**CHƯƠNG 1:**

1. **Các bước phân tán? Khi phân tán cần bật dịch vụ gì?**

* Cấu hình các server:
* Bước 1: cấu hình distribution
* Publisher (server gốc) : chứa bản cơ sở dữ liệu gốc
* Distributor: điều phối các phân mảnh phân tán
* Subscriber: server chứa cơ sở dữ liệu sau khi phân tán
* Bước 2: Tạo publication
* Định nghĩa Publication: 1 container chứa actical ( view, sp, table, UDF), chứa lược đồ phân mảnh , thể hiện mối quan hệ dữ liệu giữa các table khi chúng ta phân tán dữ liệu.
* Định nghĩa Subcrition(database): 1 container nhận Publication (chỉ định cái server nào sẽ chứa cái publication, chứa dữ liệu phân tán theo lược đồ phân mảnh)
* Khi phân tán cần bật dịch vụ: SQL Server Agent
* UDF: hàm do người dùng tự định nghĩa
* Trong 1 server có nhiều user – tên user có thể trùng nhau
* Trong 1 database có nhiều user – tên user ko đc trùng nhau
* Trong 1 server có nhiều login- tên login ko đc trùng nhau
* 1 login chỉ thuộc 1 user

1. **Thuộc tính của folder RELPDATA là gì? ở đâu? Mục đích của folder này dùng để làm gì?**

* Thuộc tính của folder RELDATA là share, có quyền read/write. Mục đích của folder là để chứa các dữ liệu trao đổi trong quá trình update dữ liệu từ các phân mảnh về CSDL gốc, và từ CSDL gốc đến các phân mảnh.

**CHƯƠNG 2: SỰ TRONG SUỐT PHÂN TÁN**

1. **Các hình thức phân mảnh:**
2. **Khi nào thì cần phân tán cơ sở dữ liệu?**

* Nếu cơ sở dữ liệu tập trung không giải quyết được nên cần phải phân tán. Do lập trình trên môi trường phân tán dễ sinh lỗi hơn.

1. **Lý do phân mảnh?**

* Khi có một quan hệ mà khi ta phân tán qua nhiều site sẽ gồm 2 khái niệm:

+ Nhân bản:

* Ưu điểm: hỗ trợ truy vấn cao
* Nhược điểm: khi cập nhật dữ liệu ở site phân mảnh bất kỳ, thời gian cập nhật sẽ lâu. Vì khi sửa dữ liệu ở một site phân mảnh => đồng bộ về dite chủ => site chủ sẽ kiểm tra trên hệ thống phân tán và nhảy đến những site còn lại để cập nhật.

+ Phân hoạch:

* Ưu điểm: thời gian cập nhật dữ liệu sẽ nhanh hơn. Vì chỉ cập nhật dữ liệu ở site chứa nó và site chủ
* Nhược điểm: hỗ trợ truy vấn thấp.

1. **Khi ta phân tán thì có bao nhiêu hình thức phân mảnh?**

* Gồm 3 hình thức: phân mảnh theo chiều ngang, phân mảnh theo chiều dọc, hỗn hợp.

1. **Quy tắc phân mảnh?**

* Tính đầy đủ: Nếu một quan hệ R được phân rã thành n mảnh thì mỗi một mục dữ liệu (dòng hay cột) trong quan hệ R nó phải nằm ít nhất ở một phân mảnh => không bị mất thông tin.
* Tính tái thiết: Nếu một quan hệ R được phân rã thành n mảnh thì ta phải định nghĩa được một phép toán trên hệ thống phân tán sao cho với phép toán đó sẽ tác động lên n phân mảnh để tái tạo lại được dữ liệu của quan hệ gốc R.

+ Phân mảnh ngang: dùng phép hội (Union).

+ Phân mảnh dọc: dùng phép kết tự nhiên theo khóa chính.

+ Phân mảnh hỗn hợp: Union + phép kết tự nhiên theo khóa chính.

* Tính tách biệt: Nếu một quan hệ R được phân rã thành n mảnh thì mỗi mục dữ liệu nằm trong R chỉ nằm duy nhất ở một phân mảnh.
* Nếu quan hệ thỏa tính đầy đủ chắc chắc sẽ tái thiết được.
* Trong thực tế, tính tách biệt sẽ bị vi phạm.

1. **Các hình thức phân mảnh?**

* Phân mảnh ngang: tách quan hệ thành nhiều dòng.

+ Phân mảnh ngang nguyên thủy: phân mảnh một quan hệ dựa trên một điều kiện, quan hệ R thỏa mãn điều kiện đó => những phân mảnh Ri

+ Phân mảnh ngang dẫn xuất: dữ liệu của nó khi di chuyển qua một site nào đó là dựa vào dữ liệu của một quạn hệ khác.

* Phân mảnh dọc: theo phép chiếu – tách hoặc chia các thuộc tính, các field, các cột của một quan hệ qua nhiều site. Sẽ đúng nếu mỗi thuộc tính của quan hệ gốc phải nằm ít nhất ở 1 site phân mảnh => đảm bảo tính đầy đủ. Để tái thiết được thì mỗi phân mảnh phải chứa khóa chính của quan hệ gốc. Những thuộc tính không sử dụng không cần đưa qua nhưng những thuộc tính NOT NULL bắt buộc phải đưa qua site phân mảnh.
* Phân mảnh hỗn hợp

1. **Sự trong suốt phân tán**

* Trong suốt:
* Nghĩa là khi một SP khi ta cho thực thi ở bất kì một server phân mảnh nào thì nó vẫn có thể thực thi được ở tất cả server phân mảnh khác mà ta không cần chỉ lại đường dẫn đến SP cần truy xuất => cách để đạt sự trong suốt là nhân bản, nhân bản đến những nơi cần thiết.
* Người dùng không cảm nhận được là SP đang chạy trên hệ thống phân tán => xử lí code
* Viết SP đạt sự trong suốt phân tán:
* Tên các database ở các Server Subscriber phải giống nhau.
* Tạo Link server giữa các phân mảnh phải có tên giống nhau.
* Sự trong suốt phân tán ở ứng dụng chỉ đọc:
* Mức trong suốt p hân tán:
* Mức 1: Sự trong suốt phân tán – cao nhất – người dùng: không cần chỉ ra các phân mảnh cũng như các vị trí cấp phát cho các phân mảnh đó. Khi người dùng thao tác sẽ không có cảm giác thao tác trên một câu truy vấn.
* Mức 2: Sự trong suốt vị trí – vô lí không thực hiện được: phải chỉ rõ vị trí – tên table, không cần chỉ rõ nơi chứa.
* Mức 3: Sự trong suốt ánh sáng cục bộ – người lập trình: phải chỉ vị trí – table nào, nơi chứa – site nào. Trong sql, cách chỉ rõ site là dùng link.
* Mức 4: Không trong suốt: phải chỉ rõ luôn hệ quản trị, tên database. => Nhờ đặt tên database phân mảnh trùng tên với database site chủ.
* Sự trong suốt phân tán ở ứng dụng cập nhật: là giao tác, mức trong suốt phân tán giống ứng dụng chỉ đọc.
* Lưu ý:
* Khi cập nhật số liệu trên một phân mảnh, thì dữ liệu có khả năng nhân bản hoàn toàn – sẽ thay đổi dữ liệu trên phân mảnh đó đồng thời cập nhật dữ liệu ở tất cả các site còn lại => hệ thống tự đồng bộ, hay 1 phần.
* Viết SP cập nhật dữ liệu, vấn đề di chuyển dữ liệu sẽ phát sinh.
* Tại sao phải phân tán đơn đặt hàng, phiếu nhập – xuất không nên theo nhân viên mà nên theo kho?
* Nhân viên có thể bị điều đi từ chi nhánh này sang chi nhánh khác – sửa mã chi nhánh của nhân viên => đơn đặt hàng, phiếu nhập – xuất sẽ di chuyển theo sang những chi nhánh khác.

1. **Nguyên tắc truy xuất cơ sở dữ liệu.**

* #Table: ảo cục bộ, người nào tạo người đó dùng.
* ##Table: ảo toàn cục, một người tạo toàn người dùng. Khi người tạo out, thì table biến mất => cùng database.

**CHƯƠNG 3: THIẾT KẾ CƠ SỞ DỮ LIỆU PHÂN TÁN**

1. **Cách thức để phân tán một database thành bao nhiêu mảnh và những mảnh đó đạt tiêu chí gì thì hợp lí?**

* Dựa vào các vị từ để cho để phân tích thành bao nhiêu mảnh thích hợp.
* Gồm 2 tiêu chí: thỏa tính đầy đủ – khi có nhiều ứng dụng xác xuất truy cập đến các phân mảnh là như nhau và tính cực tiểu – ít nhất phải có 1 ứng dụng chạy trên 1 phân mảnh bất kì.

1. **Trình bày sự cấp phát các phân mảnh?**

* Một phân mảnh trong lượt đồ phân mảnh có thể được ánh xạ qua nhiều site, ánh xạ 1:1 hay 1:n tùy nhu cầu người lập trình.

1. **Mục tiêu của việc thiết kế hệ thống phân tán?**

* Sự truy xuất cục bộ: do phân tán nên dữ liệu nhỏ => truy vấn nhanh. Giấu được dữ liệu với user khác trên cùng hệ thống.
* Tính sẵn sàng và khả tín => phụ thuộc vào quan hệ.
* Nhân bản hoàn toàn: cao nhất
* Nhân bản 1 phần: cao.
* Phân hoạch: thấp.
* Sự phân bố tải: chương 4.
* Chi phí lưu trữ: tốn kém.
* Thực tế, sẽ không đạt hoàn toàn các mục tiêu trên.

1. **Các hướng tiếp cận khi phân tán 1 database?**

* Hướng từ trên xuống: được phát triển từ một database.
* Bước 1: có database tập trung.
* Bước 2: theo yêu cầu bài toán sẽ thiết kế thành những mảnh thích hợp.
* Bước 3: cấp phát phân mảnh đến các site, tạo ảnh vật lí của chúng.
* Hướng từ dưới lên: được phát triển dựa trên tổ hợp các database sẵn có.
* Bước 1: nghiên cứu tổ hợp batabase, tìm ra mô hình chung để mô tả được tất cả lược đồ phổ quát của database riêng.
* Bước 2: Đổ dữ liệu của các database riêng vào mô hinh database chung.
* Bước 3: tiếp tục bước 1 của hướng từ trên xuống.

1. **Tính chất thiết lập phân mảnh ngang nguyên thủy?**

* Vị từ đơn giản: thuộc tính = giá trị
* Vị từ sơ cấp: chuẩn hội của tất cả vị từ đơn giản
* Một phân mảnh ứng với một vị từ sơ cấp
* Khi tạo một phân mảnh dựa trên điều kiện ta đưa ra phải thỏa được tính thích hợp – phải có ứng dụng truy cập tới.

1. **Ưu – nhược điểm:**

*Khi cho về site chủ lấy dữ liệu:*

* Ưu điểm:
* Có sẵn số liệu (luôn luôn hoạt động).
* Nhược điểm:
* Lộ những dữ liệu nhạy cảm giữa các phân mảnh khác.

*Khi không cho về site chủ lấy dữ liệu:*

* Ưu điểm:
* Không lộ dữ liệu nhạy cảm.
* Nhược điểm:
* Muốn lấy số liệu phải nhờ hỗ trợ từ các site khác. Nếu site ở trạng thái stop sẽ không lấy được.

1. **Thế nào là cấp phát phân mảnh?**

* Khi tạo ra 1 lượt đồ phân mảnh gồm nhiều phân mảnh, sau khi đẩy qua các site phân mảnh, nếu:
* Ánh xạ 1:1 cơ chế đồng bộ dữ liệu nhanh, tiết kiệm chi phí lưu trữ nhưng không an toàn – khi máy tính gặp vấn đề thì dữ liệu sẽ không chạy được.
* Ánh xạ 1:n cơ chế đồng bộ dữ liệu chậm, tốn chi phí lưu trữ nhưng an toàn dữ liệu – khi máy ở 1 site gặp vấn đề, ta có thể chuyển sang site khác.

**CHƯƠNG 4: TỐI ƯU HÓA TRUY VẤN**

* Giảm thời gian, giảm bộ nhớ.
* Dùng phép chọn – chiếu trước, phép kết sau.
* Khử phép kết nếu được
* AND xác xuất sai cao để đầu, OR ngược lại
* Field tham gia trong mệnh đề quan hệ nên được sắp xếp thứ tự trước và thứ tự này phải được sử dụng trong mệnh đề truy vấn với mệnh đề WITH(Index\_Tên Index)

**CHƯƠNG 5: GIAO TÁC PHÂN TÁN**

* Khái niệm: giao tác được xem như là một dãy các thao tác đọc (select) – ghi (insert – update – delete – merge) trên cơ sở dữ liệu cùng với các bước tính toán cần thiết (Begin Trans, Commit, Rollback, Begin Distributed Trans – thêm bên phân tán) để đảm bảo tập lệnh như một đơn vị lệnh tối thiểu.
* Quản lí giao tác là giải quyết các bài toán duy trì được trên cơ sở dữ liệu ở trong tình trạng nhất quán ngay cả khi có nhiều truy xuất đồng thời và khi có sự cố – khi giao tác xảy ra lỗi thì bỏ qua hết những lệnh chạy thành công.
* Giao tác gồm 2 loại:
* Giao tác tập trung: giao tác phẳng và giao tác lồng.
* Giao tác phẳng: chỉ có một điểm bắt đầu (Begin trans) và kết thúc (End trans)
* Giao tác lồng: gồm nhiều giao tác khác với điểm bắt đầu và kết thúc của riêng nó. Khi sp có 3 giao tác A B C khi A chạy thành công nhưng B lỗi – hủy, A vẫn giữ nguyên kết quả và hủy luôn C.
* Giao tác phân tán: chỉ có giao tác phẳng.
* Tính chất giao tác:
* Tính nguyên tử: một giao tác có nhiều lệnh được xem như một đơn vị lệnh (thành công thì thành công hết hoặc không)
* Tính nhất quán: đảm bảo tất cả các ràng buộc về mặt giao tác.
* Tính biệt lập: khi một giao tác đang thực thi không lộ ra các kết quả của nó cho những giao tác khác đang cùng hoạt động trước khi nó ủy thác.
* Read Uncommited
* **Read Commited – mức biệt lập mặc định của SQL**
* Repeatable Read
* Serializable
* Snapshot
* Tính bền vững: đãm bảo rằng khi một giao tác commit thì kết quả của nó sẽ duy trì cố định và không bị xóa khỏi database
* Dữ liệu rác: giá trị dữ liệu được cập nhật nhưng chưa xác nhận – khi đang chạy giao tác.
* Dữ liệu rác gồm 3 mức độ:
* Giao tác T thỏa mức độ 3 – cao nhất: không ghi đè lên – đọc dữ liệu rác của giao tác khác, không cho phép giao tác khác lấy dữ liệu về, những giao tác khác không làm cho dữ liệu T đã đọc – trước khi T hoàn thành – trở thành dữ liệu rác.

1. **Khi ta viết một SP trong đó có cập nhật dữ liệu, vậy trường hợp nào không cần dùng Begin trans – commit – rollback nhưng SP đó vẫn là một giao tác?**

* Khi SP chỉ có duy nhất 1 lệnh cập nhật duy nhất – giao tác tự động.
* @@TRANCOUNT: biến hệ thống, trả về số giao tác đang hoạt động trên kết nối.
* Lệnh Begin trans làm cho @@TRANCOUNT tăng lên 1.
* Lệnh Rollback trans làm cho @@TRANCOUNT trả về 0, nếu dùng Rollback trans savepoint\_name (sẽ có giao tác lồng) thì sẽ không ảnh hưởng @@TRANCOUNT.
* Lệnh Commit trans sẽ giảm đi 1.
* @@ROWCOUNT có bao nhiêu mẫu tin chịu sự tác động của DML gần nhất.

1. **Sự khác nhau giữa giao tác phân tán tập trung & giao tác phân tán?**

* Giao tác phân tán: phân bố ở nhiều nơi. Khi 2 giao tác chạy ở 2 server nếu 1 giao tác gặp sự cố thì giao tác kia vẫn chạy bình thường
* Giao tác tập trung: tập trung chỉ ở 1 nơi. Khi giao tác gặp sự cố thì tất cả thao tác đều dừng.
* Giao tác phân tán:
* Service MS DTC – trình điều phối các giao tác phân tán, nó đảm bảo rằng hoặc là tất cả các cập nhật dữ liệu trên tất cả các server là được thực hiện, hoặc là trong trường hợp có lỗi thì xem như chưa thực hiện thao tác gì trên giao tác đó.

**THỰC HÀNH**

1. **Làm sao máy tính biết được ta có bao nhiêu phân mảnh? Phân mảnh đó được đẩy về những site nào?**

* Dựa vào 2 table hệ thống Sysmergepublications và Sysmergesubscriptions

1. **SP lấy thông tin nhân viên từ logins**

* SELECT \* FROM SYS.syslogins – MASTER(không cần thiết). Vì == SUSER\_SID(@TENLOGIN) (không cần thiết)
* SELECT \* FROM SYS.sysusers -- DB NGƯỜI DÙNG
* SELECT \* FROM SYS.sysmembers -- DB NGƯỜI DÙNG

1. **Combobox có những thuộc tính nào quan trọng?**

* DataSource: thuộc tính chứa danh sách các dữ liệu cung cấp cho nó
* DisplayMenber: thuộc tính chứa giá trị mình chọn trên màn hình là thuộc tính Text
* ValueMember: thuộc tính khi ta chọn chi nhánh nào trong tên chi nhánh, nó sẽ trả về giá trị tương ứng với chi nhánh đó
* SelectedIndexChanged sẽ được sinh ra khi ta chọn giá trị mới trên combobox

1. **Tại sao SelectedIndexChanged được gọi như vậy?**

* Vì nó cũng là một thuộc tính trong combobox và được bắt đầu từ số 0. Khi chọn giá trị mới thì thuộc tính này thay đổi, khi thay đổi như vậy ta sẽ lấy giá trị cùng dòng với chi nhánh đang chọn là TÊN SERVER - là SelectedValue do ValueMember chứa

1. **Ưu nhược điểm của của kiểu trả về dưới dạng bảng DataReader và bảng DataTable?**

* DataReader sẽ tải về nhanh hơn nhưng dữ liệu không thể nào sửa được, chỉ đi tiếp không đi ngược lại.
* DataTable ngược lại.

1. **Khi ta muốn thực thi một câu lệnh sp hay function hay view thì ta làm cách nào?**

* Ta sẽ tạo ra một đối tượng command và ta sẽ nhúng vào trong đối tượng command này 2 tham số dưới dạng phương thức thiết lập tham số thứ 1 là CHUỖI LỆNH, tham số thứ 2 là ĐƯỜNG KẾT NỐI, sau đó gán đối tượng command bằng kiểu câu lệnh dạng chuỗi, sau đó gọi lệnh ExecuteReader nhận kết quả là đối tượng datareader.

1. **Thế nào là cơ sở dữ liệu phân tán?**

* Tập hợp dữ liệu có mối quan hệ tương quan luận lí với nhau nhưng nằm rải rác ở nhiều site.
* Kiến trúc cơ sở dữ liệu phân tán theo database gốc, theo yêu cầu bài toán sẽ tạo thành những phân mảnh thích hợp - tập hợp những phân mảnh này được gọi là lượt đồ phân mảnh. Với 1 phân mảnh, sẽ đẩy vào những site thích hợp - tập hợp những site chứa ảnh dữ liệu đã được định nghĩa trong lượt đồ phân mảnh được gọi là lượt đồ cấp phát.
* Kiến trúc cơ sở dữ liệu phân tán theo quan hệ: một quan hệ trên database gốc được đẩy qua nhiều phân mảnh theo nhiều cách: nhân bản hoàn toàn, nhân bản 1 phần, phân hoạch.

1. **Hệ quản trị như thế nào thì phân tán được?**

* Gồm 4 thành phần: quản trị database, truyền dữ liệu qua mạng, định nghĩa được từ điển dữ liệu, PHÂN TÁN.

1. **Nếu phân tán xong database, muốn thay đổi cấu trúc cột server gốc thì làm như thế nào?**

* Viết SP dùng lệnh alter table ở server gốc sau đó đồng bộ xuống server phân mảnh.

1. **Trong lược đồ phân mảnh, số lượng db trong publication và số lượng db trong subcriber như thế nào ?**

* Số lượng db trong publication nhiều hơn vì trong publication chứ nhiều subcription

1. **Dữ liệu sau khi nhập form sẽ được đẩy về đâu?**

* Đẩy về publisher sau đó đồng bộ xuống các subcriber

1. **Loginname nằm trong table nào?**

* Nằm trong table sys.sysuser trong DB đó

1. **Tại sao biết user liên kết với login nào?**

* Username và loginname liên kết với nhau qua trường sid (trên user và login đều có sid) nên từ loginname biết được username từ sid

1. **Tại sao khi gõ login lại biết họ tên?**

* Tại vì login và user liên hệ với nhau qua trường sid trong sys.systemlogin và sys.systemuser

1. **Hàm suy từ loginname -->username: SUSER.SID(ten login):**-->sid --> username --> userid --> groupid --> rolename
2. **Tên nhóm quyền nằm trong table nào?**

* Sys.sysuser

1. **Rowguid** : do hệ thống tự sinh ra, dùng để đồng bộ dữ liệu xuống phân mảnh, nó chỉ rõ table nào, cột nào để đưa dữ liệu xuống đúng vị trí đó
2. **Trong để tài QLVT, tại sao vật tư lại nhân bản?**

* Vì vật tư có thể có ở chi nhánh này cũng có thể có ở chi nhánh khác

1. **Đưa ra 2 cách xử lý truy vấn dữ liệu cùng lúc ở 2 server khác nhau?**

* Dùng câu truy vấn song song cho nó chạy song song ở 2 server khác nhau.

1. **Subform là gi?**

* Là form có ít nhất 2 bảng và có sự liên kết với nhau.

1. **Public static SqlConnection conn = new SqlConnection():** Biến này dùng để chạy các SP và View, trong SQL nó như là cầu nối và chạy xuyên suốt trong chương trình.
2. **ConnectionString**: thuộc tính quan trọng nhất vì chỉ khi mở kết nối kiểm tra xem chuỗi đó đúng hay sai
3. **LinkServer dùng để làm gì? Khi tạo LinkServer cần điều kiện gì? Muốn truy xuất đến phân mảnh gần nhất đầu tiên chúng ta phải tạo LinkServer như thế nào?**

* LinkServer: dùng để truy xuất dữ liệu các subcriber
* Điều kiện: LinkServer cùng tên link và cùng tên cơ sở dữ liêu.
* Phụ thuộc vào cách đặt tên link sao cho phù hợp (ví dụ: gần đặt 1, xa đặt 2)