**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

139

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 thíêt 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).

140

**5.3.1 Kiến trúc chia sẻ bộ nhớ (Shared- Memory)**

Trong ki ến 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 sẳn 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 sẳn 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.

141

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 sẳn 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

142

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 chưong 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

143

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 ch iến 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

144

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.

145

**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 ho ạch 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

146

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).