**PHẦN I: PHÂN TÁN**

**Note:**

* MSDTC hỗ trợ tự động rollback dữ liệu khi sử dụng giao tác phân tán nên không cần rollback trong sp.
* Dấu hiệu để biết đó là giao tác phân tán : BEGIN DISTRIBUTED TRANSACTION.
* Khi XACT\_ABORT được đặt thành ON, mọi lỗi gây ra trong giao dịch sẽ chấm dứt giao dịch ngay lập tức và quay trở lại điểm bắt đầu. Nếu bạn đã sử dụng SET XACT\_ABORT ON, không cần thiết phải sử dụng ROLLBACK TRAN trong quá trình xử lý lỗi. Tuy nhiên, nếu bạn muốn đảm bảo rõ ràng và tường minh về việc hủy bỏ giao dịch trong code của mình, bạn vẫn có thể sử dụng ROLLBACK TRAN, nhưng nó không cần thiết từ góc nhìn của SET XACT\_ABORT ON.

**Câu 1: Các bước phân tán? Khi phân tán cần bật dịch vụ gì? Cách bật như thế nào?**

* Các bước phân tán:

+ Định nghĩa:

* Publisher (server gốc): chứa bản gốc của csdl
* Distributor: điều phối các mảnh phân tán
* Subscriber: chứa csdl sau khi phân tán

+ 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 Subscription: 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 phải bật dịch vụ: SQL Server Agent
* Cách bật: run gõ services.msc-> start SQL Server Agent
* Lệnh select...union : dùng để lấy dữ liệu từ các server phân mảnh về.   
  Ví dụ : **SELECT** MALOP,TENLOP FROM DBO.LOP   
  **UNION** SELECT MALOP,TENLOP FROM LINK1.QLDSV.DBO.LOP

**Câu 1.1a: Khi xây dựng cơ sở dữ liệu phân tán phải đảm bảo 2 tiêu chí.**

+) Tiêu chí cực tiểu: một phân mảnh khi sinh ra thì có ít nhất một ứng dụng sẽ sử dụng tới.

+) Tiêu chí đầy đủ: xác suất để một ứng dụng sử dụng tới các phân mảnh là như nhau.

**Câu 1.1b: Nếu đã phân tán xong cơ sở dữ liệu, muốn thay đổi cấu trúc cột của server gốc thì làm ntn?**

* Có thể thay đổi bằng cách viết sp dùng lệnh ALTER TABLE ở server gốc sau đó đồng bộ xuống các server phân mảnh

**Câu 1.2: Dữ liệu sau khi nhập form sẽ được đẩy về đâu?**

* Đẩy về publisher sau đó đồng bộ xuống các subscriber

**Câu 2: thuộc tính folder RELPDATA là gì? ở đâu? Mục đích của thư mục để làm gì?**

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

**PHẦN II: SỰ TRONG SUỐT PHÂN TÁN**

**Câu 1: Các loại phân mảnh?**

* **Phân mảnh ngang (phép hội union)**: là chia (phân rã) 1 quan hệ theo các bộ. Mỗi phân mảnh ngang là một tập con của quan hệ. Có hai loại phân mảnh ngang đó là phân mảnh ngang dẫn xuất và phân mảnh ngang nguyên thủy.
  + Phân mảnh ngang nguyên thủy là sự phân rã dựa trên một quan hệ vị từ được định nghĩa trên một quan hệ. VD: TENKHOA=”CNTT”
    - Vị từ đơn gian: ví dụ như ChiNhanh = ‘BENTHANH’
    - Vị từ sơ cấp: là nhiều vị từ đơn giản hợp với nhau: ChiNhanh = ‘BENTHANH’ OR ChiNhanh =’TANDINH’ Hay là ChiNhanh = ‘BENTHANH’ AND SODU > 1000000
  + Phân mảnh ngang dẫn xuất là sự phân rã một quan hệ dựa vào các vị từ được định nghĩa trên một quan hệ khác. VD: SINHVIEN.MALOP = LOP.MALOP
* Phân mảnh dọc (phép kết tự nhiên có khóa chính) : Phân mảnh dựa trên khóa chính của một quan hệ {phải có khóa chính để đảm bảo tính tái thiết}.
* Phân mảnh hỗn hợp : vừa ngang vừa dọc
* Điều kiện để phân tán được : phải có dịch vụ SQL Server Agent

**Câu 2: Nhân bản là gì? Lí do nhân bản? Nhân bản không hiệu quả với lệnh gì?**

* Lý do : truy xuất nhanh, tính sẵn sàng cao
* Không hiệu quả: update, delete, insert
* **Trong database, cái nào là nhân bản, cái nào là phân hoạch?** (mở cây dẫn xuất ra xem, cái nào không có trong cây dẫn xuất là nhân bản, cái đầu tiên là phân mảnh ngang nguyên thủy). VD: Môn học là nhân bản, còn lại là phân hoạch.

**Câu 3: Phân hoạch là gì?**

* Phân hoạch là dữ liệu không nhân bản và nó chỉ tồn tại một bản sao duy nhất cho mỗi phân mảnh.

**Câu 4: Nêu rõ các 3 quy tắc phân mảnh: Tính tái thiết, tính đầy đủ, tính tách biệt**

* Tính tái thiết (khi một cơ sở dữ liệu chia thành nhiều phân mảnh thì khi hợp các phân mảnh lại ta sẽ có phân mảnh gốc): thỏa tính đầy đủ thì thỏa tính tái thiết. Định nghĩa 1 phép toán để tái thiết dữ liệu, ngang là phép hội (union), dọc là phép kết tự nhiên trên khóa chính (join-on).
* Tính đầy đủ: Dữ liệu nằm ít nhất ở 1 mảnh (đặc tính này nói lên sự phân mảnh không mất thông tin)
* Tính tách biệt: dữ liệu chỉ nằm duy nhất ở 1 mảnh (tính chất này có thể vi phạm được, table nào nhân bản thì vi phạm)

**Câu 4.1: Trong các table, cái nào mang tính đầy đủ, cái nào vi phạm tính tách biệt?**

* Các table nhân bản thì vi phạm tính tách biệt. Các table còn lại thì mang tính đầy đủ.

**Câu 5: Sp trong suốt là gì? Điều kiện để trong suốt? Có mấy mức trong suốt, nêu rõ các mức trong suốt đó?**

* SP trong suốt là SP khi chạy trên tất cả các phân mảnh phải cho ra các kết quả như nhau.
* Điều kiện để trong suốt:
  + Cùng tên LINKSERVER ở các server.
  + Cùng tên cơ sở dữ liệu (database) ở các phân mảnh.
* **Trong suốt:** khi ng sử dụng thực hiện truy vấn thì k có cảm giác dữ liệu đang phân tán hay tập trung
* Có 4 mức trong suốt phân tán

+) Trong suốt phân tán : không cần chỉ ra đường dẫn, nơi cấp phát phân mảnh. Người dùng cảm giác như đang thao tác trên cơ sở dữ liệu tập trung

+) Trong suốt vị trí: cần chỉ rõ tên table cần truy vấn.

+) Trong suốt ánh xạ cục bộ: cần chỉ rõ tên phân mảnh và vị trí cấp phát

+) Không trong suốt: cần phải chỉ rõ tên database cần thao tác

+) Người lập trình đứng ở mức 3 & làm sao cho tạo ra mức 1.

**Câu 5.1: Trong suốt phân tán, có bao nhiêu mức trong suốt, người lập trình đứng ở mức nào?**

* 4 mức, người lập trình đứng ở mức 3 nhưng làm cho người dùng cảm thấy như đang ở mức 1.
* Trong suốt phân mảnh: không cần chỉ rõ phân mảnh cũng như vị trí cấp phát cho phân mảnh.
* Trong suốt vị trí: cung cấp tên phân mảnh nhưng không cần chỉ ra vị trí phân mảnh
* Trong suốt ánh xạ cục bộ: cung cấp tên phân mảnh và vị trí cấp phát
* Không trong suốt

**Câu 6: 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.
* Mục đích tạo LinkServer: cho phép truy cập dữ liệu từ server này sang server khác (Khi Database nằm ở 2 server khác nhau).
* 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).
* Cách tạo: ví dụ tạo link ở server 1 để truy xuất dữ liệu ở server 2
  + Đứng ở server 2 tạo login HTKN
  + Đứng ở server 1 tạo LINK
  + Số link: n(n – 1) n là số server phân mảnh
  + Số tên link: n – 1
  + Cú pháp sử dụng: TENLINK.TENDATABASE.DBO.TENBANGCANTRUYXUAT

**PHẦN IV: Tối ưu hóa truy vấn**

**Câu 1: Cách tối ưu hóa truy vấn**

Có 5 cách để tối ưu hóa truy vấn

+) Chọn trước, chiếu sau

+) Khử phép kết

+) Mệnh đề AND nào sai thì để ở đầu. Mệnh đề OR thì để ở cuối

+) Một điều kiện WHERE xuất hiện nhiều lần thì dùng các phép biến đổi tương đương để rút gọn 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)

**—----------------------------------------------------------------------------**

* Thực hiện phép chọn, chiếu trước, phép kết sau.
* Tìm cách khử phép kết (nếu được), khử kết bằng cách thay phép kết bằng phép truy vấn lồng.
* Nếu một điều kiện xuất hiện nhiều lần trong where thì dùng các biến đổi tương đương để làm cho điều kiện xuất hiện một lần.
* Mệnh đề AND điều kiện nào có xác xuất sai cao đặt ở đầu.
* Mệnh đề OR điều kiện nào có xác suất đúng cao đặt ở đầu.
* Chiếu : Select , chọn :from , điều kiện lọc : where , kết : join- on – where ( đkiện kết)
* Ví dụ:
  + Khi chưa khử phép kết: lấy ra tên sinh viên và tên lớp của sinh viên thuộc lớp “CN04” :  
    select SINHVIEN.TEN, LOP.TENLOP from SINHVIEN join LOP on SINHVIEN.MALOP=LOP.MALOP where LOP.MALOP = “CN04”
  + Khi đã khử kết: dùng truy vấn lồng: select TEN (select TENLOP from LOP where MALOP=”CN04”) as tenlop from SINHVIEN where MALOP = “CN04”
* **Các mức cô lập dữ liệu:**

+) Read Uncommitted - mức yếu nhất - TT1 đang sửa đổi & TT2 có thể vào xem dữ liệu ngay cả khi TT1 chưa lưu thay đổi lại. Nói nôm na là “tôi không cần biết dữ liệu có đang được cập nhật hay không, hãy cho tôi dữ liệu hiện có ngay tại thời điểm này”.

+) Read Commited - mức mặc định - TT1 đang sửa đổi dữ liệu thì TT2 không thể xem dữ liệu này. Chỉ khi TT1 xong thì TT2 mới xem được dữ liệu. Nếu TT2 sửa dữ liệu thì TT1 sẽ có được dữ liệu được sửa đổi từ TT2.

+) Repeatable Read - mức an toàn - TT2 đang đọc một bảng dữ liệu thì TT1 không thể chỉnh sửa bảng dữ liệu này cho đến khi TT2 hoàn tất việc đọc dữ liệu đó. Nói nôm na là dữ liệu đang được đọc sẽ được bảo vệ khỏi cập nhật bởi các transaction khác

+) Serializable - mức an toàn cao - TT1 đang sửa đổi một bảng dữ liệu thì TT2 không thể làm bất cứ hành động gì với bảng dữ liệu đó như: INSERT, UPDATE, DELETE,.....

+) Snapshot - mức an toàn cao nhất - hoạt động tương tự Serializable, TT1 vẫn hoạt động, TT2 sẽ sửa dữ liệu trên một bản ghi đã được Snapshot sao chép.

**Phần V: Giao tác phân tán**

**Câu 1:** Giao tác phân tán là gì? Các loại giao tác? Sự giống nhau và khác nhau giữa hai giao tác này? Sự giống nhau và khác nhau của giao tác phân tán và giao tác tập trung? Khi viết cần bật dịch vụ gì? Cách bật như thế nào?

* **Giao tác** là: 1 dãy các thao tác đọc ghi trên cơ sở dữ liệu cùng với các phép toán cần thiết: thêm, xóa, sửa có tính nhất quán để giải quyết các tình huống khi dữ liệu bị mất tính nhất quán khi có nhiều truy xuất đồng thời.
* **Giao tác phân tán** là: tập các lệnh đọc ghi và các bước tính toán cần thiết lên cơ sở dữ liệu và có thêm khả năng tái thiết dữ liệu (nếu không có khả năng tái thiết thì giao tác phân tán là một SP thuần túy).
* **Giao tác** thì thực thi trên môi trường cơ sở dữ liệu tập trung (gồm có giao tác phẳng và giao tác lồng), còn **giao tác phân tán** thì thực thi trên môi trường CSDLPT. Giống nhau thì nêu 4 tính chất giao tác.
* Các loại giao tác: **Giao tác phẳng** (chỉ có một lệnh BEGIN TRANSACTION và một lệnh COMMIT TRANSACTION) và **giao tác lồng** (có thể có nhiều lệnh BEGIN và COMMIT lồng vào nhau).
* Sự khác nhau giữa **giao tác phân tán** và **giao tác tập trung**

+ **Giao tác phân tán**: Giao tác phân tán hoạt động trên môi trường phân tán. Chúng có thể chạy đồng thời ở các phân mảnh được. Chỉ có một loại là giao tác phẳng

+ **Giao tác tập trung**: Giao tác tập trung là giao tác hoạt động trên môi trường tập trung. Nếu xảy ra sự cố thì mọi hành động đều dừng. (gồm 2 loại là giao tác phẳng và lồng)

* **Khi viết giao tác cần** bật dịch vụ MSDTC Microsoft DISTRIBUTE TRANSATION COODINATOR.
* Vào run gõ services.msc => start Distribute Transaction Coodinator.

**Câu 2: Nêu rõ 4 tính chất giao tác: nguyên tử, nhất quán, biệt lập, bên vững? ý nghĩa của các tính chất?**

* **Tính nguyên tử** (tính chất quan trọng nhất của giao tác): mọi lệnh sẽ được thực hiện thành công hoặc nếu một lệnh thất bại thì tất cả thất bại.
* **Tính nhất quán**: khi thực hiện các lệnh insert, update, delete nếu thành công thì tất cả dữ liệu phải được cập nhật hết, còn nếu bị lỗi thì dữ liệu phải trở về vị trí ban đầu.
* **Tính biệt lập**: khi một giao tác đang hoạt động thì nó sẽ không để lộ kết quả cho giao tác khác.

Ví dụ đang delete thì không thể add, đợi delete xong mới được add, giải quyết vấn đề dữ liệu rác

* **Tính bền vững**: một giao tác khi được commit thì kết quả của nó vẫn được duy trì cố định và không bị xóa ra khỏi csdl.

**Câu 3: Giải thích cách lệnh BEGIN TRANSACTION, COMMIT TRANSACTION, @@TRANCOUNT, DEADLOCK, ROLLBACK, ABORT?**

* @@TRANCOUT: số giao tác đang được thực hiện trên giao tác hiện tại.
* BEGIN TRANSACTION: bắt đầu một thao tác, tăng @@TRANCOUNT thêm 1
* COMMIT TRANSACTION: giảm 1 trên @@TRANCOUNT
* ROLLBACK: đưa @@TRANCOUNT về 0
* Một cv B gọi A: Nhưng A bị lỗi nên cv B cứ đợi A-> gọi là DeadLock
* Cách xử lý DeadLock: thiết lập Timeout , sau bao lâu đó thì trở về trạng thái ban đầu

**Câu 4: Dịch vụ MS Distributed Transaction Coordinator là gì?**

* Là một trình quản lí điều phối các giao tác phân tán, nó cho phép các ứng dụng của client thao tác lên dữ liệu của các datasource trong một giao tác phân tác.
* MS DTC service điều phối sự đúng đắn của một giao tác phân tán, nó bảo đảm rằng hoặc là tất cả được cập nhật trên các server là được thực hiện, hoặc trong trường hợp lỗi thif xem như chưa thực hiện giao tác gì trên giao tác đó.
* Lưu ý: Nếu MSDCT không hoạt động thì cách khắc phục như sau:
  + Vào cmd gõ msdtc.exe-install
  + Sau đó start service Distribute Distributed Coordinator

Nếu không khởi động được thì làm một trong hai cách sau:

* + - Cách 1: Re-create msdtc log for DTC service
      * Vào cmd gõ msdtc -resetlog
      * Gõ tiếp net start msdtc
    - Cách 2: set the logon account permission for msdtc

**Phần VI : FORM**

**Câu 1: Các thuộc tính của ComboBox là gì? Ý nghĩa của nó?**

* BindingSource: chỉ ra dữ liệu mà ComboBox đó nằm ở đâu.
* DisplayMemBer: Dữ liệu hiển thị để người dùng chọn. Dữ liệu có thể là một cột trong DataTable
* ValueMember: là giá trị tương ứng với displaymember. Dữ liệu có thể là một cột trong DataTable

**Câu 2: Tại sao đăng nhập ở form đăng nhập thì mình có thể đăng nhập thành công được? khi đăng nhập thành công , tại sao biết được mã, họ tên , nhóm của người đăng nhập**

**Câu 3: “Device” trong BACKUP DATABASE QLVR TO DEVICE\_QLVT là gì?**

* Device là tên thiết bị dùng để lưu trữ database khi backup và restore, đường dẫn: tên server /Server Object/Backup Device/DEVICE\_QLVT.

Đặt tên có Device để phân biệt đây là 1 colum, không phải là một DB

**Câu 4: Khi backup db, thì bản sao lưu đó sẽ được lưu trữ ở đâu?**

* Được lưu ở table backupset, đường dẫn tên server.Databases/SystemDatabase/msdb/Tables/System Table/backupset
* Có 2 dạng backup là update là override
  + Dạng UPDATE: backup bản này nối tiếp bản kia
  + Dạng OVERRIDE: khởi tạo lại bản backup

**Câu 5: Client và server liên kết, trao đổi dữ liệu bằng cách nào?**

* Thông qua DataAdapter

**Câu 6: BindingSource là gì?**

* Là nơi chỉ ra (Binding) để thao tác dữ liệu, là cầu nối giữa DataGridView và DataTable.

**Câu 7: Nút lệnh thêm, cái gì gọi AddNew();**

* BindingSource

**Câu 8: Để update dữ liệu và db, trước khi gọi lệnh Update thì cần phải thực hiện lệnh gì?**

* EndEdit();
* ResetCurrentItem();

**Câu 9: Loginname nằm trong table nào?**

* Nằm trong table sys.sysuser trong Database đó.

**Câu 10: 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.

**Câu 11:** **Hàm suy từ loginname -->username:** SUSER.SID(ten login) -->sid --> username --> userid --> groupid --> rolename

**Câu 12: Tên nhóm quyền nằm trong table nào?** Sys.sysuser

**Câu 13: DataSet, DataTable, DataAdapter là gì?**

* DataSet có thể hiểu là một bản ghi tạm, nó lưu trữ và chỉnh sửa dữ liệu ở Local Cache, tức là khi có ngắt kết nối tới server đi nữa thì vẫn thao tác với dữ liệu trên DataSet. Sau khi xem sét và chính sửa dữ liệu xong ta tạo kết nối (DataAdapter) để Update dữ liệu từ Local(DataSet) về DataSource(SQL)
* Dữ liệu trong DataSet được lưu ở dạng DataTable(DT). DT là một đối tượng chứa dữ liệu từ DataBase gởi qua, cấu trúc gồm có các DataRow và DataColumn. DT có thể là 1 Table, 1 View, hoặc 1 SP, chứa dữ liệu nhưng không chứa lệnh.
* DataAdapter(DA) là một dạng cầu nối, giúp các lệnh như Select, Insert, Update, Delete để trao đổi dữ liệu giữa Client và Server, và DA đổ dữ liệu từ DataSource về DS bằng phương thức Fill, GetData

**Câu 14: Các lệnh EndEdit, ResetCurrentItem sử dụng trong bài có ý nghĩa gì?**

* AddNew() thêm một item/row vào danh sách,
* CancelEdit() bỏ qua các dữ liệu đang chỉnh sửa trong rơ và tả con trỏ về row cuối cùng. Chỉ có tác dụng với lệnh AddNew và dữ liệu đang được hiệu chỉnh nhưng chưa ghi vào DataSet.
* EndEdit: kết thúc quá trình chỉnh sửa dữ liệu và gửi vào DataSet.
* ResetCurrentItem: Reread/ đọc lại row hiện tại ta đang đứng, dữ liệu hiện tại ở row ta đang đứng không còn ở dạng tạm nữa(tức là dữ liệu có thể cancel edit) mà hiển thị dữ liệu đã chính thức ghi vào DataSet
* Update cập nhật data từ DataSet về DB thông qua TableAdapter.
* Count(): đếm xem số record tương ứng với row đang chọn tồn tại bao nhiêu bảng trong bindingsource được kiểm tra.
* RemoveCurrent Xóa row đang chọn ra khỏi DataSet

**Câu 15: Sau khi phân tán có Rowguid để làm gì?**

* 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.

**Câu 16: Ưu và khuyết điểm của nhân bản, phân hoạch?**

* **Ưu khuyết điểm của nhân bản:** truy xuất nhanh, đứng ở đâu cũng có thể select được. Nhưng update chậm vì có quá nhiều bản sao.
* **Ưu khuyết điểm của phân hoạch:** select chậm nhưng insert và update nhanh do chỉ thao tác trên server gốc hoặc server phân mảnh cần truy xuất.

**Câu 17: Tại sao table đó nhân bản?**

* Trong đề vật tư thì vật tư nhân bản: vì vật tư có thể có ở chi nhánh này cũng có thể có ở chi nhánh khác.
* Trong đề trắc nghiệm thì giáo viên nhân bản vì giáo viên có thể dạy ở cở sở này cũng có thể dạy ở cơ sở khác, bộ đề đi theo giáo viên nên cũng nhân bản. Giáo viên cơ sở 1 có thể dạy ở cơ sở 2 nhưng không thể đăng kí thi cho lớp ở cơ sở 2.

**Câu 18: Trong lược đồ phân mảnh, số lượng database trong publication và số lượng database trong subscriber như thế nào?**

* Số lượng database trong publication nhiều hơn trong subscription vì trong publication chứa nhiều subscription.

**Câu 19: 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.

**Câu 20:** **Tại sao biết user liên kết với login nào?**

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

**Câu 21: Chuyển chữ thường thành chữ hoa:** dùng lệnh upper

**Câu 22: 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

**Câu 23: Subform là gì?**

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

**Câu 24: Câu lệnh bắt đầu & kết thúc của 1 giao tác là gì?**

Begin distributed Transaction >> bắt đầu

Commit transaction >> Kết thúc

Bắt lỗi : Try (commit transaction) catch (Rollback transaction)

- **PGV**: đc quyền xem cả 2 khoa, và chỉ đc: một tài khoản duy nhất ở 1 phân mảnh nhưng đc quyền xem 2 phân mảnh (PGV)

**- 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.

**-ConnectionString**: thuộc tính quan trọng nhất vì chỉ khi mở kết nối kiểm tra xem chuỗi đó đúng hay sai

-**Lệnh Fill** : là lệnh tự động tải dữ liệu về

**-DropDownList:** trong cmb chỉ cho phép chọn, k đc ghi

**-Liên kết dữ liệu giữa 2 form**: biến thì nằm ở program, mà khi chạy thì nó nằm ở đăng nhập.

**-SelectedIndexChanged**: biến cố này tự động chạy khi ta chọn 1 giá trị mới trong cmb và khi nó chạy lấy ra servername nằm trong program.

-**DataSet**: là 1 đối tượng có thể chứa nhiều DataTable.

-**DataTable** dùng để lưu trữ 1 tập dữ liệu dạng bảng đc trích ra từ DataSource

-**DataTable khác với Table:** (1sp là 1 dataTable) là 1 đối tượng mà dữ liệu của nó là 1 phần của table.

**- Rowguid** : Tự sinh ra để đồng bộ hóa dữ liệu.

**Câu 25:** Có mấy loại Join: 3 loại

+ INNER JOIN

+ OUTER JOIN : Left Join & Right Join

+ CROSS JOIN: ko có khóa nó gộp lại nhưng vẫn lấy

**INNER JOIN** trả về các bản ghi có giá trị phù hợp giữa hai bảng

**LEFT JOIN** mọi bản ghi bảng bên trái trả về kết hợp với bản ghi phù hợp với bên phải nếu có (nếu không có thì nhận NULL)

**RIGHT JOIN** mọi bản ghi bảng bên phải trả về có kết hợp với giá trị phù hợp nếu có ở bảng trái

**OUTER JOIN** (full join) mọi bản ghi ở bảng trái và bảng phải kết hợp lại

**Câu 26: Đơn giản hóa 1 b/thức đại số quan hệ được t/hiện dựa trên các tiêu chuẩn :**

+ Tiêu chuẩn 1: Sử dụng phép biến đổi tương đương, phép chọn và chiếu.

+ Tiêu chuẩn 2:Thực hiện các phép càng xa càng tốt, tức là đẩy các phép chọn và các phép chiếu xuống phía dưới càng xa càng tốt.

+ Tiêu chuẩn 3: Khi các phép chọn đc thực hiện sau 1 phép tích thì kết hợp các phép toán này để tạo thành 1 phép kết.

+ Tiêu chuẩn 4: Kết hợp chuỗi các phép toán 1 ngôi liên tiếp nhau áp dụng cho 1 quan hệ toán hạng.

+ Tiêu chuẩn 5: Khi phát hiện các biểu thức con truy vấn, áp dụng các phép biến đổi tương đương để đơn giản hóa biểu thức truy vấn.

* Truy vấn (query) là 1 biểu thức đc biểu diễn bằng 1 ngôn ngữ và dùng để xác định 1 phần dữ liệu đc chứa trong CSDL.
* Phân quyền:

Server 1+2: PGV : datareader, writer, securityadmin,access admin

KHOA: datareader, writer

USER: datareader, writer

Server 3: PKTOAN: accessadmin, reader, writer, securityadmin

- db\_accessadmin: ng dùng có quyền quản lý tài khoản SQL server đăng nhập

- db\_securityadmin: ng dùng có thể chỉnh sửa vai trò (role) và quản lý các bậc quản lý phân quyền khác

+ muốn tạo tài khoản thì dùng accessadmin + securityadmin ( cấp quyền)

- db\_datareader: ng dùng có thể đọc được toàn bộ dữ liệu.

- db\_datawrite: ng dùng có quyền thêm, xóa, sửa dữ liệu trong bảng

- db\_owner: toàn bộ ng dùng có quyền full-access

- sysAdmin: toàn bộ người dùng đều có thể t/hiện các thao tác trên server.

- Có 3 loại server

+Local server: cài máy nào thì chạy máy đó

+Remote server: điều khiển từ xa giữa 2 máy

+ Link server : Liên kết server này với server khác.

-Dùng link cơ chế nối từ sql server đến database của server khác SQL server hay hệ thống khác

-Tạo link thì tạo remotelogin(HTKN) trước

**Câu 27: Vị từ là gì?**

* Thực chất là phép chọn trên 1 field trên 1 giá trị const

**Câu 28: Dữ liệu rác là gì?**

* Là dữ liệu chưa được ủy thác

**Câu 29: Các phân quyền?**

sysadmin: có toàn quyền truy cập vào SQL Server và có thể thực hiện mọi tác vụ.

serveradmin: có thể quản lý cấu hình SQL Server và tài khoản đăng nhập.

securityadmin: có thể quản lý phân quyền và đăng nhập.

processadmin: có thể quản lý các tiến trình của SQL Server.

setupadmin: có thể cài đặt và cấu hình SQL Server.

bulkadmin: có thể thực hiện các tác vụ liên quan đến dữ liệu lớn, chẳng hạn như import và export dữ liệu.

dbcreator: có thể tạo cơ sở dữ liệu mới.

diskadmin: có thể quản lý các tập tin dữ liệu trên đĩa.

datareader: có thể đọc dữ liệu từ các bảng trong cơ sở dữ liệu.

datawriter: có thể thêm, sửa và xóa dữ liệu từ các bảng trong cơ sở dữ liệu.