuất trên quan hệ một lần.
- Các phép toán đơn ngôi có thể giao hoán với các phép toán hai ngôi sao cho một số phép toán như chiếu, chọn được thực hiện trước.
- Các phép toán hai ngôi có thể được sắp xếp lại.



Hình 4.8: Cây đại số quan hệ tương đương

Viết lại câu truy vấn nghĩa là tìm một cây có cấu trúc tốt hơn, có chất lượng hơn so với cây gốc. Cây đại số quan hệ trong hình 4.9 được tái tạo từ cây đại số quan hệ trong hình 4.7. Nó được xem là có cấu trúc đạt chất lượng theo nghĩa là nó tránh truy xuất nhiều lần đến cùng một quan hệ. Các phép chọn $\sigma_{ENAME \neq "J.Doe"}(EMP)$, $\sigma_{DUR =12 \text{ OR } DUR =24}(ASG)$ và $\sigma_{PNAME = "CAD/CAM"}(PROJ)$ được thực hiện sớm nhất làm giảm đáng kể kích thước của các quan hệ trung gian. Tiếp theo phép chọn là các phép chiếu được thực hiện:

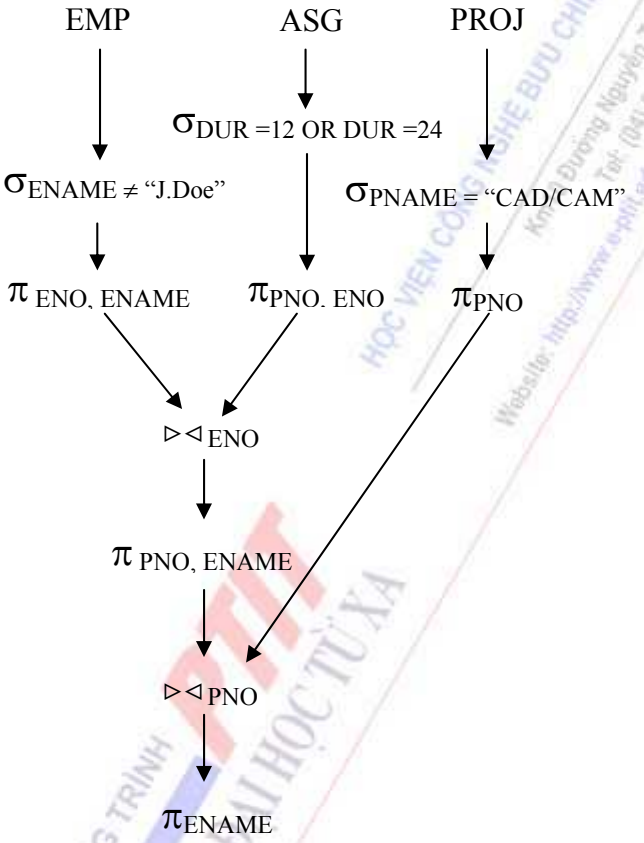
$\pi_{ENO, ENAME}(\sigma_{ENAME \neq "J.Doe"}(EMP))$, $\pi_{PNO, ENO}(\sigma_{DUR =12 \text{ OR } DUR =24}(ASG))$ và $\pi_{PNO}(\sigma_{PNAME = "CAD/CAM"}(PROJ))$. Tiếp theo là các phép kết nối với chi phí thấp. Tuy nhiên, cây đại số quan hệ hình 4.9 còn xa mới đạt được tối ưu. Vì phép chọn trên EMP với vị từ ENAME = "J. Doe" sẽ không hiệu quả, kích thước quan hệ kết quả giảm không đáng kể.

Trên đây là bốn giai đoạn kế liên tiếp trong quá trình phân rã câu truy vấn, bao gồm:

- Chuẩn hoá
- Phân tích.

- Loại bỏ dư thừa
- Viết lại truy vấn

nhằm ánh xạ một câu truy vấn dạng phép tính quan hệ thành câu truy vấn đại số quan hệ. Có thể có nhiều câu truy vấn đại số tương đương với cùng một câu truy vấn ban đầu. Để phân rã có hiệu quả, các biểu thức truy vấn gốc sẽ được tái cấu trúc bằng một số qui tắc biến đổi và các Heuristic. Các qui tắc cho phép tách các phép toán đơn ngôi, nhóm các phép toán đơn ngôi trên cùng một quan hệ, hoán vị các phép toán đơn ngôi với hai ngô và hoán vị các phép toán hai ngôi. Thí dụ về các heuristic là đẩy các phép chọn xuống và thực hiện các phép chiếu càng sớm càng tốt...



Hình 4.9: Cây đại số quan hệ đã được viết lại

4.6 CỤC BỘ HÓA DỮ LIỆU PHÂN TÁN

Đầu vào của tầng hai là một truy vấn đại số trên quan hệ phân tán. Chức năng chủ yếu của tầng này là cục bộ hoá dữ liệu truy vấn sử dụng các thông tin dữ liệu phân tán, nghĩa là tầng cục bộ hoá dữ liệu chịu trách nhiệm dịch câu truy vấn đại số quan hệ trên quan hệ toàn cục sang câu truy vấn đại số quan hệ trên các mảnh vật lý có sử dụng các thông tin được lưu trữ trong lược đồ phân mảnh. Nó xác định mảnh quan hệ nào sẽ được sử dụng trong truy vấn và chuyển đổi câu truy vấn phân tán thành một truy vấn trên mảnh cụ thể. Để tạo một truy vấn trên mảnh được thực hiện bởi hai bước:

Bước 1: Truy vấn phân tán được ánh xạ sang một truy vấn trên mảnh bằng việc thay thế mỗi quan hệ phân tán bằng chương trình xây dựng lại có chứa các phép toán đại số quan hệ thao tác trên mảnh, gọi là chương trình cục bộ hoá (Localization Program)
Bước 2: Truy vấn trên mảnh được đơn giản hoá và xây dựng lại để tạo ra một truy vấn khác tốt hơn. Quá trình đơn giản hoá và xây dựng lại có thể được thực hiện dựa theo cùng một quy tắc được sử dụng trong tầng phân rã.

Phân mảnh được định nghĩa bằng các quy tắc phân mảnh. Các mảnh được biểu diễn dưới dạng quan hệ. Một quan hệ toàn cục có thể được xây dựng lại bằng cách áp dụng các quy tắc phân mảnh đảo và dẫn xuất một chương trình cục bộ hoá mà các toán hạng quan hệ là các mảnh. Để đơn giản, giả thiết không xét các trường hợp các mảnh nhân bản.

4.6.1 Rút gọn cho phân mảnh ngang nguyên thuỷ

Phân mảnh ngang phân tán một quan hệ dựa trên các vị từ chọn (Select Predicate). Ví dụ quan hệ EMP(ENO, ENAME, TITLE) có thể được phân mảnh ngang thành:

$EMP_1 = б_{ENO \leq \text{“E3”}}(EMP)$
$EMP_2 = б_{\text{“E3”} < ENO < \text{“E6”}}(EMP)$
$EMP_3 = б_{ENO > \text{“E6”}}(EMP).$

Khi đó chương trình cục bộ hoá cho quan hệ phân mảnh ngang là hợp các mảnh:

$$EMP = EMP_1 \cup EMP_2 \cup EMP_3$$

Vì vậy dạng truy vấn gốc được xác định trên EMP sẽ thu được bằng cách thay EMP bởi ($EMP_1 \cup EMP_2 \cup EMP_3$). Như vậy, để giảm các thao tác truy vấn trên quan hệ đã được phân mảnh theo chiều ngang, trước hết phải xác định rõ cần thao tác trên mảnh nào và sau đó xây dựng lại cây con, xem xét loại bỏ các quan hệ rỗng. Phân mảnh ngang sẽ được sử dụng để làm đơn giản hoá các phép chọn và phép kết nối.

a. Rút gọn phép chọn: Phép chọn thực hiện trên các mảnh có lượng từ hoá mâu thuẫn với lượng từ hoá của quy tắc phân mảnh sinh ra các quan hệ rỗng. Cho một quan hệ R được phân mảnh theo chiều ngang là $R_j = \sigma_{pj}(R)$, j=1..n, quy tắc này có thể được biểu diễn một cách hình thức như sau:

Quy tắc 1: $\sigma_{pi}(R_j) = \varnothing$ nếu $\forall x \in R: \neg (pi(x) \wedge pj(x))$,

trong đó p_i, p_j là các vị từ chọn, x là một bộ.

Ví dụ vị từ ENO = “E1” mâu thuẫn với các vị từ của mảnh EMP_2 và EMP_3, nghĩa là không có bộ nào thoả vị từ ENO = “E1”.

Ví dụ 4.9: Xét câu truy vấn sau:

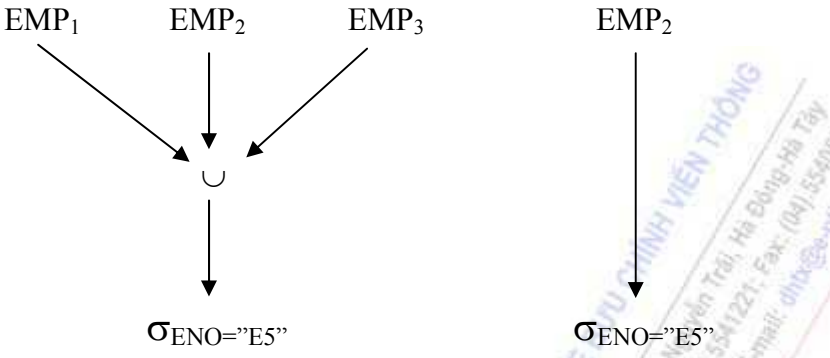
```
SELECT      *
FROM        EMP
WHERE       ENO = :E5”
```

Áp dụng cách tiếp cận thô sơ để cục bộ hoá EMP từ EMP_1, EMP_2 và EMP_3, cho câu truy vấn gốc Hình 4.10[a] bằng cách hoán vị phép chọn và phép hợp. Dễ dàng nhận thấy vị từ chọn mâu thuẫn với EMP_1 và EMP_3. Câu truy vấn đã rút gọn chỉ ứng dụng có một mảnh EMP_2, như trong Hình 4.10b.

$\sigma_{ENO = :E5"} (EMP_1 \cup EMP_2 \cup EMP_3)$

$= \sigma_{ENO = :E5"} (EMP_1) \cup \sigma_{ENO = :E5"} (EMP_2) \cup \sigma_{ENO = :E5"} (EMP_3)$

$= \sigma_{ENO = :E5"} (EMP_2)$



(a) Câu truy vấn gốc (b) Câu truy vấn đã rút gọn

Hình 4.10: Rút gọn phân mảnh ngang với phép chọn

b. Rút gọn với phép kết nối: Phép kết nối thực hiện trên các phân mảnh ngang có thể đơn giản khi kết nối dựa theo các thuộc tính kết nối., bằng cách phân phối kết nối trên các hợp và sau đó loại bỏ các kết nối không tác dụng. Việc phân phối các kết nối trên phép hợp được phát biểu như sau:

$$(R_1 \cup R_2) \triangleright\triangleleft S = (R_1 \triangleright\triangleleft S) \cup (R_2 \triangleright\triangleleft S)$$

Trong đó, R_i là các mảnh của R và S là một quan hệ.

Bằng phép biến đổi này, các phép hợp có thể được di chuyển xuống dưới trong cây đại số quan hệ, nên tất cả các phép kết nối có thể của các mảnh đều được lộ ra. Các phép kết nối vô dụng được xác định khi lượng từ hoá các mảnh có mâu thuẫn. Giả sử các mảnh R_i và R_j được xác định theo các vị từ chọn p_i và p_j trên cùng một quan hệ, quy tắc đơn giản hoá có thể được phát biểu như sau

Quy tắc 2: $R_i \triangleright\triangleleft R_j = \varnothing$ nếu $\forall x \in R_i , \forall y \in R_j : \neg (p_i(x) \wedge p_j(y))$

Việc xác định các kết nối vô dụng có thể được thực hiện bằng cách chỉ xét các vị từ của các mảnh Áp dụng quy tắc này cho phép kết nối hai quan hệ được cài đặt như các nối từng phần song song các mảnh. Câu truy vấn rút gọn chưa chắc đã đơn giản hơn câu truy vấn gốc, nếu như trong câu truy vấn gốc có rất nhiều nối từng phần, vif rất ít các vị từ phân mảnh mâu thuẫn. Như vậy câu truy vấn rút gọn sẽ tốt hơn khi có ít các nối từng phần. Trường hợp xấu nhất xảy ra khi mỗi mảnh của quan hệ này được kết nối với mỗi mảnh khác của quan hệ khác, nghĩa là thực hiện phép tích Đề Các hai quan hệ. Ưu điểm của truy vấn rút gọn theo phép kết nối là các nối từng phần có thể được thực hiện song song và vì vậy nó sẽ làm giảm thời gian đáp ứng.

Ví dụ 410: Giả sử quan hệ EMP(ENO, ENAME, TITLE) được phân mảnh ngang thành EMP_1, EMP_2 và EMP_3.

$EMP_1 = \sigma_{ENO \leq "E3"}(EMP)$

$EMP_2 = \sigma_{"E3" < ENO < "E6"}(EMP)$

$EMP_3 = \sigma_{ENO>"E6"}(EMP)$.

Quan hệ ASG(ENO, PNO, RESP, DUR) được phân mảnh như sau:

$ASG_1 = \sigma_{ENO\leq "E3"}(ASG)$

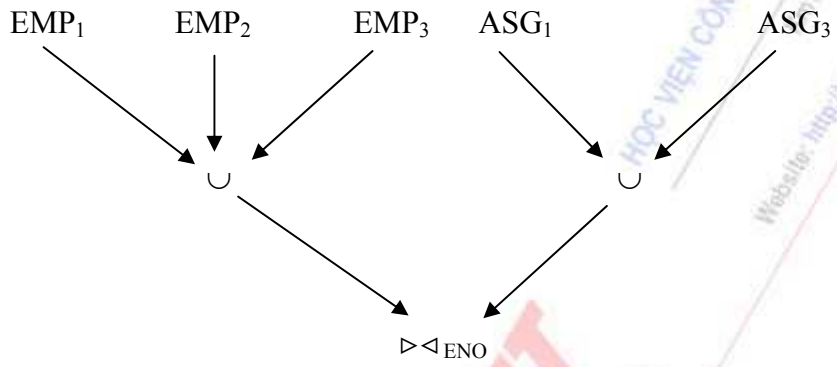
$ASG_2 = \sigma_{ENO > "E3"}(ASG)$

EMP_1 và ASG_1 cùng định nghĩa bởi vị từ ENO≤ "E3" . Vị từ định nghĩa trong ASG_2 là hợp của các vị từ được định nghĩa trong EMP_2 và EMP_3:

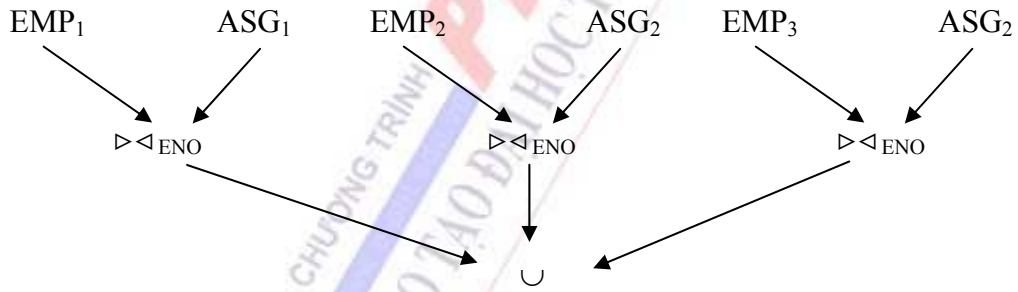
ENO > "E3" = ("E3" < ENO < "E6") ∪ (ENO>"E6"). Xét câu truy vấn sau:

```
SELECT      *
FROM        EMP, ASG
WHERE       EMP.ENO = ASG.ENO
```

Câu truy vấn được rút gọn bằng cách phân phối các nối trên các hợp và áp dụng quy tắc 2 có thể cài đặt như hợp của ba nối từng phần được thực hiện song song.



Hình 4.11a Cây đại số quan hệ truy vấn gốc



Hình 4.11b Rút gọn phân mảnh ngang với phép kết nối

4.6.2 Rút gọn cho phân mảnh dọc

Phân mảnh dọc phân tán một quan hệ dựa trên các thuộc tính chiếu. Vì vậy phép kết nối sẽ là phép toán tái xây dựng các phân mảnh dọc, Chương trình cục bộ hoá cho quan hệ phân mảnh dọc bao gồm các kết nối của các mảnh trên các thuộc tính chung.

Ví dụ 4.11: Giả sử quan hệ EMP(ENO, ENAME, TITLE) được phân mảnh như sau:

Phân mảnh ngang phân tán một quan hệ dựa trên các vị từ chọn (Select Predicate). Ví dụ quan hệ EMP(ENO, ENAME, TITLE) có thể được phân mảnh ngang thành:

$EMP_1 = \pi_{ENO, ENAME}(EMP)$

$EMP_2 = \pi_{ENO, TITLE}(EMP)$

Khi đó chương trình cục bộ hoá cho quan hệ phân mảnh dọclà:

$EMP = EMP_1 \rhd\lhd_{ENO} EMP_2$

Cũng như phân mảnh ngang, các câu truy vấn trên các mảnh dọc được rút gọn bằng cách xác định các quan hệ trung gian vô dụng và loại bỏ các cây con đã sinh ra chúng. Phép chiếu trên một mảnh dọc không có thuộc tính chung với các thuộc tính chiếu sinh ra các quan hệ vô dụng có thể không rỗng. Cho một quan hệ R định nghĩa trên tập các thuộc tính $A = \{A_1, A_2,...,A_n\}$ và được phân thành $R_i = \pi_{A'}(R)$, i=1..k, $A' \subseteq A$. Quy tắc được phát biểu một cách hình thức như sau:

Quy tắc 3: $\pi_{D, K}(R_i)$ là vô dụng nếu tập các thuộc tính chiếu D không nằm trong A'.

Ví dụ 4.12: Giả sử

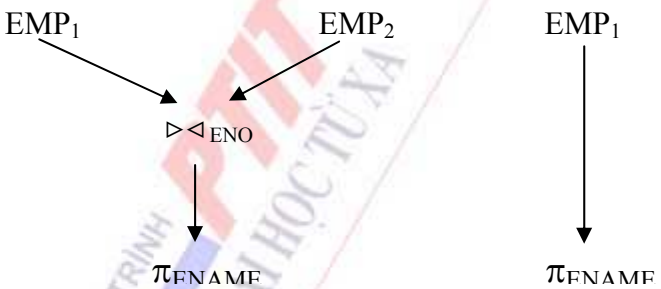
$EMP_1 = \pi_{ENO, ENAME}(EMP)$

$EMP_2 = \pi_{ENO, TITLE}(EMP)$

Xét câu truy vấn SQL như sau

```
SELECT      ENAME
FROM        EMP
```

Bằng cách hoán vị phép chiếu và phép kết nối , nghĩa là thực hiện phép chiếu trên các thuộc tính ENO và ENAME , khi đó có thể nhận thấy rằng phép chiếu trên thuộc tính ENAME trên quan hệ EMP_2 là vô dụng, vì ENAME không phải là thuộc tính của EMP_2. Vì vậy phép chiếu chỉ cần thực hiện trên EMP_1.



Hình 4.12: Rút gọn phân mảnh dọc

4.6.3 Rút gọn cho phân mảnh dẫn xuất

Phép kết nối thường xuyên xảy ra và có chi phí cao. Tối ưu hoá bằng cách sử dụng các phân mảnh ngang nguyên thuỷ khi các quan hệ nối được phân mảnh theo các thuộc tính nối. Trong trường hợp này nối của hai quan hệ được cài đặt như hợp của các nối từng phần. Tuy nhiên phương pháp này ngăn cản không cho một trong các quan hệ phân mảnh theo một phép chọn trên một thuộc tính khác. Phân mảnh ngang dẫn xuất phân phối hai quan hệ, cải thiện khả năng xử lý các điểm giao nhau giữa các phép chọn và phép kết nối. Nếu quan hệ R phân mảnh dẫn xuất theo quan hệ S, các mảnh của R và S có giá trị như nhau ở thuộc tính kết nối sẽ nằm cùng vị trí. Quan hệ S có thể phân mảnh theo một vị từ chọn.

Vì các bộ của quan hệ R được đặt tuỳ chọn theo các bộ của S. để cho đơn giản, giả sử chỉ xét phân mảnh dẫn xuất chỉ được sử dụng cho mối liên hệ một - nhiều, trong đó một bộ của S tương ứng với n bộ của R và một bộ của R chỉ khớp đúng với một bộ của S.

Ví dụ 4.13: Cho mối quan hệ một - nhiều EMP → ASG. Giả sử ASG được phân mảnh gián tiếp theo các quy tắc sau:

$ASG_1 = ASG \triangleright\prec_{ENO} EMP_1$

$ASG_2 = ASG \triangleright\prec_{ENO} EMP_2$

Trong đó $EMP_1 = \sigma_{TITLE = \text{“Programmer”}}(EMP)$

$EMP_2 = \sigma_{TITLE \neq \text{“Programmer”}}(EMP)$

Chương trình cục bộ hoá cho quan hệ phân mảnh ngang là

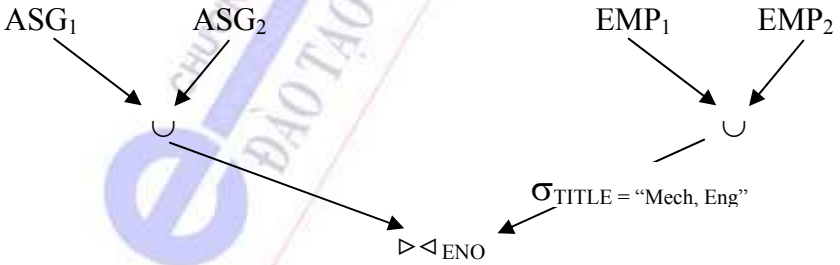
$ASG = ASG_1 \cup ASG_2$

Các câu truy vấn trên các mảnh dẫn xuất có thể được rút gọn bằng cách phân phối các nối trên các phép hợp và áp dụng quy tắc 2. Vì quy tắc phân mảnh chỉ rõ các bộ sẽ khớp với nhau, một số nối sinh ra quan hệ rỗng, các vị từ phân mảnh có mâu thuẫn. Chẳng hạn các vị từ của ASG_1 và EMP_2 có mâu thuẫn, vì vậy $ASG_1 \triangleright\triangleleft EMP_2 = \varnothing$

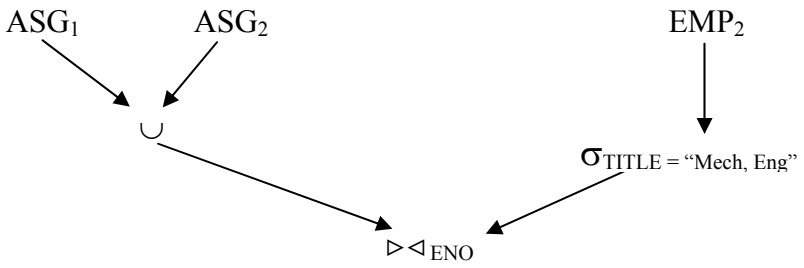
Xét câu truy vấn sau

```
SELECT    *
FROM      EMP, ASG
WHERE     EMP.ENO = ASG.ENO
          AND TITLE = “Mech, Eng”
```

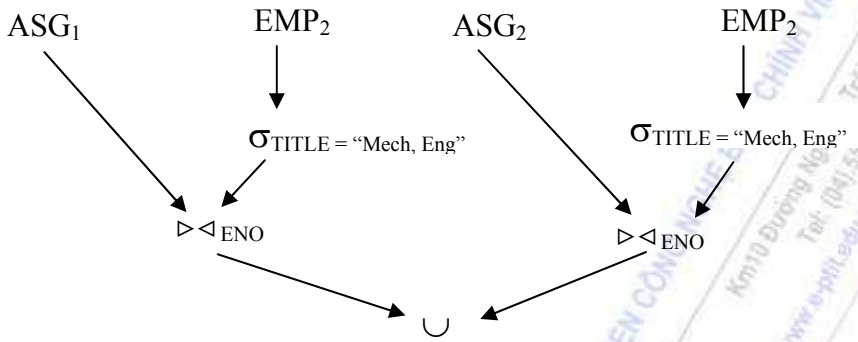
Câu truy vấn gốc được thao tác trên các mảnh EMP_1, EMP_2, ASG_1 và ASG_2. (Hình 4.13a). Thực hiện phép chọn trên các mảnh EMP_1, EMP_2 , vì vị từ chọn mâu thuẫn trên mảnh EMP_1, nên kết quả câu truy vấn rút gọn thu được như trong Hình 4.13b. Nhằm xác định các vị từ kết nối mâu thuẫn, cần phải phân phối các nối trên các hợp. Kết quả là cây Hình 4.13c. Cây con bên trái nối hai mảnh ASG1 và EMP2 với các lượng từ hoá mâu thuẫn bởi các vị từ chọn TITLE = “Programmer: trong ASG_1 và TITLE ≠ “Programmer” trong EMP_2 . Vì vậy có thể loại bỏ cây bên trái và thu được kết quả câu truy vấn rút gọn như được chỉ ra trong Hình 4.13d. Với thí dụ này muốn minh hoạ một điều là giá trị phân mảnh trong việc cải thiện hiệu năng của các câu truy vấn phân tán.



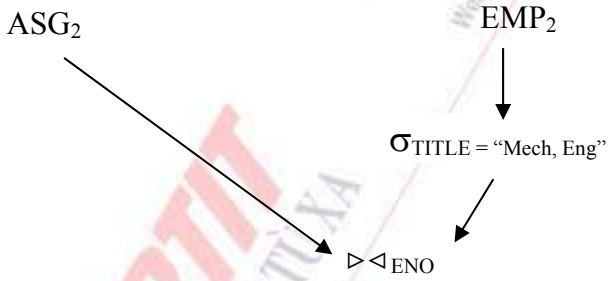
(a) Câu truy vấn gốc



(b) Câu truy vấn sau khi đẩy phép chọn xuống



(c) Truy vấn sau khi đẩy các phép hợp xuống



(d) Câu truy vấn đã rút gọn sau khi loại cây con bên trái

Hình 4.13 Rút gọn cho phân mảnh gián tiếp

4.6.4 Rút gọn cho phân mảnh hỗn hợp

Phân mảnh hỗn hợp bao gồm việc phân mảnh ngang và phân mảnh dọc. Mục đích của phân mảnh hỗn hợp là hỗ trợ một cách hiệu quả các câu truy vấn có chứa các phép chọn, phép chiếu va phép kết nối. Chương trình hoá cục bộ cho một quan hệ phân mảnh hỗn hợp có sử dụng phép hợp và kết nối các mảnh. Điều đáng quan tâm là để tối ưu hoá một phép toán hay tổ hợp các phép toán luôn luôn phải trả chi phí cao cho các phép toán khác. Ví dụ phân mảnh hỗn hợp dựa trên phép chiếu - chọn sẽ làm cho phép chiếu hoặc phép chọn kém hiệu quả hơn so với phân mảnh ngang hoặc phân mảnh dọc.

Ví dụ 4.14: Phân mảnh hỗn hợp của quan hệ EMP như sau:

EMP1 = $\sigma_{ENO \leq "E4"}(\pi_{ENO,ENAME}(EMP))$

EMP2 = $\sigma_{ENO > "E4"}(\pi_{ENO,ENAME}(EMP))$

EMP3 = $\pi_{ENO,TITLE}$(EMP)

Chương trình cục bộ hoá như sau:

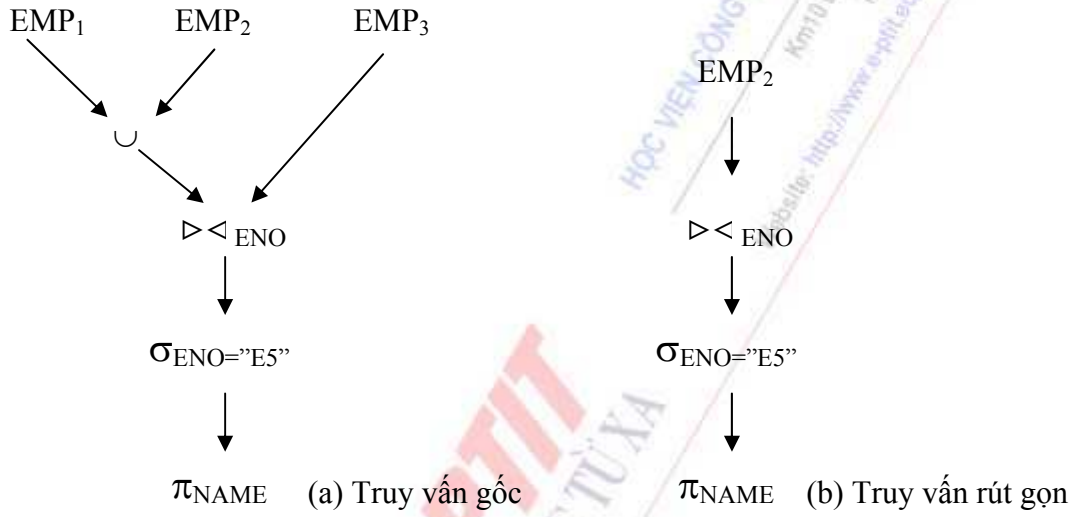
EMP = (EMP$_1$ $\cup$ EMP$_2$) $\triangleright\triangleleft_{ENO}$ EMP$_3$.

Các truy vấn trên những mảnh hỗn hợp có thể được rút gọn bằng cách kết hợp lần lượt các quy tắc trong phân mảnh ngang nguyên thuỷ, phân mảnh dọc và phân mảnh ngang dẫn xuất. Các quy tắc này có thể tóm tắt như sau:

4. Loại bỏ các quan hệ rỗng được tạo ra do các phép chọn mâu thuẫn nhau trên các mảnh ngang.
5. Loại bỏ các kết nối vô dụng được tạo do các phép chiếu trên các mảnh dọc.
6. Phân phối các kết nối cho các phép hợp nhằm cô lập và loại bỏ các kết nối vô dụng.

Ví dụ 4.15: Xét câu truy vấn sau

SELECT ENAME
FROM EMP
WHERE ENO = "E5" $\triangleright\triangleleft$



Hình 4.14: Rút gọn phân mảnh hỗn hợp

CÂU HỎI VÀ BÀI TẬP

1. Xử lý truy vấn trong các hệ cơ sở dữ liệu quan hệ phân tán. là:
   A. Cung cấp các phương tiện xây dựng các câu truy vấn và thực hiện tối ưu hoá truy vấn .
   B. Cung cấp các phương tiện thực hiện tối ưu hoá truy vấn .
   C. Cung cấp các câu truy vấn và thực hiện tối ưu hoá truy vấn .
2. Câu truy vấn phân tán là:
   A. Một chuỗi các thao tác đại số quan hệ trên CSDL cục bộ.
   B. Một chuỗi các thao tác đại số quan hệ trên CSDL cục bộ tối ưu hoá các nguồn tài nguyên. và trao đổi truyền thông.
   C. Một chuỗi các thao tác đại số quan hệ trên các mảnh dữ liệu được phân rã, được mở rộng với các thao tác truyền thông và tối ưu các nguồn tài nguyên.
3. Các phương pháp tối ưu cơ bản:

A. Biến đổi câu truy vấn tương đương và có chi phí thấp.
B. Chọn một biểu thức có chi phí thời gian và sử dụng tài nguyên là ít nhất.
C. Biến đổi câu truy vấn tương đương

4. Mục đích của việc xử lý truy vấn trong môi trường phân tán là:
   A. Thực hiện tối ưu hoá truy vấn.
   B. Biến đổi thành câu truy vấn tương đương.
   C. Tối ưu chi phí sử dụng tài nguyên của mạng.
5. Các kiểu tối ưu hoá
   A. Lựa chọn trong các giải pháp có chi phí là nhỏ nhất.
   B. Phương pháp tìm kiếm vét cạn, giải pháp ngẫu nhiên.
   C. Giải pháp thay thế phép kết nối bằng các tổ hợp các nối nửa
6. Thời điểm tối ưu hoá
   A. Kiểu tĩnh
   B. Tại các thời điểm khác nhau phụ thuộc thời gian thực hiện truy vấn.
   C. Kiểu động
7. Ưu điểm tối ưu hoá truy vấn theo kiểu tĩnh (Statically):
   A. Thực hiện khi biên dịch, chi phí giảm dần qua nhiều lần thực hiện. Kích thước của các quan hệ trung gian không biết trước
   B. Thực hiện khi biên dịch, chi phí giảm dần qua nhiều lần thực hiện.
   C. Thực hiện khi bắt đầu truy vấn, chi phí giảm dần qua nhiều lần thực hiện. Kích thước của các quan hệ trung gian không biết trước.
8. Ưu điểm tối ưu hoá truy vấn theo kiểu động
   A. Thực hiện khi biên dịch, chi phí giảm dần qua nhiều lần thực hiện. Kích thước của các quan hệ trung gian không biết trước
   B. Được thực hiện khi truy vấn. Thao tác tiếp theo tối ưu dựa trên kết quả của các thao tác trước đó.
   C. Đánh giá kích thước của các quan hệ trung gian không cần thiết.
9. Nhược điểm của phương pháp động là:
   A. Các thao tác tối ưu hoá có chi phí cao. Lặp lại nhiều lần cho mỗi thao tác.
   B. Các thao tác có chi phí cao, chi phí tăng dần qua nhiều lần thực hiện
   C. Kích thước của các quan hệ trung gian không phù hợp cho xử lý truy vấn.
10. Tối ưu hoá truy vấn hỗn hợp có các ưu điểm:
   A. Tối ưu hoá truy vấn tĩnh, tránh được các đánh giá không chính xác gây ra.
   B. Tối ưu hoá truy vấn động , có thể phát hiện có sự khác biệt giữa kích thước dự đoán và kích thước thực tế của các quan hệ trung gian.
   C. Của tối ưu hoá truy vấn động , hạn chế các nhược của truy vấn tĩnh.
11. Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp tĩnh, có thể :
   A. Sử dụng một vị trí hay nhiều vị trí.
   B. Sử dụng một vị trí
   C. Sử dụng nhiều vị trí.
12. Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp quyết định tập trung:
   A. Có một vị trí đưa ra giải pháp.
   B. Có nhiều vị trí đưa ra giải pháp.

C. Có một hoặc nhiều vị trí đưa ra giải pháp.

13. Khi thực hiện việc tối ưu hoá truy vấn bằng phương pháp hỗn hợp
    A. Có một vị trí quyết định chính, các vị trí khác đưa ra các quyết định cục bộ.
    B. Có thể được phân tán cho nhiều vị trí tham gia.
    C. Đòi hỏi phải biết toàn bộ về các thông tin cục bộ.
14. Quá trình cục bộ hoá là quá trình:
    A. Ánh xạ câu truy vấn phân tán mô tả trên quan hệ toàn cục thành các câu truy vấn trên các mảnh
    B. Nhân bản các mảnh ở nhiều vị trí khác nhau.
    C. Giảm số lần truyền thông
15. Thao tác nửa kết nối:
    A. Làm giảm kích thước của các quan hệ trung gian, làm giảm dữ liệu cần trao đổi giữa các vị trí.
    B. Làm giảm số lượng các thông điệp và thời gian xử lý cục bộ.
    C. Làm giảm các thao tác tối ưu hoá truy vấn
16. Thứ tự đúng caác tầng của quá trình xử lý truy vấn là:
    A. Tầng phân rã truy vấn, tập trung hoá dữ liệu, tối ưu hoá truy vấn toàn cục và tối ưu hoá truy vấn cục bộ.
    B. Tầng tập trung hoá dữ liệu, tối ưu hoá truy vấn toàn cục, tối ưu hoá truy vấn cục bộ và phân rã truy vấn,
    C. Tầng tập trung hoá dữ liệu, phân rã truy vấn, tối ưu hoá truy vấn toàn cục và tối ưu hoá truy vấn cục bộ.

17. Phân rã truy vấn và tập
    A. Có chức năng ánh xạ câu truy vấn phân tán ở dạng phép tính quan hệ thành câu truy vấn đại số trên quan hệ toàn cục.
    B. Có chức năng thực hiện tối ưu hoá truy vấn tại một vị trí tập trung và sử dụng các thông tin toàn cục.
    C. Có chức năng biến đổi phân rã truy vấn phân tán trên các quan hệ toàn cục.
18. Phân rã câu truy vấn có thể thực hiện các bước liên tiếp nhau:
    A. Bước chuẩn hoá, phân tích, loại bỏ dư thừa và xây dựng lại câu truy vấn
    B. Bước chuẩn hoá, phân tích và loại bỏ dư thừa
    C. Bước phân tích, loại bỏ dư thừa và xây dựng lại câu truy vấn
19. Chức năng chủ yếu của tầng cục bộ hoá dữ liệu phân tán:
    A. Chịu trách nhiệm chuyển câu truy vấn trên quan hệ toàn cục sang câu truy vấn trên các mảnh.
    B. Cung cấp các thông tin lưu trữ trong lược đồ phân mảnh cho quá trình cục bộ hoá phân tán.
    C. Xác định mảnh được sử dụng trong truy vấn và chuyển đổi câu truy vấn phân tán thành một truy vấn trên mảnh cụ thể.
20. Rút gọn phép chọn cho phân mảnh ngang nguyên thuỷ
    A. Bằng cách hoán vị phép chọn và phép hợp.
    B. Bằng cách hoán vị phép chọn và phép chiếu.

C. Bằng cách hoán vị phép chọn và phép kết nối

21. Rút gọn phép kết nối cho phân mảnh ngang nguyên thuỷ
    A. Bằng cách phân phối các phép kết nối trên các phép hợp
    B. Bằng cách phân phối các phép kết nối dưới các phép hợp
    C. Bằng cách phân phối các phép kết nối dưới các phép giao
22. Rút gọn cho phân mảnh dọc
    A. Bằng cách hoán vị phép chiếu và phép kết nối.
    B. Bằng cách hoán vị phép chọn và phép kết nối.
    C. Bằng cách hoán vị phép chiếu và phép chọn
23. Các câu truy vấn trên các mảnh dẫn xuất có thể được rút gọn
    A. Bằng cách phân phối các phép kết nối trên các phép hợp.
    B. Bằng cách phân phối các phép kết nối dưới các phép hợp
    C. Bằng cách phân phối các phép kết nối dưới các phép giao
24. Các truy vấn trên những mảnh hỗn hợp có thể được rút gọn bằng cách:
    A. Kết hợp các quy tắc trong phân mảnh ngang và phân mảnh dọc
    B. Kết hợp các quy tắc trong phân mảnh ngang nguyên thuỷ và phân mảnh dọc.
    C. Kết hợp các quy tắc trong phân mảnh ngang dẫn xuất và phân mảnh dọc.

CHƯƠNG V: TỔNG QUAN VỀ CSDL SONG SONG

Chương này bao gồm các nội dung chính sau đây:

- Chức năng hệ xử lý song song
- Kiến trúc hệ song song
- Các kỹ thuật hệ quản trị CSDL song song

5.1 MỞ ĐẦU

Sự tích hợp của các máy trạm trong môi trường phân tán cho phép phân phối chức năng hiệu quả hơn cho các chương trình ứng dụng chạy trên máy trạm, gọi là các máy phục vụ ứng dụng (Application Server), trong khi các chức năng dữ liệu được điều khiển bởi các máy tính chuyên dụng, gọi là các máy phục vụ dữ liệu (Database Server), dẫn đến khuynh hướng cấu trúc hệ thống phân tán ba bên (Three Tie Distributed System Architecture), trong đó các vị trí (Sites) được tổ chức thành các đại lý chuyên trách (Specialized Server).

5.2 CHỨC NĂNG HỆ XỬ LÝ SONG SONG

Các hệ thống dữ liệu song song kết hợp việc quản trị dữ liệu và xử lí song song làm tăng hiệu năng và tính sẵn sàng. Hiệu năng là mục tiêu của các máy CSDL (Database Machine) với hệ quản trị cơ sở dữ liệu truyền thống thường xẩy ra ùn tắc trong vào ra (I/O Bottleneck) do thời gian truy cập bộ nhớ phụ cao hơn so với thời gian truy cập bộ nhớ chính. Việc phân vùng CSDL trên nhiều đĩa sẻ đạt được khả năng song song của các liên truy vấn và nội truy vấn (Inter và Intra Query), cải thiện một cách đáng kể về thời gian đáp ứng và thông lượng các giao tác.

Một hệ CSDL song song có thể định nghĩa đơn giản như một hệ quản trị CSDL được cài đặt trên bộ đa xử lý kết chặt (Tightly Couple), bao gồm từ kết nối hệ quản trị CSDL hiện có với yêu cầu ghi lại các thủ tục giao diện hệ điều hành đến sự kết hợp phức tạp giữa xử lí song song và các chức năng hệ thống CSDL thành kiến trúc phần cứng/ phần mềm mới.

5.2.1 Các khía cạnh chức năng

Hệ thống CSDL song song hoạt động như Database Server cho Application Server trong mô hình Client - Server chung trong mạng máy tính. Hệ thống CSDL song song hỗ trợ các chức năng CSDL và giao diện Client - Server và có thể chức năng đa năng. Để hạn chế trao đổi thông tin giữa Client Server, cần thiết có một giao diện ở mức cao khuyến khích xử lý dữ liệu trên máy chủ.

Một hệ CSDL song song lý tưởng là cần phải cung cấp ưu điểm dưới đây với tỷ lệ giá thành/ hiệu năng tốt hơn so với máy tính lớn (mainframe) tương ứng. Những ưu điểm sau cũng là những ưu điểm của hệ thống CSDL phân tán.

5.2.2 Các ưu điểm CSDL song song

- *Hiệu năng cao (High Pperformance):* Các giải pháp hỗ hệ điều hành hướng CSDL, khả năng song song, tối ưu hóa, cân bằng tải. Cơ chế hoạt động song song có thể làm tăng lưu

lượng bằng việc sử dụng khả năng song song liên truy vấn, giảm thời gian đáp ứng các giao tác bằng việc sử dụng khả năng song song của các nội truy vấn. Tuy nhiên, việc làm giảm thời gian đáp ứng các truy vấn phức tạp qua cơ chế song song quy mô lớn cũng có thể sẻ tăng tổng thời gian bởi thời gian truyền thông, và làm ảnh hưởng đến lưu lượng.

- *Tính sẵn sàng cao (High Availability)*: Hệ thống CSDL song song bao gồm nhiều thanh phần tương tự nhau, có thể khai thác khả năng nhân bản dữ liệu để tăng tính sẵn sàng của CSDL. Trong hệ thống song song mức cao với nhiều ổ đĩa nhỏ, xác suất đĩa hỏng ở bất cứ thời điểm nào có thể cao. Vì vậy điều quan trọng sự cố đĩa hỏng không làm mất cân bằng tải, bằng giải pháp yêu cầu phân vùng các bản copy có thể truy cập song song
- *Khả năng mở rộng (Extensibility):* Trong môi trường song song, dễ dàng tăng kích thước CSDL hoặc tăng thông lượng sẻ dễ. Khả năng mở rộng dễ dàng bởi thêm khả năng xử lý và lưu trữ cho hệ thống, thể hiện thuận lợi sau: đường tuyến tính tỉ lệ (Linear Scaleup) và tuyến tính tốc độ Llinear Speedup). Linear Scaleup nói đến việc hiệu năng vẫn duy trì khi tăng tuyến tính kích thước CSDL và khả năng xử lý và lưu trữ. Linear Speedup nghĩa là đường tuyến tính làm tăng thêm tính thực thi với kích thước CSDL không đổi và tăng tính tuyến tính trong khả năng xử lý và lưu trữ.

5.2.3 Chức năng CSDL song song

- *Quản lý phiên (Sesion Manager)* giám sát giao tác, hỗ trợ các giao tác giữa Client với Server. Thực hiện kết nối và giải phóng kết nối giữa các tiến trình Client và hai hệ thống con khác. Vì thế nó khởi tạo và đóng phiên người sử dụng nhiều giao tác. Trong trường hợp phiên OLTP, quản lý phiên có thể bắt đầu sự thực hiện mã hóa giao tác được nhập vào trước trong Modul quản lý dữ liệu.
- *Quản lý yêu cầu (Request Manager)* nhận yêu cầu phía Client có liên quan tới biên dịch và thực thi truy vấn. Nó có thể truy cập vào thư mục CSDL chứa tất cả thông tin về dữ liệu và các chương trình, tác động vào các giai đoạn biên dịch khác nhau, bắt đầu thực hiện truy vấn và trả về kết quả , các lỗi mã cho ứng dụng Client. Bởi vì nó giám sát việc thực hiện các giao tác và xác nhận, nó có thể khởi đầu cho thủ tục phục hồi lại trong trường hợp giao tác bị lỗi. Để tăng tốc độ thực hiện truy vấn, tối ưu và xử lí song song các truy vấn tại cùng thời điểm biên dịch.
- *Quản lý dữ liệu (Data manager)* cung cấp tất cả chức năng mức thấp cần thiêt để chạy các truy vấn được biên dịch song song Nếu quản lý yêu cầu có thể biên dịch điều khiển luồng dữ liệu, sau đó thực hiện đồng bộ và truyền thông giữa các modul quản lý dữ liệu sau đó đồng bộ hóa và truyền thống giữa các module quản lí dữ liệu bởi các modul quản lý yêu cầu. Mặt khác, điều khiển giao tác và đồng bộ hóa phải được thực hiện bởi module quản lý yêu cầu

5.3 KIẾN TRÚC HỆ SONG SONG

Một hệ thống song song diễn tả sự dàn xếp trong các lựa chọn thiết kế cung cấp các ưu điểm với sự tốt nhất về mặt giá thành và sự thực thi. Một trong những vấn đề quan trọng về mặt thiết kế có liên quan đến tốc độ truyền thông là thiết bị phần cứng như bộ xử lý, bộ nhớ và các ổ đĩa. Kiến trúc hệ thống song song được phân chia thành hai loại lớn đó là kiến trúc chia sẻ bộ nhớíahared Memory) và kiến trúc không chia sẻ (Shared Nothing).

5.3.1 Kiến trúc chia sẻ bộ nhớ (Shared- Memory)

Trong kiến trúc này, một số bộ xử lý truy cập đến một số vùng nhớ hay đơn vị ổ đĩa thông qua liên kết nối nhanh (tôc độ bus cao) một số máy chủ mới thiết kế như IB3090 và đa bộ xử lý đối xứng như Sequent và Escala của Bull đều áp dụng mô hình này.

Các ví dụ về hệ thống CSDL song song chia sẻ bộ nhớ bao gồm XPRS và Volcano hiệu quả như các hệ quản trị cơ sở dữ liệu trong thương mại sử dụng đa bộ xử lý chia sẻ bộ nhớ. Trước tiên có thể nêu ra một ví dụ đó là hệ thống DB2 chạy trên IBM3090 với 6 bộ xử lý. Phần lớn các sản phẩm chia sẻ bộ nhớ dùng cho thương mại ngày nay có thể khai thác (Exploit) liên truy vấn song song (Inter Query Ơarallelism) để tăng hiệu năng giao tác và truy vấn nội song song (Intra Query Parallelism) để giảm thời gian đáp ứng của các truy vấn hỗ trợ quyết định (Decision Support).

Chia sẻ bộ nhớ có hai ưu điểm: tính đơn giản và tải trọng cân bằng. Siêu thông tin (thư mục) và thông tin điều khiển (ví dụ khóa bảng) có thể chia sẻ bởi tất cả các bộ xử lý, việc viết ứng dụng cơ sở dữ liệu trên kiến trúc đa bộ xử lý này không khác biệt so với viết trên máy tính đơn bộ xử lý. Đặc biệt liên truy vấn song song trở nên uyển chuyển, cân bằng tải trọng có thể đạt được tại thời điểm chạy sử dụng chia sẻ bộ nhớ.

Chia sẻ bộ nhớ có ba vấn đề cơ bản: giá thành (Cost), giới hạn mở rộng (Limited Extendsibility) và tính sẵn sàng (Availability) thấp. Giá thành cao do sự liên kêt nối phức tạp bởi vì cần thiết phải liên kết mỗi bộ xử lý tới mỗi Modul nhớ hay ổ đĩa. Với một bộ xử lý nhanh (thậm chí bộ nhớ catche lớn) , sự xung đột khi truy cập đến bộ nhớ chia sẻ tăng nhanh và giảm hiệu năng . Vì vậy sự mở rộng là giới hạn đến vài chục bộ xử lý (20 trên Sequent hoặc Encore). Cuối cùng khi bộ nhớ trống được chia sẻ bởi tất cả các bộ xử lý, một lỗi bộ nhớ có thể ảnh hưởng đến phần lớn các bộ xử lý khác do đó gây ra tổn thất về CSDL, giải pháp Sequoia sử dụng bộ nhớ kép

5.3.2 Kiến trúc chia sẻ đĩa (Shared-Disk)

Trong kiến trúc này , một số bộ xử lý truy cập đến các đơn vị đĩa thông qua liên kết nối nhưng không được phép (không chia sẻ) truy cập dến bộ nhớ chính. Khi đó mỗi bộ xử lý có thể truy cập đến các trang dữ liệu (database page) trên ổ đĩa chia sẻ và sao chép chúng đến bộ nhớ catche của nó. Để tránh xung đột khi truy cập đến cùng một trang, cần phải có cơ chế khóa toàn cục (Global Locking) và các giao thức dùng để bảo trì sự gắn kết của cache.

Các ví dụ về các hệ thống CSDL song song chia ser ổ đĩa bao gồm sản phẩm chia sẻ dữ liệu IMS/VS của IBM và các sản phẩm VAX DBMS, Rdb của DEC. Sự thực thi của Oracle trên VAXcluster của DEC và các máy tính NCUBE cũng sử dụng kiến trúc chia sẻ ổ cứng khi nó yêu cầu mở rộng của hệ quản trị cơ sở dữ liệu quan hệ (RDBMS).

Chia sẻ đĩa có một số ưu điểm về giá thành, khả năng mở rộng, cân bằng tải trọng, tính sẵn sàng và dễ dàng di chuyển từ các hệ thống có một bộ xử lý. Giá thành của liên kết nối (Interconnect) giảm đáng kế so với phương pháp chia sẻ bộ nhớ từ khi công nghệ Bus được dùng. Cho rằng mỗi bộ xử lý co đủ bộ nhớ cache, sự truy nhập vào vào đĩa chia sẻ là nhỏ nhất, do đó sự mở rộng có thể tốt hơn. Khi bộ nhớ bị lỗi có thể bị cô lập với các bộ xử lý khác, các node nhớ, tính sẵn sàng có thể cao hơn. Cuối cùng sự di chuyển từ hệ thống trung tâm tới đĩa chia sẻ dễ dàng hơn vì dữ liệu trên đĩa không cần tổ chức lại.

Chia sẻ đĩa có độ phức tạp cao hơn và hiệu năng cao hơn. Nó yêu cầu các giao thức của hệ phân tán dữ liệu như khóa phân tán và commit hai giai đoạn. việc bảo trì độ kết dính của các bản sao có thể làm quá tải truyền thông giữa các node. Việc truy cập đĩa chia sẻ có thể gây ra hiện tượng "nút cổ chai".

5.3.3 Kiến trúc không chia sẻ

Trong kiến trúc này , mỗi bộ xử lý truy cập độc lập đến bộ nhớ chính và đơn vị ổ đĩa. Vì vậy mỗi node có thể được xem như một site cục bộ (về CSDL và phần mềm) trong một hệ CSDL phân tán. Vì vậy phần lớn các giải pháp được thiết kế cho các hệ phân tán như phân đoạn dữ liệu, quản lý phân tán giao tác và xử lý truy vấn phân tán có thể được áp dụng. Các ví dụ về các hệ thống song song không chia sẻ bao gồm DBC của Teradata và NonStopSQL của Tandem cũng hiệu quả như các sản phẩm truyền thống như GRACE, EDS, GAMMA, BUBBA, PRISMA.

Giải thích sự tồn tại của các sản phẩm kiến trúc không chia sẻ có ba ưu điểm: về giá thành khả năng mở rộng và tính sẵn sàng. Ưu điểm về giá thành của phương pháp này cũng giống như ở phương pháp chia sẻ đĩa. Hệ cơ sở dữ liệu phân tán được cài đặt trong kiến trúc này có thể dễ dàng tăng thêm hiệu năng khi thêm các node mới. khả năng mở rộng tốt hơn (có thể lên tới hàng ngàn node). Ví dụ hệ thống DBC của Teradata có thể cung cấp 1024 bộ xử lý. Với các phân vùng dữ liệu có ích được đặt trên nhiều đĩa. Tốc độ tăng lên theo tuyến tính và phạm vi tăng tuyến tính có thể đạt được khối lượng công việc đơn giản. việc tạo các bản sao dữ liệu trên nhiều node có thể tăng tính sẵn sàng dữ liệu.

Kiến trúc không chia sẻ phức tạp hơn kiến trúc chia sẻ bộ nhớ bởi vì sự cần thiết phải cài đặt các chức năng phân tán dữ liệu tại nhiều node. Không giống như kiến trúc chia sẻ bộ nhớ và chia sẻ đĩa, độ cân bằng tải trọng quyết định bởi vị trí dữ liệu và tải trọng không hiện thực của hệ thống , hơn nữa khi thêm các node mới vào hệ thống có thể yêu cầu tổ chức lại dữ liệu cũng đề cập đến vấn đề độ cân bằng tải trọng.

5.3.4 Các kiến trúc phân cấp (Hierachical Architectures)

Kiến trúc phân cấp cũng, tên khác gọi là kiến trúc nhóm (Cluster Architecture) là kiến trúc kết hợp của hai kiến trúc không chia sẻ và kiến trúc chia sẻ bộ nhớ. Là kiến trúc không chia sẻ, các node được thiết kế có kiến trúc chia sẻ bộ nhớ. Kiến trúc này được đề xuất bởi Bhide, sau đó là Pirahesh và Boral . Một mô tả chi tiết được đề xuất bởi Graefe

Ưu điểm của kiến trúc phân cấp là hiển nhiên, nó kết hợp đặc điểm linh hoạt và hiệu năng của thành phần chia sẻ bộ nhớ với khả năng mở rộng của thành phần không chia sẻ. Trong mỗi node chia sẻ bộ nhớ (SM-Node) giao tiếp được thực thi có hiệu quả bởi thành phần chia sẻ bộ nhớ của kiến trúc, do đó hiệu năng tăng lên. Độ cân bằng tải trọng cũng tăng bởi thành phần chia sẻ bộ nhớ.

5.3.5 Các kiến trúc NUMA

Với mục đích mở rộng và tăng tính linh hoạt, kiển trúc chia sẻ bộ nhớ đa bộ xử lý hướng đến các kiến trúc NUMA với mục đích là cung cấp mô hình lập trình chia sẻ bộ nhớ và các lợi ích của nó trong phạm vi kiến trúc song song. Có hai lớp nổi bật trong kiến trúc NUMA: máy Cache Coherent NUMA (CC-NUMA) chuyển đổi bộ nhớ tại các node thành bộ nhớ cache có dung lượng không gian địa chỉ chia sẻ lớn. Vì vậy, vị trí của các mục dữ liệu (Data

Item) được tách ra hoàn toàn từ địa chỉ vật lý và mục dữ liệu của nó tự động di chuyển hay tái tạo lại trong bộ nhớ chính.

Vì bộ nhớ chia sẻ và cache liên kết hỗ trợ phần cứng nên bộ nhớ truy cập từ xa rất hiệu quả (chỉ một vài lần với giá thành của việc truy cập cục bộ NUMA dựa trên các chuẩn quốc tế và các thành phần xây dựng sẵn, ví dụ máy Data General nuSMP và Sequent NUMA-Q 2000 sử dụng chuẩn ANSI/IEEE Standard Scalable Coherent Interface (SCI) liên kết nối với các máy chủ SHV (Standard Hight Value), mỗi node SHV chứa 4 bộ xử lý pentium, hỗ trợ dung lượng bộ nhớ tối đa lên tới 4GB và hai hệ thống con ngang hàng PCI/IO, [Data General,. các ví dụ khác về loại máy tính NUMA đó là KSR1 của Kendal Square Research và SPP1200 của Convex có thể mở rộng ra hàng trăm bộ xử lý.

5.4 CÁC KỸ THUẬT HỆ QUẢN TRỊ CSDL SONG SONG

Việc thực thi hệ thống CSDL song song phụ thuộc vào các kĩ thuật CSDL phân tán. Về bản chất, giải pháp quản trị giao tác được sử dụng. Tuy nhiên, vấn đề tới hạn cho kiến trúc như trên là việc sắp đặt dữ liệu, khả năng truy vấn song song, xử lí dữ liệu song song và tối ưu hóa truy vấn song song. Giải pháp cho các vấn đề này phức tạp hơn DDBMS bởi vì số lượng các node nhiều hơn. Phần này sẻ ứng dụng kiến trúc không chia sẻ, vì nó là trường hợp chung và các kỹ thuật thực thi cũng có thể được áp dụng cho các kiến trúc khác.

5.4.1 Sắp đặt dữ liệu

Việc sắp đặt dữ liệu trong hệ thống CSDL song song được mô tả giống như việc phân mảnh trong CSDL phân tán. Những đặc điểm trong phân mảnh có thể được sử dụng để làm tăng tính song song của CSDL. Khái niệm Partitionning và Partition có thể hiểu như khái niệm phân mảnh ngang và phân mảnh dọc, trái ngược với các chiến lược lựa chọn bao gồm Clustering- nhóm một quan hệ vào một node đơn. Phân mảnh dọc có thể làm tăng tính song song và cân bằng tải như trong CSDL phân tán. Điểm giống nhau nũa là dữ liệu thường nhiều hơn các chương trình, các chương trình được thực hiện càng nhiều càng tốt tại nơi dữ liệu được tập trung.

Tuy nhiên, có hai điểm khác nhau cơ bản với CSDL phân tán. Một là, không cần tăng tối đa việc xử lí cục bộ tại mỗi node khi người sử dụng được liên kết đến các node đặc biệt. Hai là, việc cân bằng tải khó hoàn thành hơn trong số lượng node có sẵn. Vấn đề chính là để tránh việc tranh chấp tài nguyên, nó mang lại kết quả phá vỡ toàn bộ hệ thống (ví dụ, một node xử lí tất cả các công việc trong khi các node khác rỗi). Kể từ khi các chương trình được thực hiện nơi dữ liệu được tập trung, việc sắp đặt dữ liệu là vấn đề thực thi tới hạn.

Việc sắp đặt dữ liệu phải được thực hiện để tăng tối đa khả năng thực thi hệ thống, nó được đo bởi sự tổ hợp toàn bộ các công việc đã hoàn thành bởi hệ thống và thời gian đáp ứng các câu truy vấn đơn lẻ. Thông qua khả năng song song của các truy vấn trong, có thể làm tăng tối đa thời gian đáp ứng , kết quả là toàn bộ công việc được tăng lên thay vì việc truyền thông. Vì vậy, khả năng song song của các truy vấn làm cho toàn bộ công việc được tăng lên. Mặt khác, việc phân nhóm (Clustering) tất cả các dữ liệu cần thiết một chương trình giảm tối thiểu việc truyền thông và do đó toàn bộ công việc được làm bởi hệ thống trong việc thực hiện chương trình đó. Trong khái niệm sắp đặt dữ liệu, tăng tối đa thời gian đáp ứng hoặc khả năng song song của các truy vấn với nhau dẫn đến việc phân vùng trong khi đó việc giảm tối thiểu

các công việc dẫn đến phân nhóm. Vấn đề này được đề cập trong CSDL phân tán theo cách thức tĩnh. Người quản trị CSDL kiểm tra các đoạn Fragment theo định kỳ dựa theo tần suất, nếu cần thiết thì phải di chuyển hoặc tổ chức lại các Fragment.

Giải pháp lựa chọn cho việc sắp đặt dữ liệu là Full Partitioning phân vùng toàn bộ, do đó mỗi một quan hệ được phân mảnh ngang tới tất cả các node trong hệ thống. Phân vùng toàn bộ được sử dụng trong DBC/1012. GAMMA, Nonstoip SQL. Dưới đây là ba chiến lược cơ bản cho việc phân vùng dữ liệu: Round- Robin (luân chuyển), Hashing (hàm băm), Interval (khoảng cách)

5.4.2 Phân vùng luân chuyển (Round Rrobin Partitioning)

Là chiến lược đơn giản nhất, đảm bảo sự phân tán dữ liệu được đồng nhất. Với n vùng Partition, hàng thứ i được chèn vào vùng thú i mod n. Chiến lược này cho phép truy cập tuần tự tới một quan hệ được thực hiện song song. Tuy nhiên, khả năng truy cập tới các hàng riêng lẻ dựa trên việc truy cập đến các yêu cầu, thuộc tính của toàn bộ quan hệ.

5.4.3 Phân vùng băm (Hash Partitioning):

Chiến lược này áp dụng hàm băm cho một vài thuộc tính. Nó tạo ra một số Partition. Chiến lược này cho phép một node nhất định xử lí các truy vấn chính xác để lựa chọn các thuộc tính và tất cả các node xử lí tất cả các truy vấn khác một cách song song.

5.4.4 Phân vùng theo khoảng cách (Range Partitioning)

Chiến lược phân tán các hàng dựa trên miền giá trị của một vài thuộc tính. Ngoài ra, để hỗ trợ các truy vấn chính xác như là việc sử dụng bảng băm, nó phù hợp với các truy vấn theo miền. Ví dụ, một truy vấn "A between A1 and A2" có thể được xử lí bởi một node duy nhất chứa các hàng mà giá trị của nó nằm trong khoảng từ A1 đến A2. Tuy nhiên, việc phân vùng theo miền dẫn đến kích thước các vùng biến đổi nhiều.

5.4.5 Các giải pháp phân vùng

Việc thực thi phân vùng toàn bộ được so sánh với kĩ thuật phân nhóm các quan hệ trên một đĩa đơn. Kết quả đòi hỏi khối lượng công việc của nhiều người sử dụng khác nhau, việc phân vùng phù hợp hơn. Tuy nhiên, kĩ thuật phân nhóm có thể có ưu thế hơn trong việc xử lí các truy vấn phức tạp. Mặc dù việc phân vùng toàn bộ có nhiều ưu điểm về khả năng thực thi, việc thực hiện tính song song cao có thể gây ra việc thực thi liên quan đến các câu truy vấn phức tạp. Ví dụ, giả sử một cấu trúc có 1024 node, số lượng bản tin xấu nhất cho một kết nối nhị phân (không có lệnh Select) sẻ là 10242. Hơn nữa, phân vùng toàn bộ không phù hợp với các quan hệ nhỏ mà việc phân vùng toàn bộ được liên kết các khối đĩa lại với nhau. Các hạn chế này cần một sự thỏa hiệp giữa kĩ thuật phân nhóm và phân vùng toàn bộ.

Giải pháp cho việc sắp đặt dữ liệu là phân vùng biến đổi: Nói cách khác mức độ phân vùng, số lượng các node mà một quan hệ được phân mảnh, là hàm của kích thước và tần suất truy cập quan hệ đó. Chiến lược này phức tạp hơn kĩ thuật phân nhóm hay phân vùng toàn bộ bởi các thay đổi trong phân tán dữ liệu có thể phải tổ chức. Ví dụ, ban đầu một quan hệ được đặt tại 8 node, số các phần tử của nó có thể gấp đôi bằng cách chèn vào sau, và trong trường hợp này nó được đặt vào 16 node. Hệ thống song song với việc phân vùng biến đổi, tổ chức lại cho cân bằng tải một định kì là cần thiết và thường xuyên trừ khi khối lượng công việc là tĩnh và ít cập nhật dữ liệu. Sự tổ chức lại như vậy nên được trong suốt để biên dịch chương trình

chạy trên Server. Cụ thể hơn, các chương trình không nên biên dịch lại vì việc tổ chức lại này. Do đó, các chương trình đã biên dịch sẻ giữ lại độc lập với vị trí của dữ liệu, nó sẻ có thể thay đổi nhanh chóng. Sự độc lập như vậy có thể hoàn thành nếu hệ thống thời gian thực hỗ trợ truy cập kết hợp tới dữ liệu phân tán. Đây là sự khác biệt so với hệ quản trị CSDL phân tán, việc truy cập kết hợp được hoàn thành tại thời điểm biên dịch bởi bộ xử lí truy vấn sử dụng thư mục dữ liệu.

Một giải pháp cho việc truy cập kết hợp là có một cơ chế đánh chỉ mục toàn cục được sao chép cho mỗi một node. Chỉ mục toàn cục cho thấy việc sắp đặt một quan hệ vào một tập các node. Dựa trên các khái niệm đó, có hai mức chỉ mục với một kĩ thuật phân nhóm chính trên tên quan hệ và phân nhóm phụ trên một vài thuộc tính của quan hệ. Chỉ mục toàn cục hỗ trợ viêc phân vùng biến đổi, trong đó mỗi một quan hệ có mức phân vùng khác nhau. Cấu trúc chỉ mục có thể dựa trên cấu trúc B cây và hàm băm. Trong các trường hợp này, các truy vấn chính xác có thể được xử lí một cách hiệu quả với việc truy cập một node đơn. Tuy nhiên, với việc sử dụng hàm băm, các truy vấn theo miền được xử lí bởi việc truy cập tất cả các node chứa dữ liệu từ các quan hệ được truy vấn. Việc sử dụng bảng chỉ mục theo cấu trúc B cây sẻ lớn hơn theo cấu trúc hàm băm, nó cho phép xử lí các truy vấn theo miền một cách hiệu quả hơn, tại đó chỉ có một node duy nhất chứa dữ liệu trong miến dữ liệu cụ thế được truy cập.

Vấn đề đặt ra trong việc chọn đặt dữ liệu là giải quyết với các phân phối dữ liệu lệch mà chúng có thể dẫn đến phân hoạch không thống nhất và làm ảnh hưởng đến cân bằng tải. Phân hoạch theo khoảng cách dễ bị ảnh hưởng do lệch hơn so với phân hoạch xoay vòng hoặc băm. Một giải pháp là xử lý các phân hoạch không thống nhất một cách thích hợp, thí dụ bằng cách phân mảnh tiếp tục cho các phân hoạch lớn. Tách biệt giữa các nút logic và vật lý cũng có ích vì một nút logic có thể tương ứng với nhiều nút vật lý.

Tác nhân cuối cùng là sao chép dữ liệu để bảo đảm tính sẵn sang cao. Giải pháp đơn giản là duy trì hai bản sao của cùng một dữ liệu, một bản chính và một bản dự phòng trên hai máy riêng biệt. Đây là kiến trúc đĩa ảnh (Mirroed Disk) như đã được vận dụng trong hệ thống NonStop SQL của Tandem. Tuy nhiên trong trường hợp một node bị sự cố, tải trọng có thể bị nhân đôi lên tại node có bản sao, vì thế ảnh hưởng đến việc cân bằng tải. Để tránh vấn đề này, nhiều chiến lược sao chép dữ liệu có tính sẵn sàng cao đã được đề xuất cho các hệ CSDL song song. Một số giải pháp đáng chú ý là phân hoạch đan xen của Teradata. Nó phân hoạch bản dự phòng trên một số node. Ở tình huống có sự cố, tải trọng của bản chính sẻ được cân đối giữa các nút bản sao. Nhưng nếu cả hai nút có sự cố thì quan hệ đó không truy xuất được và vì thế làm ảnh hưởng đến tính khả dụng. Xây dựng lại bản chính từ các bản sao dự phòng riêng biệt có thể tốn nhiều chi phí. Ở tình huống bình thường, duy trì tính nhất quán cho các bản cũng có thể có chi phí cao.

Một giải pháp tốt hơn là phân hoạch xâu mắt xích của Gamma, lưu bản chính và bản dự phòng trên hai nút kế cận . Ý tưởng chính là xác suất hai node kế cận bị sự cố thường nhỏ hơn so với xác suất hai node bất kỳ bị sự cố. Ở tình huống bị sự cố, tải trọng của node bị sự cố và các node dự phòng được cân đối cho các node còn lại băng cách dùng node bản chính và bản dự phòng. Ngoài ra, việc duy trì tính nhất quán các bản đều rẻ hơn. Một vấn đề còn bỏ ngỏ là thực hiện việc chọn đặt dữ liệu có xem xét đến sao chép dữ liệu. Tương tự như việc cấp phát mảnh trong CSDL phân tán, điều này có thể được xem như một bài toán tối ưu hóa.

5.5 TRUY VẤN SONG SONG

Truy vấn song song cho phép thực hiện song song nhiều câu vấn tin sinh ra bởi các giao dịch đồng thời làm tăng lưu lượng giao dịch. Bên trong một câu vấn tin song hành nội toán tử và liên toán tử được sử dụng để giảm thời giam đáp ứng. Song hành liên toán tử có được bằng cách cho thực thi song song nhiều toán tử của cấu trúc cây vấn tin. Trên nhiều bộ xử lý trong khi đó song hành nội toán tử, một toán tử sẻ được nhiều bộ xử lý thực hiện, mỗi bộ xử lý thao tác trên một tập con dữ liệu.

5.5.1 Song hành nội toán tử

Song hành nội toán tử dựa trên việc phân rã một toán tử thành tập con các toán tử độc lập, được gọi là thể hiện tóan tử (Operator Instance). Phân rã này được thực hiện bằng cách dùng kỹ thuật phân hoạch tĩnh hoặc động cho các quan hệ. Sau đó mỗi thể hiện tóan tử sẻ thực hiện một phân hoạch quan hệ, thường gọi là lô(batch). Để minh họa cho việc song hành nội toán tử chúng ta xét một vấn tin chọn nối đơn giản. Tóan tử chọn (Select) có thể phân rã trực tiếp thành nhiều toán tử chọn, mỗi toán tử thao tác trên một phân hoạch khác nhau không cần phải thực hiện tái phân phối.

5.5.2 Song hành liên toán tử

Song hành liên toán tử có thể được dùng với song hành ống dẫn (Pipeline Parallelism) nhiều toán tử với một đường nối sản xuất – tiêu dùng được thực thi song song. Thí dụ toán tử Select được thực thi song song với toán tử nối (join) kế tiếp. Ưu điểm của thự thi theo phương pháp này là kết quả trung gian không phải cụ thể hóa (không phải lưu lại), vì thế tiết kiệm bộ nhớ và truy xuất đĩa. Tuy nhiên, nó chỉ có thể xảy ra với cách thực thi nhiều nhánh và đòi hỏi nhiều tài nguyên hơn.

5.5.3 Xử lý dữ liệu song song

Phân hoạch dữ liệu và sắp xếp chúng là cơ sở cho việc thực hiện truy vấn dữ liệu song song. Việc sắp xếp dữ liệu khi đã được phân hoạch rất quan trọng trong việc thiết kế các thuật toán song song và điều hành xử lý dữ liệu một cách hiệu quả (quan hệ giữa toán tử đại số), và các câu truy vấn dữ liệu bao gồm nhiều toán tử. Vấn đề này rất khó bởi vì cần phải đảm bảo sự cân bằng tốt giữa tính song song và chi phí cho quá trình truyền thông. Thuật toán song song cho các toán tử quan hệ đại số được xây dựng thành các khối cần thiết cho việc xử lý truy vấn song song.

Xử lý dữ liệu song song cho phép khai thác phép toán song hành nội toán tử toán tử. Như đã biết thuật toán song song cho việc điều hành dữ liệu dựa trên toán tử Select và Join, các toán tử cơ sở khác có thể điều khiển rất nhiều kết nối. Xử lý toán tử Select trong ngữ cảnh sắp xếp dữ liệu đã phân hoạch cũng giống như trong việc phân mảnh dữ liệu trong cơ sở dữ liệu phân tán, phụ thuộc vào sự lựa trọn Select , các toán tử được thực hiện tại một nod đơn hoặc trong trường hợp xác định một cách tuỳ ý tại tất cả các node thông qua quan hệ được phân hoạch. Nếu toàn bộ chỉ mục được tổ chức như cấu trúc B-Tree, khi đó toán tử Select với kích thước xác định có thể thực hiện bởi những nodelưu trữ dữ liệu thích hợp.

Việc xử lý song song cho toán tử kết nối Join phức tạp hơn nhiều so với toán tử chọn Select. Thiết kế thuật toán kết nối phân tán cho mạng có tốc độ cao có thể được áp dụng thành công trong CSDL phân tán. Có lẽ tính sẵn sàng của toàn bộ chỉ mục tại thời gian chạy cung

cấp đem lại nhiều thuận lợi cho việc thực hiện song song một cách có hiệu quả. Có ba thuật toán kết nối song song cơ bản cho việc phân hoạch dự liệu: Thuật toán vòng lập lồng song song PNL (The Parallel Nested Loop) ,thuật toán nối kết hợp song song PAJ (The Parallel Associative Join), thuật toán nối băm song song PHJ (The Parallel Hash Join).

CÂU HỎI TRẮC NGHIỆM

1. Nguyên nhân xẩy ra ùn tắc vào ra:
   A. Hiệu năng thấp
   B. Hệ quản trị cơ sở dữ liệu thường xẩy ra ùn tắc trong vào ra
   C. Thời gian truy cập bộ nhớ phụ nhiều hơn thời gian truy cập bộ nhớ chính.
2. Hiệu năng CSDL song song cao, nếu:
   A. Phân vùng CSDL trên nhiều đĩa.
   B. Có khả năng song song của các liên truy vấn và nội truy vấn.
   C. Cải thiện đáng kể về thời gian đáp ứng và thông lượng các giao tác.
3. Hiệu năng cao (High Pperformance):
   A. Các giải pháp hỗ hệ điều hành hướng CSDL, khả năng song song, tối ưu hóa, cân bằng tải.
   B. Các giải pháp tăng lưu lượng bằng việc sử dụng khả năng song song liên truy vấn
   C. Các giải pháp giảm thời gian đáp ứng các giao tác bằng việc sử dụng khả năng song song của các nội truy vấn
4. Tính sẵn sàng cao (High Availability)
   A. Hệ thống CSDL song song bao gồm nhiều thanh phần tương tự nhau
   B. Hệ thống CSDL song song có thể khai thác khả năng nhân bản dữ liệu
   C. Hệ thống CSDL song song sự cố đĩa hỏng không làm mất cân bằng tải
5. Khả năng mở rộng (Extensibility)
   A. Tăng kích thước CSDL hoặc tăng thông lượng.
   B. Tăng khả năng xử lý và lưu trữ cho hệ thống
   C. Tăng tính tuyến tính trong khả năng xử lý và lưu trữ.
6. Chức năng CSDL song song
   A. Quản lý phiên, yêu cầu và quản lý dữ liệu
   B. Quản lý giao tác
   C. Quản lý phiên và quản lý giao tác
7. Kiến trúc hệ song song bao gồm:
   A. Kiến trúc chia sẻ bộ nhớ, chia sẻ đĩa và không chia sẻ
   B. Kiến trúc bộ nhớ chính và bộ nhớ đệm.
   C. Kiến trúc câu truy vấn song song
8. Các kỹ thuật hệ quản trị CSDL song song bao gồm
   A. Phân vùng luân chuyển,phân vùng băm,phân vùng theo khoảng cách
   B. Phân vùng theo khái niệm Partitionning và Partition
   C. Phân vùng Partition và tất cả các node xử lí truy vấn song song.
9. Song hành nội toán tử là

A. Phân rã một toán tử thành tập con các toán tử độc lập
B. Phân rã một quan hệ thành mảnh con độc lập
C. Phân rã tóan tử chọn (Select) thành tập con các toán tử độc lập

10. Song hành liên toán tử
A. Thực hiện nhiều toán tử với một đường nối sản xuất
B. Thực hiện nhiều toán tử song song
C. Thực hiện nhiều toán tử và nhiều tài nguyên

11. Xử lý dữ liệu song song
A. Phân hoạch dữ liệu và sắp xếp dữ liệu
B. Thực hiện truy vấn dữ liệu song song.
C. Thuật toán song song

TÀI LIỆU THAM KHẢO

[1] Date C.J., " An introduction to data base systems", Second editon 1977.

[2] Codd, E.F., " Data models in data base management", ACM SIGMOD record,11,2(Feb,1981).

[3] Michanel V. Mannino, " Database Application Development & Design", Published by McGaw-Hill /Irwin, New Yor.k, 2001.

[4] Abram Siberschatz, Henry F.Korth, S.Sudarshan " Database Systems Concepts", Published by McGaw-Hill /Irwin, New Yor.k, 2002.

[5] M. Tamer Ozsu and Patrick Vaduriez, " Principles of Distributed Database Systems", Prentice-Hall 2003.

MỤC LỤC