VII

Các hệ CSDL phân tán

Nội dung chi tiết

- ***Kiến trúc Client – Server***
- ***Kiến trúc Phân tán***
- ***Thiết kế CSDL phân tán***
- ***Các khái niệm cơ bản***
- ***Các vấn đề của hệ CSDL phân tán***
  - Xử lý truy xuất đồng thời bằng kỹ thuật khóa
  - Kiểm soát đồng bộ giao tác
  - Tối ưu hóa câu truy vấn phân tán
  - Kiểm soát quyền đăng nhập và quyền thao tác

- ***Ban đầu, kiến trúc File-Server***
  - Server là máy Main-Frame, chịu trách nhiệm thực hiện mọi xử lý
  - Client là các bộ thiết bị IO, không có khả năng xử lý
- ***Sau đó, kiến trúc Client-Server***
  - Client là một máy tính hoàn chỉnh có khả năng xử lý và lưu trữ Dữ liệu
  - Do đó, nảy sinh ý tưởng chia sẻ các xử lý cho cả Client và Server đảm trách
  - Việc chia sẻ xử lý
    - Cần có một giải pháp phân lớp hợp lý các công việc xử lý
    - Phụ thuộc vào từng giải pháp Client Server cụ thể

Kiến trúc Client Server (tt)

- ***Giải pháp phân lớp công việc xử lý***
  - Ban đầu người ta tập trung mọi xử lý vào một chương trình hợp nhất
    - Khó quản lý, tiềm ẩn nhiều hiểm họa
    - Thiếu tính khoa học, phụ thuộc nhiều vào kỹ thuật cá nhân của người lập trình (The art of programming → không ổn, lập trình không phải là 1 nghệ thuật)
  - Giải pháp hợp lý : Các xử lý của chương trình máy tính chia làm 3 lớp
    - Tạo và xử lý giao diện (1)
    - Tính toán chức năng (2)
    - Truy cập (đọc/ghi) dữ liệu (3)

- ***Các giải pháp Client-Server***
  - Giải pháp 1
    - Client : Toàn bộ (1) và (2)
    - Server : Toàn bộ (3)
    - Nhận xét : Bất hợp lý
  - Giải pháp 1
    - Client : Toàn bộ (1)
    - Server : Toàn bộ (2) và (3)
    - Nhận xét : Khá hơn giải pháp 1, nhưng vẫn chưa tối ưu
  - Giải pháp 3
    - Client : Toàn bộ (1) và một phần (2)
    - Server : Toàn bộ (3) và một phần (2)
    - Nhận xét : Hợp lý và tối ưu
    - Vấn đề : Chia sẻ (2) như thế nào ??? → Thảo luận

Kiến trúc Client Server (tt)

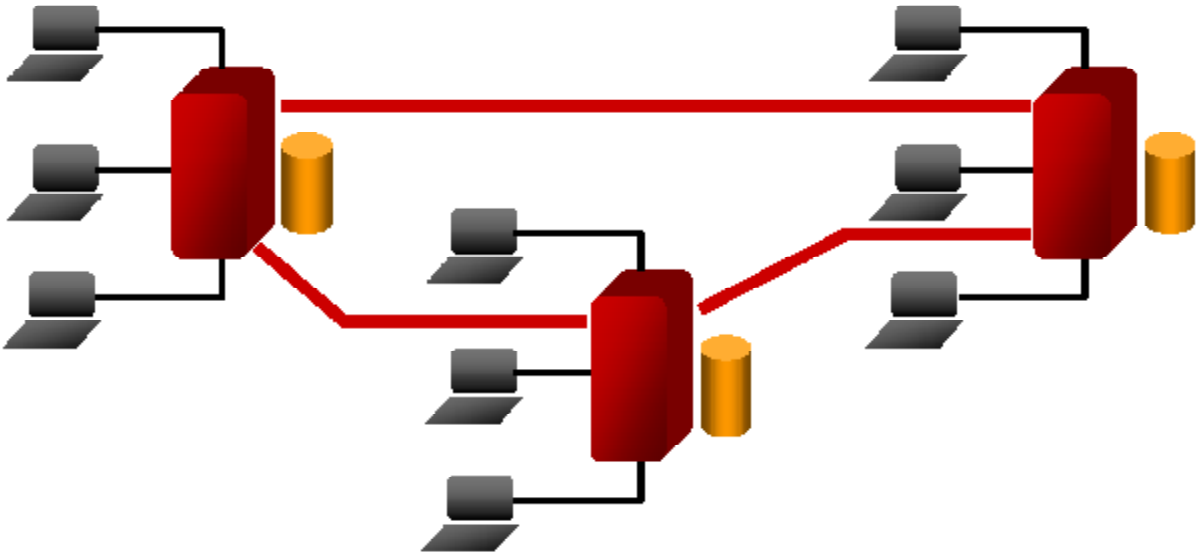
- ***Các bộ phần mềm cho kiến trúc Client-Server***
  - Microsoft
    - Hệ điều hành mạng : MS Windows Server 2003
    - DBMS : MS SQL Server
    - DE : Visual Studio.Net
  - Oracle
    - Hệ điều hành mạng : MS Windows Server 2003,Unix, Linux
    - DBMS : Oracle Server
    - DE : Developer 2000

- *Kiến trúc Client – Server*
- *Kiến trúc Phân tán*
- *Thiết kế CSDL phân tán*
- *Các khái niệm cơ bản*
- *Các vấn đề của hệ CSDL phân tán*
  - Xử lý truy xuất đồng thời bằng kỹ thuật khóa
  - Kiểm soát đồng bộ giao tác
  - Tối ưu hóa câu truy vấn phân tán
  - Kiểm soát quyền đăng nhập và quyền thao tác

Kiến trúc Phân tán

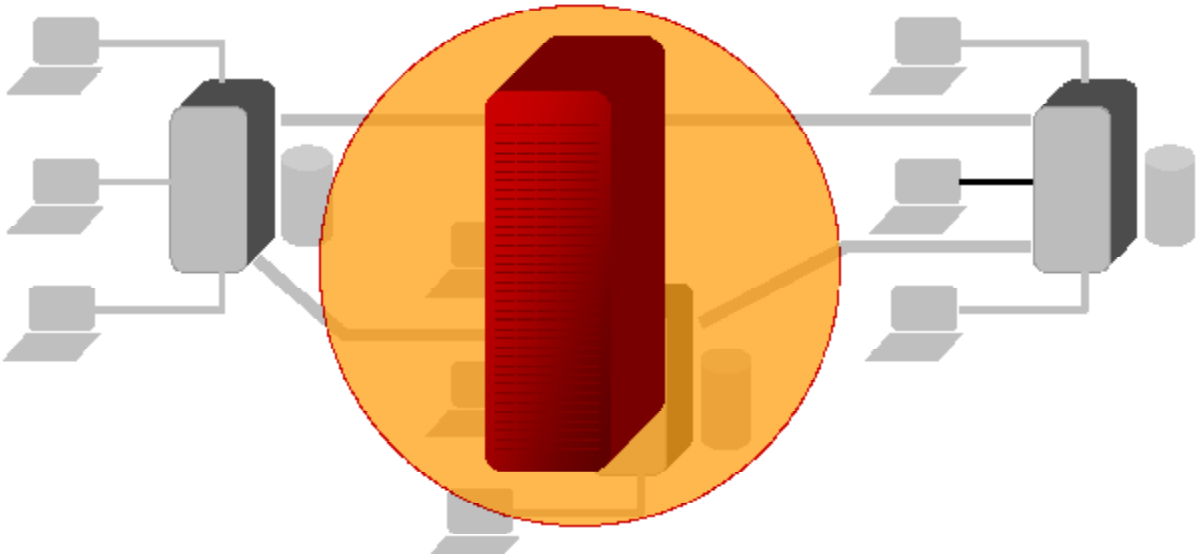
- *Tại sao cần kiến trúc phân tán ?*
  - Rất nhiều tổ chức thực tế trong cuộc sống là phân tán chứ không tập trung
  - Các cơ sở của một tổ chức như vậy đặt ở những vị trí địa lý khá xa nhau. Do vậy CSDL tập trung không đáp ứng được yêu cầu về tốc độ xử lý
  - Ví dụ
    - Ngân hàng ACB có các chi nhánh ở các tỉnh thành khác nhau trên toàn quốc
    - Yahoo, Google đều có các Server ở nhiều quốc gia khác nhau trên toàn thế giới
    - Hệ thống bán vé cho các chuyến bay quốc tế (cần nắm thông tin về lịch trình của nhiều hãng hàng không các nước)
  - Mục tiêu : Đưa Dữ liệu đến càng gần nơi cần nó càng tốt

- ***Định nghĩa kiến trúc phân tán***
  - Một hệ thống phân tán là một tập hợp các hệ thống Client-Server được nối mạng với nhau nhằm chia sẽ dữ liệu
  - Một người dùng kết nối vào hệ thống thì cảm thấy như mình đang làm việc với 1 server luận lý rất lớn duy nhất



Kiến trúc Phân tán (tt)

- ***Định nghĩa kiến trúc phân tán***
  - Một hệ thống phân tán là một tập hợp các hệ thống Client-Server được nối mạng với nhau nhằm chia sẽ dữ liệu
  - Một người dùng kết nối vào hệ thống thì cảm thấy như mình đang làm việc với 1 server luận lý rất lớn duy nhất



- ***Kiến trúc Client – Server***
- ***Kiến trúc Phân tán***
- ***Thiết kế CSDL phân tán***
- ***Các khái niệm cơ bản***
- ***Các vấn đề của hệ CSDL phân tán***
  - Xử lý truy xuất đồng thời bằng kỹ thuật khóa
  - Kiểm soát đồng bộ giao tác
  - Tối ưu hóa câu truy vấn phân tán
  - Kiểm soát quyền đăng nhập và quyền thao tác

Thiết kế CSDL Phân tán

- ***Thiết kế kiểu Top-down***
  - Giả định rằng ta đang thiết kế 1 CSDL tập trung
  - Khi đã có CSDL tập trung, phân rã nó ra từng lược đồ cục bộ cho từng server trong hệ phân tán
  - Nhận xét
    - Giải pháp này lý tưởng hóa bài toán CSDL phân tán → không thực tế
    - Khó áp dụng vì rất khó hình dung một bức tranh toàn cảnh ngay từ khi tổ chức / doanh nghiệp mới thành lập
    - Thông thường : Ban đầu các trụ sở, đại lý tự mình vận hành 1 CSDL cho ổn. Khi tổ chức / doanh nghiệp đạt đến một mức độ phát triển nhất định thì mới nghĩ đến việc thống nhất các trụ sở, đại lý lại với nhau thành một 1 hệ thống hợp nhất (hệ thống phân tán) → Thiết kế kiểu Bottom-up

- ***Thiết kế kiểu Bottom-up***
  - Dựa trên các kiến trúc Client-Server có sẵn (với những lược đồ cục bộ) để tổng hợp nên hệ phân tán (với lược đồ tổng thể)
  - Nhận xét
    - Đối diện thách thức lớn : Bài toán Tích hợp lược đồ (Schema-mapping) vì thông thường các lược đồ cục bộ là không thống nhất
    - Giải pháp
      - *Xây dựng lược đồ tổng thể trung gian để thống nhất các lược đồ cục bộ*
      - *Dùng WebService : Mỗi lược đồ cục bộ có 1 "người đại diện" là 1 webservice. Các webservice trao đổi Dữ liệu với nhau thông qua chuẩn XML*

Nội dung chi tiết

- ***Kiến trúc Client – Server***
- ***Kiến trúc Phân tán***
- ***Thiết kế CSDL phân tán***
- ***Các khái niệm cơ bản***
- ***Các vấn đề của hệ CSDL phân tán***
  - Xử lý truy xuất đồng thời bằng kỹ thuật khóa
  - Kiểm soát đồng bộ giao tác
  - Tối ưu hóa câu truy vấn phân tán
  - Kiểm soát quyền đăng nhập và quyền thao tác

Phân mảnh dữ liệu (Data fragmentation)

- Mỗi Server S_i có 1 lược đồ cục bộ Q_i của nó
- Các hệ thống phân tán thế hệ 1 đòi hỏi $\cap_{i=1..n} Q_i = \varnothing$
  - Nhận xét : Không hiệu quả, đánh mất ½ ý nghĩa của hệ phân tán, trả giá quá lớn khi truy xuất Dữ liệu từ vị trí địa lý quá xa
- Hiện nay $\cap_{i=1..n} Q_i \neq \varnothing$ → Thể hiện tính chia sẻ, các Server chia bản sao dữ liệu của mình cho các server khác cùng giữ
  - Không thể chia sẻ quá nhiều (Vd : Cho hết cả Table hay DB) vì mất tính bảo mật và làm nặng nề cho bài toán đồng bộ hóa
  - Giải pháp : Cắt dữ liệu thành những mảnh nhỏ (có thể chia sẻ được) và chép cho các server khác những bản sao của mảnh dữ liệu ấy → Phân mảnh dữ liệu

Các khái niệm cơ bản (tt)

Phân mảnh dữ liệu (tt)

- Phân mảnh ngang (horizontal fragmentation) nguyên thủy
  - Một mảnh Dữ liệu được hình thành từ phép chọn trên một quan hệ
  - Vd :
    - *KH(MSKH, TenKH, DiaChi, DienThoai, ThanhPho)*
    - $KH\_TpHCM = \sigma_{ThanhPho='HCM'}(KH)$
- Phân mảnh ngang dẫn xuất
  - Do mối quan hệ 1-n giữa 2 quan hệ → Đã phân mảnh ngang nguyên thủy trên 1 quan hệ thì phải phân mảnh ngang trên quan hệ còn lại (Phân mảnh ngang dẫn xuất)
  - Vd :
    - *GD(SoGD,LoaiGD,…,**MaKH**)*
    - $GD\_TpHCM = GD \ltimes KH\_TpHCM$

Phép kết nửa : Trong kết quả chỉ có thuộc tính của GD

Phân mảnh dữ liệu (tt)

- Các tính chất của Phân mảnh ngang
  - Tính đầy đủ : Nếu R được phân thành n phân mảnh ngang r_1, r_2, .., r_n thì hội n phân mảnh ngang đó sẽ phục hồi đúng R ban đầu
  - Tính tách biệt : Các r_i giao nhau bằng rỗng (do chúng được phân mảnh bằng cùng một tiêu chí phép chọn)

Các khái niệm cơ bản (tt)

Phân mảnh dữ liệu (tt)

- Phân mảnh dọc (Vertical Fragmentation)
  - Một mảnh Dữ liệu được hình thành từ phép chiếu trên một phần thuộc tính của quan hệ gốc
  - Vd :
    - *GD(SoGD, Ngay, Loai, SoTien, MSKH)*
    - $GD1 = \pi_{(SoGD,Ngay,Loai,SoTien)}GD$
    - $GD2 = \pi_{(SoGD,MaKH)}GD$
  - Nhận xét : Nếu ta chấp nhận sự trùng lắp thông tin khóa chính (trong ví dụ trên là SoGD) thì việc phân mảnh ngang thỏa tính đầy đủ và tính tách biệt
- Phân mảnh hỗn hợp : Vừa phân mảnh ngang, vừa phân mảnh dọc theo 1 trình tự hợp lý → Bài toán khó, linh động tùy từng trường hợp

- ***Nhân bản Dữ liệu (Replication)***
  - Sau khi phân mảnh Dữ liệu, một mảnh Dữ liệu sẽ cần được nhân ra nhiều bản để chép đến những Server cần dùng nó → Việc nhân bản dữ liệu
  - Do các bán sao của cùng một mảnh dữ liệu đồng thời tồn tại ở nhiều nơi → Vấn đề đồng bộ dữ liệu
    - Đồng bộ tức thời
    - Đồng bộ trễ (lazy update)

Nội dung chi tiết

- ***Kiến trúc Client – Server***
- ***Kiến trúc Phân tán***
- ***Thiết kế CSDL phân tán***
- ***Các khái niệm cơ bản***
- ***Các vấn đề của hệ CSDL phân tán***
  - Xử lý truy xuất đồng thời bằng kỹ thuật khóa
  - Kiểm soát đồng bộ giao tác
  - Tối ưu hóa câu truy vấn phân tán
  - Kiểm soát quyền đăng nhập và quyền thao tác

- ***Xử lý truy xuất đồng thời bằng kỹ thuật khóa***
  - Giải pháp Node trung tâm : Chọn 1 server thật mạnh làm lock manager cấp phát khóa cho toàn hệ thống
    - Ưu : Dễ cài đặt triển khai
    - Khuyết : Quá nguy hiểm nếu Node trung tâm có sự cố
  - Giải pháp khóa bản chính (primary copy) : Một yêu cầu xin khóa 1 đơn vị dữ liệu A phải được gửi về server giữ bản chính của A và chỉ server này có quyền quyết định cấp khóa hay không
    - Ưu : Khắc phục nguy cơ tê liệt hệ thống
    - Khuyết : Khó khăn khi Server chứa bản chính ở quá xa
  - Giải pháp khóa phân tán (Read lock one/Write lock all)
  - Giải pháp xin khóa quá bán : Dù đọc hay ghi cũng xin khóa trên n/2 + 1 node khác

Các vấn đề của một HQT CSDL phân tán (tt)

- ***Kiểm soát đồng bộ giao tác***
  - Giao tác T nảy sinh ở 1 server sẽ tạo ra các giao tác con t_i tại một số server khác trong hệ phân tán
  - Nếu giải pháp đồng bộ tức thời thì
    - T chỉ commit khi <u>tất cả</u> t_i đều đã commit
    - Chỉ cần một t_i bị hủy thì tất cả t_j khác và T sẽ bị hủy theo
    - Server phát sinh ra T gọi là "người điều phối"
    - Các server nơi chạy các t_i gọi là "người tham dự"
  - Việc đồng bộ giữa T với các ti và giữa các ti với nhau được thực hiện thông qua *Giao thức hoàn tất hai giai đoạn* (2 phase commit – 2PC)

Kiểm soát đồng bộ giao tác (tt)

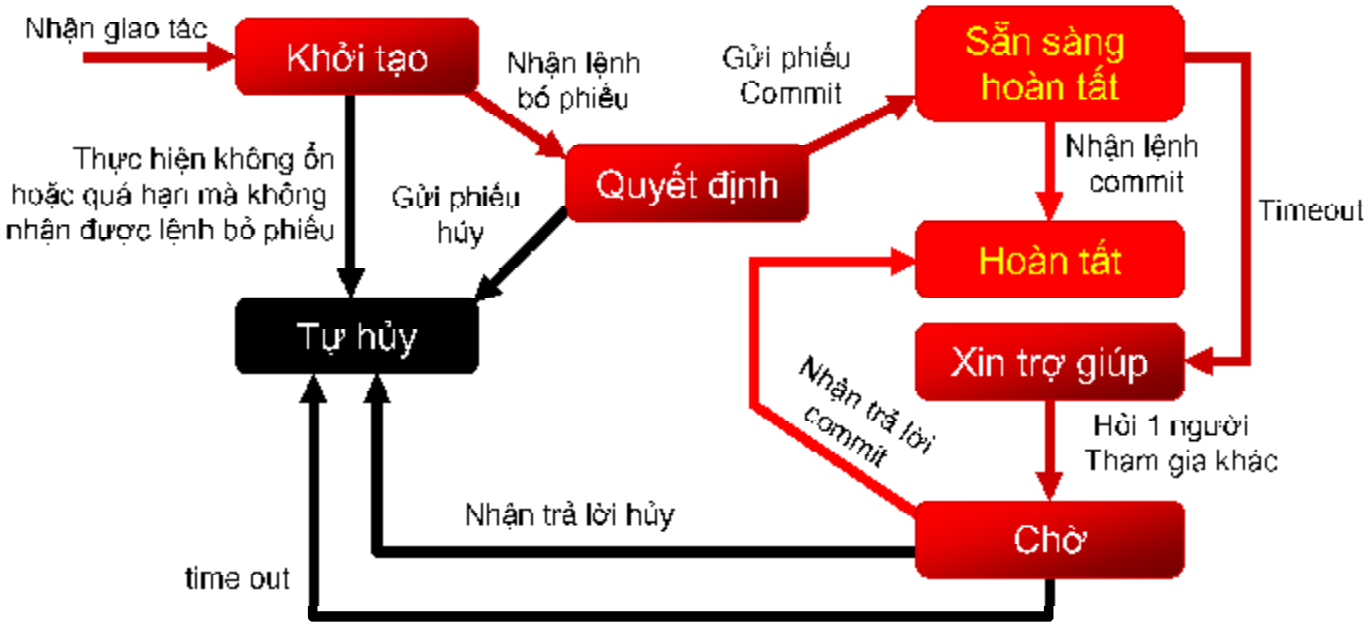
Giao thức hoàn tất hai giai đoạn (2 phase commit – 2PC)

- Sơ đồ trạng thái người điều phối

Kiểm soát đồng bộ giao tác (tt)

Giao thức hoàn tất hai giai đoạn (2 phase commit – 2PC)

- Sơ đồ trạng thái người tham gia



Kiểm soát đồng bộ giao tác (tt)

- Giao thức hoàn tất hai giai đoạn (2 phase commit – 2PC)
  - Cách người tham gia trả lời khi nhận yêu cầu trợ giúp từ người tham gia khác :
    - *Một thành viên đang trong trạng thái "Xác nhận" sẽ trả lời hoàn tất.*
    - *Một thành viên đang trong trạng thái "Hủy" sẽ trả lời hủy.*
    - *Một thành viên đang trong trạng thái "Sẵn sàng xác nhận" sẽ không trả lời (Do không có khả năng giải quyết vấn đề)*
    - *Một thành viên chưa biểu quyết (nghĩa là đang trong trạng thái bắt đầu) sẽ không trả lời (Do không có khả năng giải quyết vấn đề)*

Kiểm soát đồng bộ giao tác (tt)

- Giao thức hoàn tất hai giai đoạn (2 phase commit – 2PC)
  - Lưu ý :
    - *Một giao dịch có thể đã chuyển đến trạng thái kết thúc và vẫn được yêu cầu giúp đỡ qua thông báo "Trợ giúp"*
    - *Hệ thống sẽ dựa vào nhật ký để trả lời. Tất cả mọi thông báo và thay đổi trạng thái đều do hệ thống quản lý, chứ không phải được cài vào trong các giao dịch.*

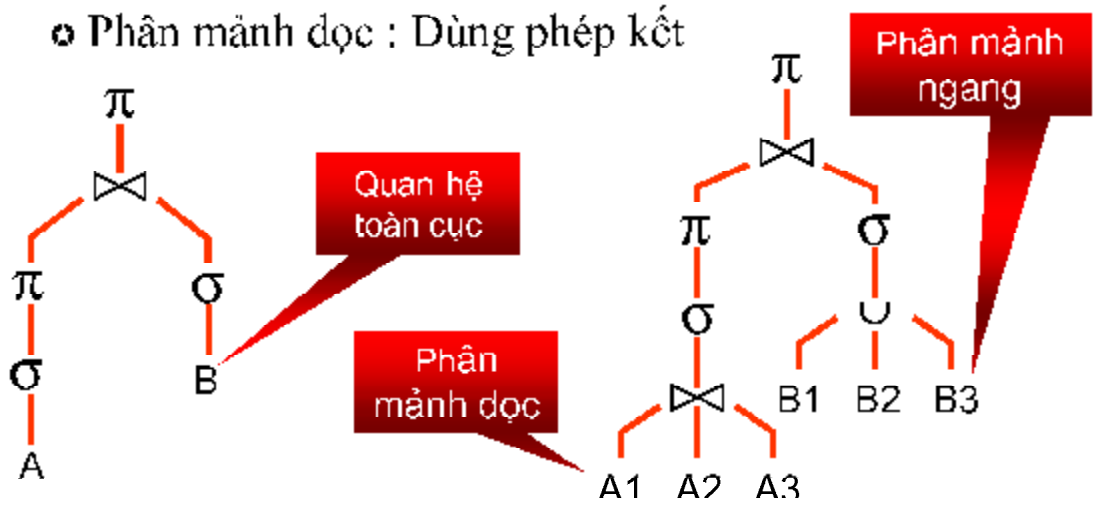
Kiểm soát đồng bộ giao tác (tt)

- Giao thức hoàn tất hai giai đoạn (2 phase commit – 2PC)
  - Các tồn tại :
    - *Xét giao tác con đã ở trạng thái sẵn sàng xác nhận, nhưng điều phối bắt đầu lỗi hoặc đường truyền mạng lỗi → Giao tác con bị phong tỏa, giữ lại toàn bộ tài nguyên cho đến khi nhận lệnh kết thúc → Giảm tính sẵn sàng của hệ thống trong trường hợp chịu lỗi.*
    - *Chưa giải quyết trường hợp người tham gia gửi yêu cầu trợ giúp nhưng không được bất cứ người tham gia nào khác đáp lại.*
    - *Chưa giải quyết trường hợp lỗi sảy ra ở người điều phối sau khi đã gửi phiếu commit mà chưa nhận lệnh commit từ người điều phối.*

Các vấn đề của một HQT CSDL phân tán (tt)

Tối ưu hóa câu truy vấn phân tán

- Giả sử câu truy vấn thao tác trên lược đồ toàn cục → Tiến hành tối ưu trên lược đồ toàn cục (như CSDL tập trung)
- Trong cây truy vấn, thay quan hệ toàn cục bằng các phân mảnh
  - Phân mảnh ngang : Dùng phép hội
  - Phân mảnh dọc : Dùng phép kết



- ***Kiểm soát quyền đăng nhập***
  - Mức 1 : User muốn thao tác DL ở Server nào thì phải có tài khoản đăng nhập server đó → vô cùng bất tiện
  - Mức 2 : Tùy mức độ chia sẻ mà tạo sẵn những đường link từ server A đến server B. Một lớp người dùng nào đó khi đăng nhập với quyền phù hợp ở A sẽ thông qua những đường link phù hợp để truy cập B dưới danh nghĩa của A
- ***Kiểm soát quyền thao tác***
  - Truy cập được 1 server không có nghĩa là được thao tác lên mọi đối tượng dữ liệu trong server ấy
  - Cần có cơ chế phân quyền cụ thể trên từng đối tượng dữ liệu tùy vào mức đố chia sẻ của đối tượng dữ liệu đó
    - Cách 1 : Giao cho 1 server đảm trách phân quyền trên các đối tượng DL của các server khác → hình thành node trung tâm
    - Cách 2 : Mỗi server tự cấp quyền cho các đối tượng dữ liệu của mình và cấp quyền cho các Server khác trong hệ

Hết chương VII