HỌC VIỆN QUÂN SỰ

KHOA CÔNG NGHỆ THÔNG TIN

Trần Đăng Công

GIÁO TRÌNH

QUẢN TRỊ VÀ PHÁT TRIỂN ỨNG DỤNG VỚI MICROSOFT SQL SERVER

Năm 2006

MỤC LỤC

MỞ ĐẦU

Khi nhu cầu phát triển ứng dụng và quản trị với số lượng bản ghi lớn, kích thước lớn, nhiều kiểu dữ liệu phức tạp (âm thanh, hình ảnh,…) thì việc đặt ra với các hãng phần mềm là phát triển các hệ quản trị cơ sở dữ liệu lớn. Việc những nhà lập trình phát triển ứng dụng trên hệ quản trị cơ sở dữ liệu lớn cũng đòi hỏi phải có những nắm bắt tích cực về sự phát triển của các hệ quan trị cơ sở dữ liệu.

Trong lịch sử đến này, hệ quản trị cơ dở dữ liệu ta có thể điểm nhanh gồm các hệ sau: Foxpro, Access, MySQL, SQL Server, Oracle,… mỗi hệ quản trị cơ sở dữ liệu đều có những phiên bản, phiên bản sau phát triển tiến bộ hơn, đáp ứng tốt hơn yêu cầu thực tế đặt ra phiên bản trước.

Trong giáo trình này sẽ giới thiệu cho bạn đọc hệ quản trị CSDL (cơ sở dữ liệu) Microsoft SQL Server. SQL Server là hệ quản trị cơ sở dữ liệu lớn do hãng Microsoft phát triển, được cài đặt và chạy trên hệ điều hành Windows, SQL Server tỏ ra khá phổ biến và thân thiện với người dùng thông qua giao diện đồ họa trên Windows. SQL Server phát triển theo các phiên bản 6.0, 6.5, 7.0, 8.0 (phiên bản 2000), 2003, 2005.

Với mục đích giúp cho bạn đọc, đặc biệt là sinh viên đại học chuyên ngành Công nghệ thông tin có thể nắm bắt được những kỹ năng quản trị cơ sở dữ liệu cũng như kỹ thuật xây dựng ứng dụng từ các ngôn ngư lập trình (Visual Basic, Visaul Basic.net, ASP, ASP.net) trên hệ quản trị CSDL SQL Server, giáo trình này sẽ trình bày một cách dễ hiểu, theo hướng phát triển ứng dụng, hệ quản trị CSDL SQL Server 2000.

Phần 1. QUẢN TRỊ SQL SERVER
BẮT ĐẦU VỚI SQL SERVER

TÌM HIỂU VỀ HỆ QUẢN TRỊ CSDL SQL SERVER

Giới thiệu SQL Server.

SQL Server là hệ thống quản trị cơ sở dữ liệu quan hệ (Relational DataBase Management System- RDBMS) sử dụng các lệnh giáo chuyển Transaction-SQL để trao đổi dữ liệu giữa Client Computer và Server Computer.

SQL Server có một số đặc tính sau:

- Cho phép quản trị một hệ CSDL lớn (lên đến vài tega byte), có tốc độ xử lý dữ liệu nhanh đáp ứng yêu cầu về thời gian.
- Cho phép nhiều người cùng khai thác trong một thời điểm đối với một CSDL và toàn bộ quản trị CSDL (lên đến vài chục ngàn user).
- Có hệ thống phân quyền bảo mật tương thích với hệ thống bảo mật của công nghệ NT (Network Technology), tích hợp với hệ thống bảo mật của Windows NT hoặc sử dụng hệ thống bảo vệ độc lập của SQL Server.
- Hỗ trợ trong việc triển khai CSDL phân tán và phát triển ứng dụng trên Internet
- Cho phép lập trình kết nối với nhiều ngôn ngữ lập trình khác dùng xây dựng các ứng dụng đặc thù (Visual Basic, C, C++, ASP, ASP.NET, XML,...).
- Sử dụng câu lệnh truy vấn dữ liệu Transaction-SQL (Access là SQL, Oracle là PL/SQL).

Các ấn bản của SQL Server.

SQL Server có các ấn bản chính sau:

- Enterpise Manager: Là ấn bản đầy đủ của SQL Server có thể chạy trên 32CPU và 64GB RAM. Có các dịch vụ phân tích dữ liệu Analysis Service.
- Standard: Giống như Enterprise nhưng bị hạn chế một số tính năng cao cấp, có thể chạy trên 2CPU, 4GB RAM.

- Personal: Phiên bản này chủ yếu để chạy trên PC, nên có thể chạy trên các hệ điều hành Windows 9x, Windows XP, Windows 2000, Windows 2003…
- Developer: Là phiên bản tương tự như Enterprise nhưng bị giới hạn bởi số user kết nối đến.
- Desktop Engine: Là phiên bản một engine chỉ chạy trên desktop và không có giao diện người dùng (GUI), kích thước CSDL giới hạn bởi 2GB.
- Win CE: Sử dụng cho các ứng dụng chạy trên Windows CE.
- Trial: Phiên bản dùng thử, bị giới hạn bởi thời gian.
- SQL Client: Là phiên bản dành cho máy khách, khi thực hiện khai thác sẽ thực hiện kết nối đến phiên bản SQL Server, phiên bản này cung cấp giao diện GUI khai thác cho người sử dụng.
- SQL Connectivity only: Là phiên bản sử dụng chỉ cho các ứng dụng để kết nối đến SQL Server, phiên bản này không cung cấp công cụ GUI cho người dùng khai thác SQL Server.

Các phiên bản này được cài đặt phụ thộc vào bộ cài đặt mà bạn chọn hoặc lựa chọn khai cài đặt (ví dụ phiên bản Enterprise, Standard, Personal,… bạn phải chọn theo bộ cài đặt, phiên bản SQL Client, Connectivity,… do bạn chọn trong các hộp thoại trong quá trình cài đặt).

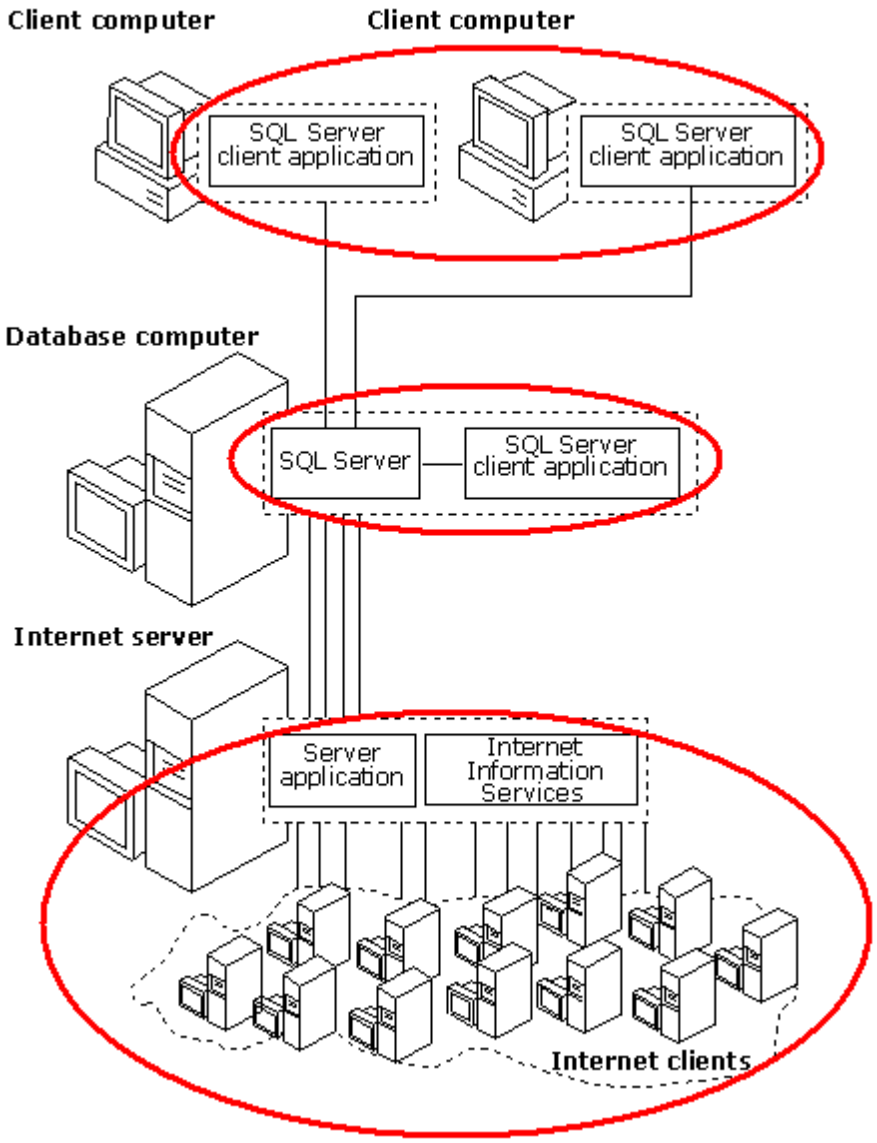
Một số tính năng của Enterprise manager.

- Dễ cài đặt
- Hỗ trợ mô hình Client/Server.
- Thích hợp trên các hệ điều hành Windows.
- Hoạt động với nhiều giao thức truyền thông.
- Hỗ trợ dịch vụ Data Warehousing.
- Thích hợp với chuẩn ANSI/ISO SQL-92.
- Hỗ trợ nhân bản dữ liệu.
- Cung cấp dịch vụ tìm kiếm Full-Text.
- Sách trợ giúp- Book Online.

MÔ HÌNH HOẠT ĐỘNG CỦA SQL SERVER TRÊN MẠNG MÁY TÍNH.

Mô hình chung SQL Server trên mạng.

SQL Server là hệ quản trị CSDL hoạt động trên mạng, có thể thực hiện trao đổi dữ liệu theo nhiều mô hình mạng khác nhau, nhiều giao thức và phương thức truyền tin khác nhau.



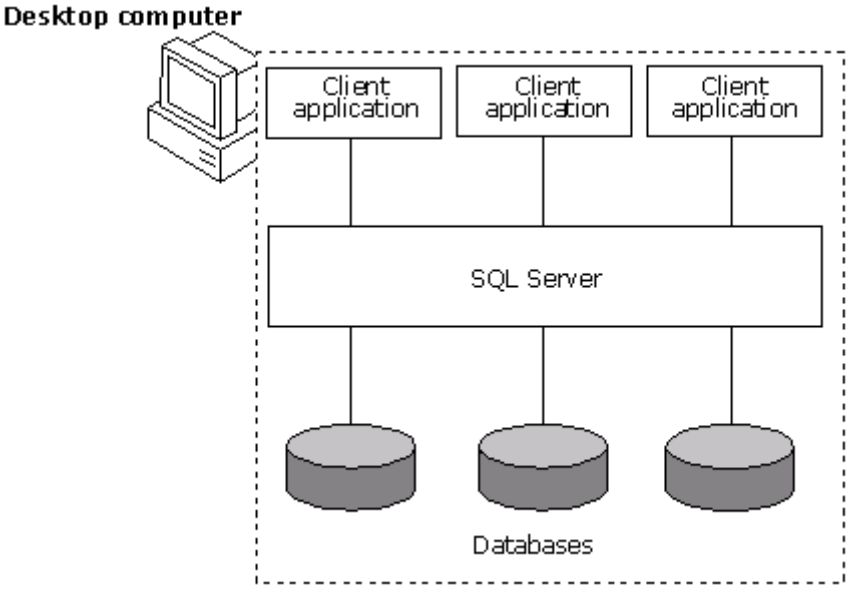
Trong sơ đồ trên thể hiện ba kiểu kết nối ứng dụng đến SQL Server:

- Kết nối trên Desktop: Có thể trên cùng máy tính với SQL Server hoặc kết nối qua mạng nội bộ.

- Kết nối qua mạng diện rộng: Thông qua đường truyền mạng xa kết nối đến SQL Server.
- Kết nối qua mạng Internet: Các ứng dụng kết nối thông qua máy chủ Internet, dịch vụ IIS thực hiện ứng dụng trên Internet (ASP, JSP, ASP.net, …)

Mô hình Desktop.

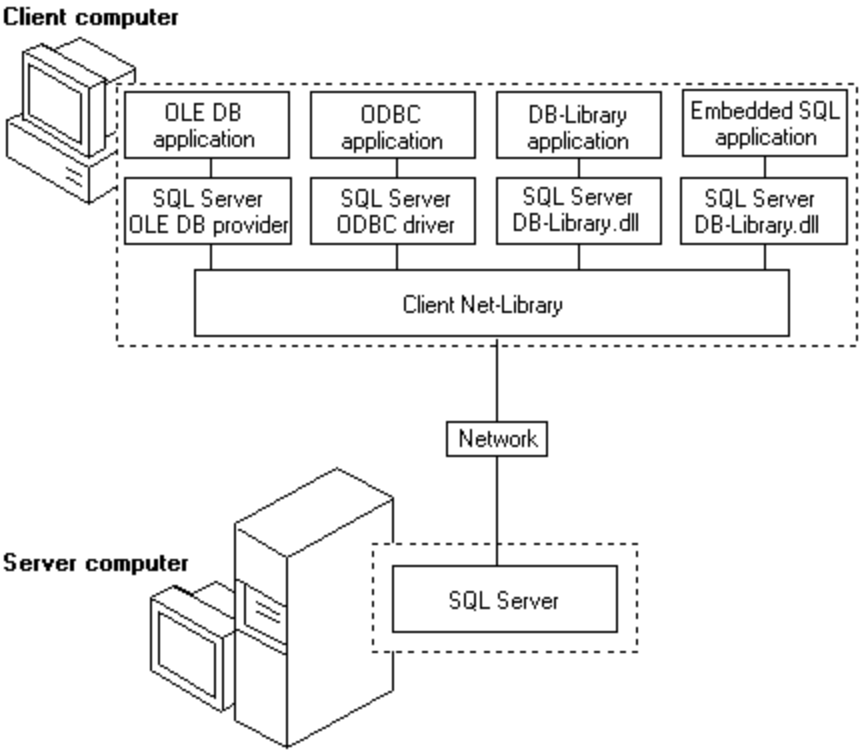
Nếu xét trên một máy Desktop sơ đồ kết nối trao đổi dữ liệu được thể hiện như sau:



Trên một Desktop có thể có nhiều ứng dụng, mỗi ứng dụng có thể thực hiện thao tác với nhiều CSDL.

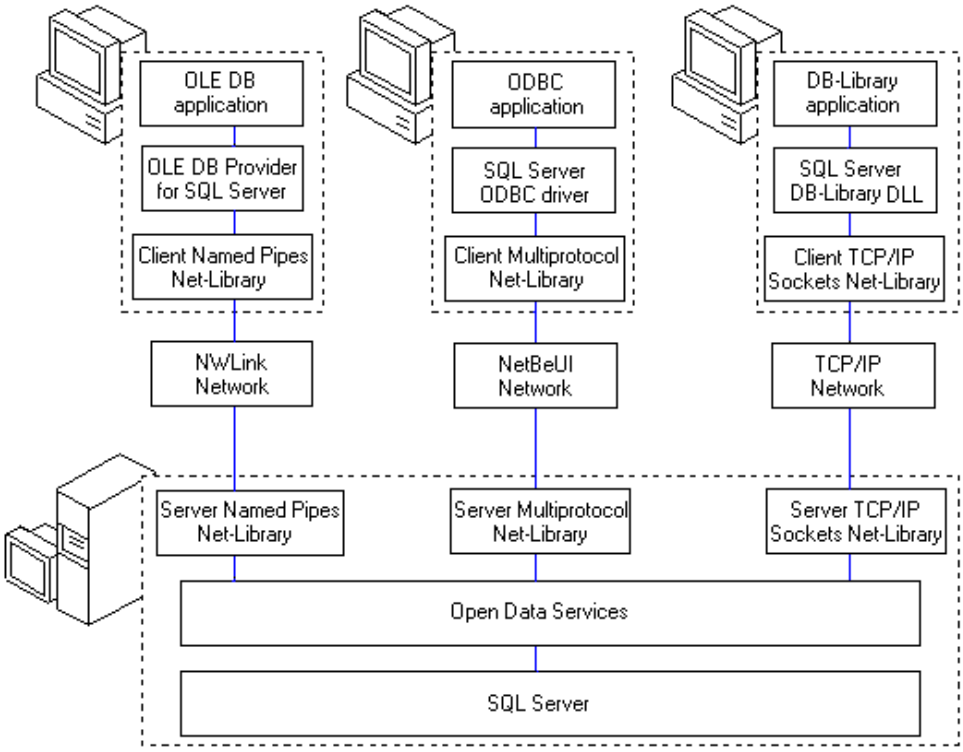
Mô hình Client/Server.

Nếu xét theo mô hình client/server, ứng dụng trao đổi với SQL Server theo sơ đồ sau:



Như sơ đồ trên nhận thấy SQL Server cho phép các ứng dụng kết nối theo các phương thức sau: OLE DB, ODBC, DB-Library, Embedded SQL, đây là các phương thức kết nối hữ ích cho những nhà phát triển ứng dụng.

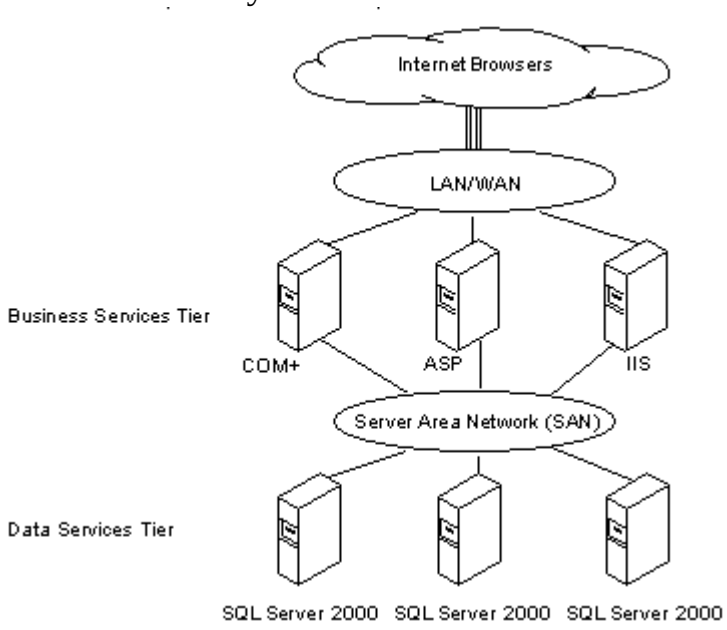
Nếu xem xét cụ thể hơn ta có thể xem sơ đồ sau:



Trong sơ đồ trên cho thấy, SQL Server có thể thực hiện trao đổi dữ liệu với các ứng dụngt heo nhiều giao thực truyền tin khác nhau (TCP/IP, NetBeUI, Names Pipes,…), các ứng dụng có thể sử dụng nhiều phương thức kết nối khác nhau (OLE DB, ODBC, DB-Library).

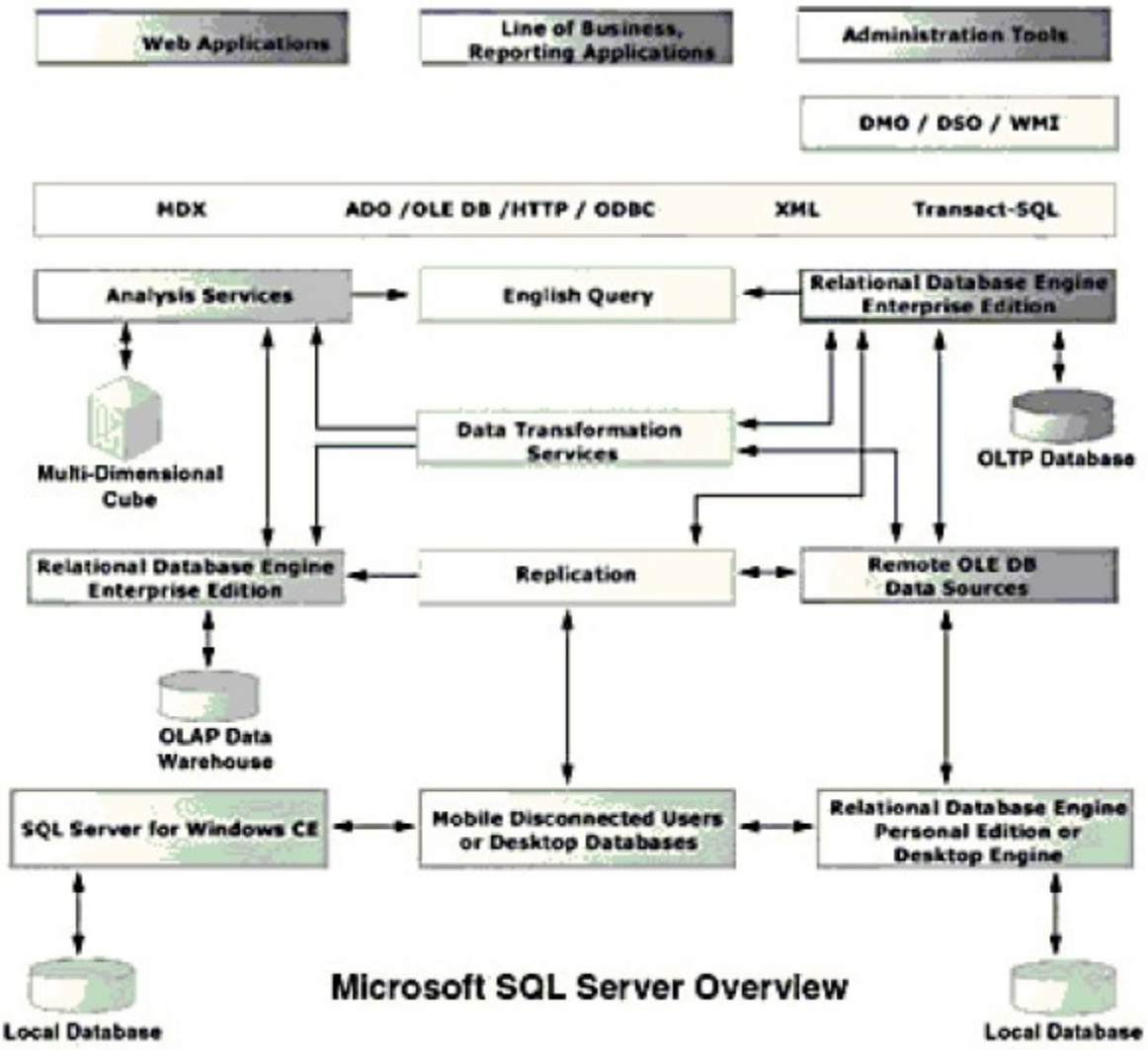
Mô hình kết nối ứng dụng trên mạng Internet.

Nếu xét riêng các ứng dụgn kết nối với SQL Server trên mạng Internet, các máy chủ SQL Server sẽ được quản lý thông qua các hệ thống máy chủ mạng, hệ điều hành mạng, các ứng dụng (COM+, ASP, IIS) sẽ thông qua máy chủ mạng kết nối đến SQL Server, mô hình này có thể áp dụng cho các mạng nội bộ, diện rộng, ứng dụng được khai thác trên trình duyệt Internet Browser. Xem xét mô hình dưới đây:



CÁC THÀNH PHẦN CỦA SQL SERVER.

SQL Server được cấu thành bởi nhiều thành phần khác nhau, các thành phần có mối quan hệ trong một hệ thống, phối hợp với nhau để tạo thành một giải pháp hoàn chính, nâng cao hiệu quả quản trị, phân tích, lưu trữ dữ liệu.



Relational DataBase Engine.

Đây là một engine có khả năng chứa dữ liệu dưới nhiều quy mô khác nhau, theo dạng bảng, hỗ trợ nhiều phương thức kết nối ADO, OLE DB, ODBC.

Replication.

Là công cụ dùng nhân bản dữ liệu, bạn có thể tạo một Server khác với bộ dữ liệu giống bộ dữ liệu trên Server chính. Công cụ tạo cơ chế tự đồng bộ dữ

liệu giữa Server chính và Server nhân bản. Mục đích của việc tạo Server nhân bản là giảm tải cho Server chính, nâng cao hiệu quả phục vụ với số lượng người, phiên giao dịch lớn.

Data Transformation Service – DTS.

Là công cụ giúp bạn chuyển dữ liệu giữa các Server quản trị CSDL khác nhau, DTS có thể chuyển dữ liệu từ SQL Server sang Oracle, Access, DB,… trước khi chuyển dữ liệu DTS định dạng kiểu dữ liệu để chuyển sang hệ quản trị CSDL khác.

Analysis service.

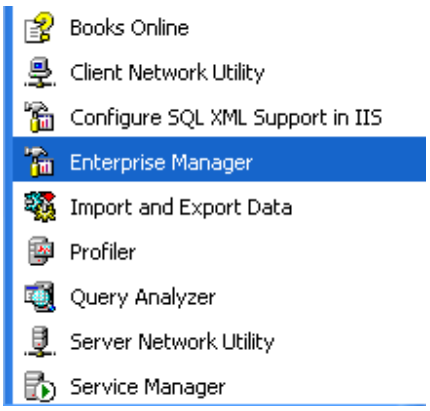
Là công cụ giúp khai thác phân tích dữ liệu, hay khai phá dữ liệu theo phương thức đa chiều. Từ một tập dữ liệu sẵn có bạn có thể khai phá rồi từ đó đưa ra những nhận định, phân tích, đánh giá và dự đoán theo lĩnh vực nào đố, mỗi chiều trong ngữ cảnh này được coi là một tiêu chí xem xét của dữ liệu.

English query.

Đây là công cụ tra cứu dữ liệu bằng tiếng anh, cú pháp có thể sử dụng theo văn phạm tiếng anh thông thường.

SQL Server tools.

Là bộ công cụ cung cấp giao diện cho người quản trị như Enterprise amanger, Query Analyzer ,…SQL Server sau khi cài đặt SQL Server group gồm những thành phần cơ bản trong group như sau:



Một số công cụ quan trong: Enterprise manager, Query Analyzer, Profiler…, các công cụ sẽ được giới thiệu khai thác sau.

CÀI ĐẶT SQL SERVER.

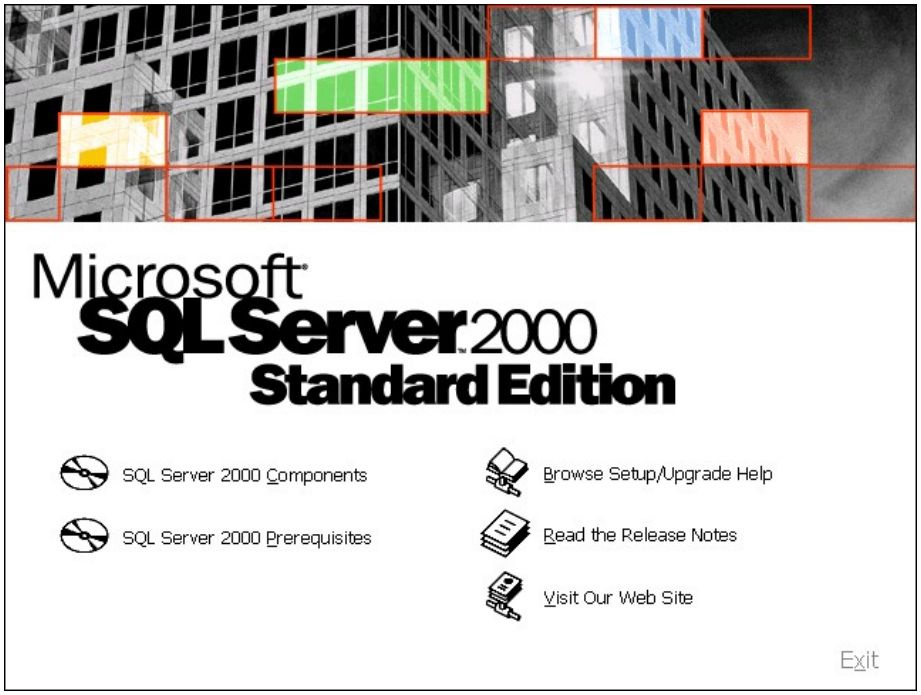
Chuẩn bị cài đặt.

Tùy theo môi trường của máy tính của bạn mà thực hành cài đặt phiên bản nào, bảng sau là tham số với SQL Server 2000 phiên bản Standard.

Computer	Intel® hoặc tương đương Pentium 166 MHz hoặc cao hơn
Memory (RAM)	Enterprise Edition: Tối thiểu 64 MB, 128 MB hoặc nhiều hơn. Standard Edition: Tối thiểu 64 MB. Personal Edition: Tối thiểu 64 MB trên Windows 2000, tối thiểu 32 MB trên các hệ điều hành khác. Developer Edition: Tối thiểu 64 MB. Desktop Engine: Tối thiểu 64 MB trên Windows 2000, tối thiểu 32 MB trên hệ điều hành khác.
Hard disk	SQL Server database components: Từ 95 đến 270 MB, thông thường 250 MB. Analysis Services: Tối thiểu 50 MB, thông thường130 MB. English Query: 80 MB Desktop Engine: 44 MB
Monitor	VGA hoặc độ phân dải cao hơn. 800x600 hoặc độ phân dải cao hơn.

Thực hành cài đặt.

- Sử dụng đĩa CD ROM có bộ cài đặt SQL Server 2000 (tuỳ theo yêu cầu của bạn là Standard, Personal hay Enterprise,...)
- Chạy trình Autorun.exe (thường tự chạy khi đưa đĩa vào máy tính)

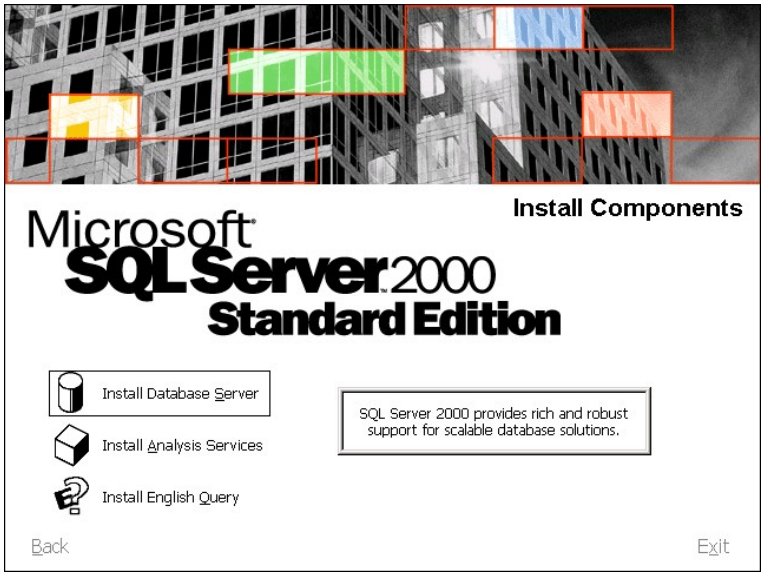


Trong màn hình trên ta có một số lựa chọn:

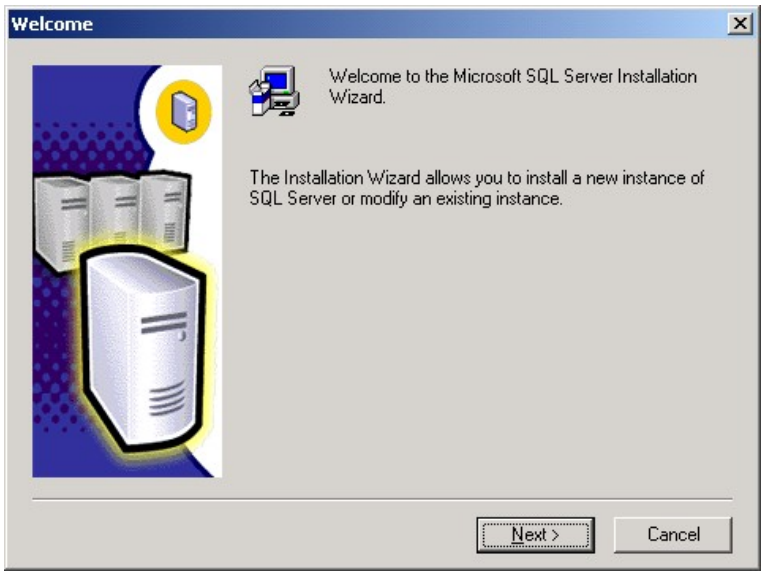
SQL Server Components: Sẽ thực hành trong bước tiếp.

SQL Server 2000 Prerequisites: Dùng cài đặt những yêu cầu được cung cấp sẵn cho việc cài đặt nếu hệ thống trong máy cài đặt chưa đủ.

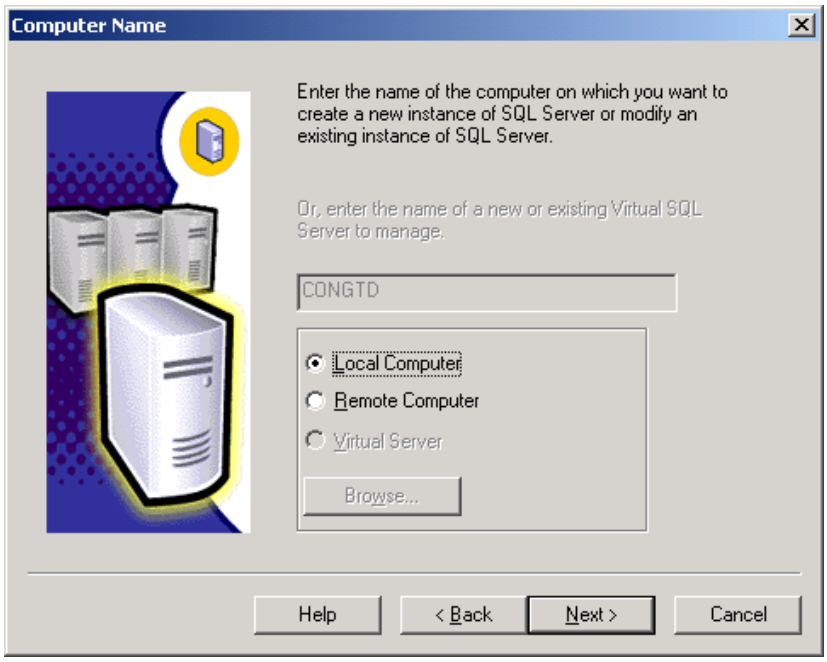
- Chọn SQL Server Components.



- Chọn Install Database Server.



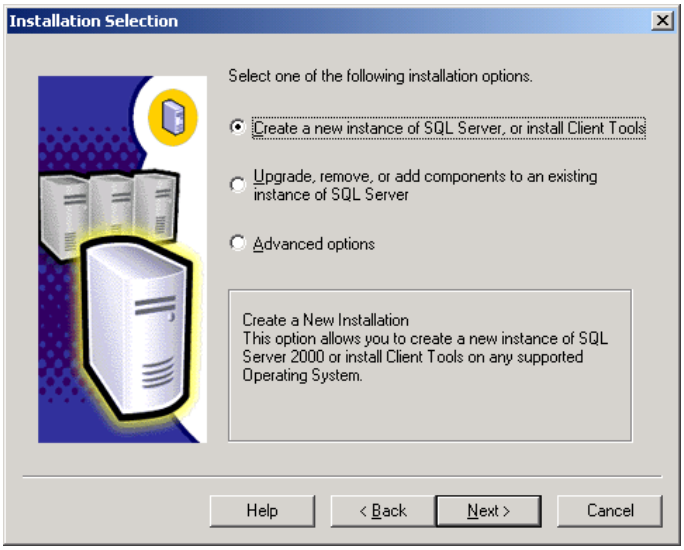
- Chọn Next.



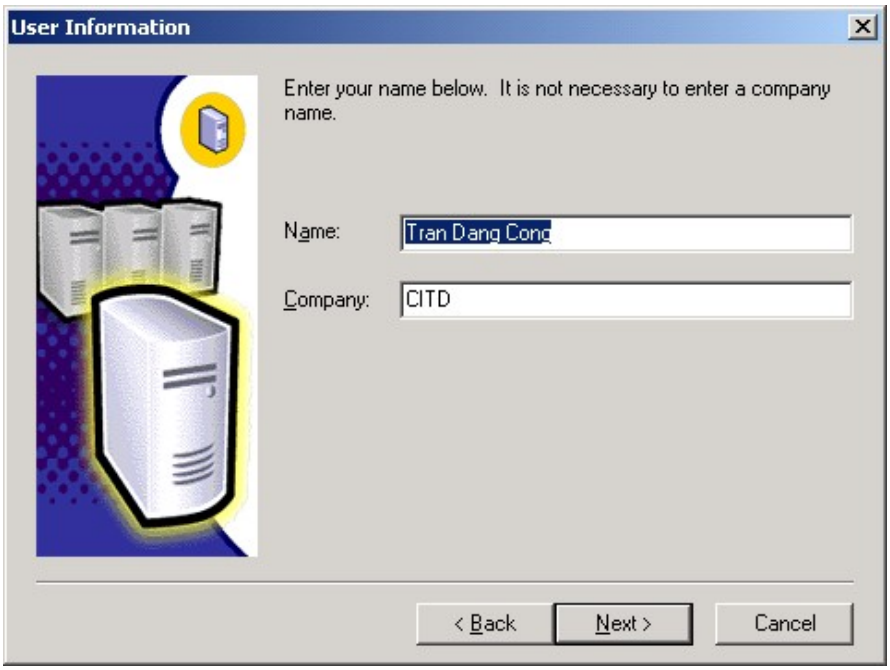
Nếu cài đặt SQL Server trên chính máy bạn đang ngồi thì sử dụng Local Computer

Nếu cài đặt dùng kết nối với máy khác thì sử dụng Remote Computer sau đó nhập tên máy hoặc chọn vị trí máy bằng cách sử dụng Browse

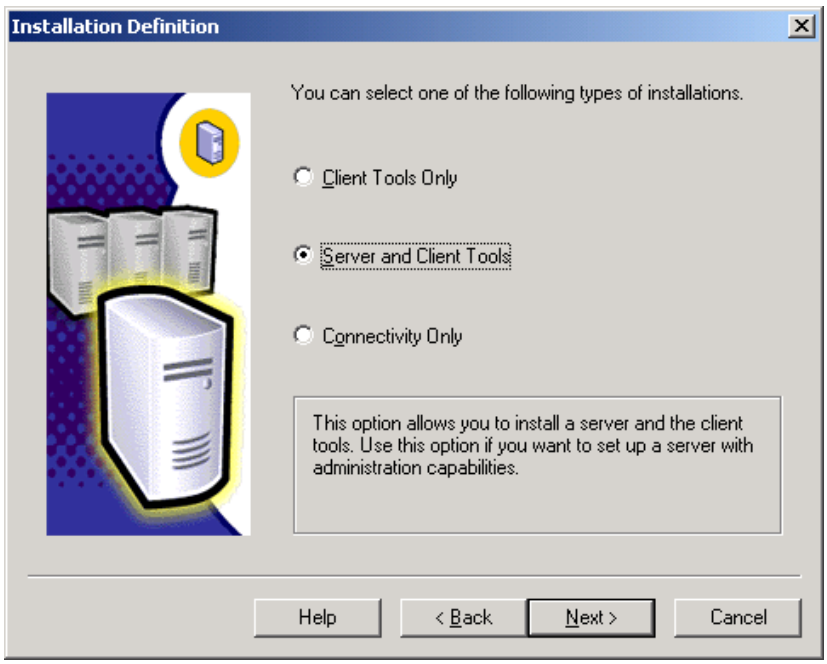
- Chọn next.



- Chọn tuỳ chọn theo chỉ dẫn (tạo mới, thay đổi cái đã có, thêm các chức năng khác,...).
- Trong trường hợp chọn tạo mới (lựa chọn thứ nhất) sau đó ấn Next.



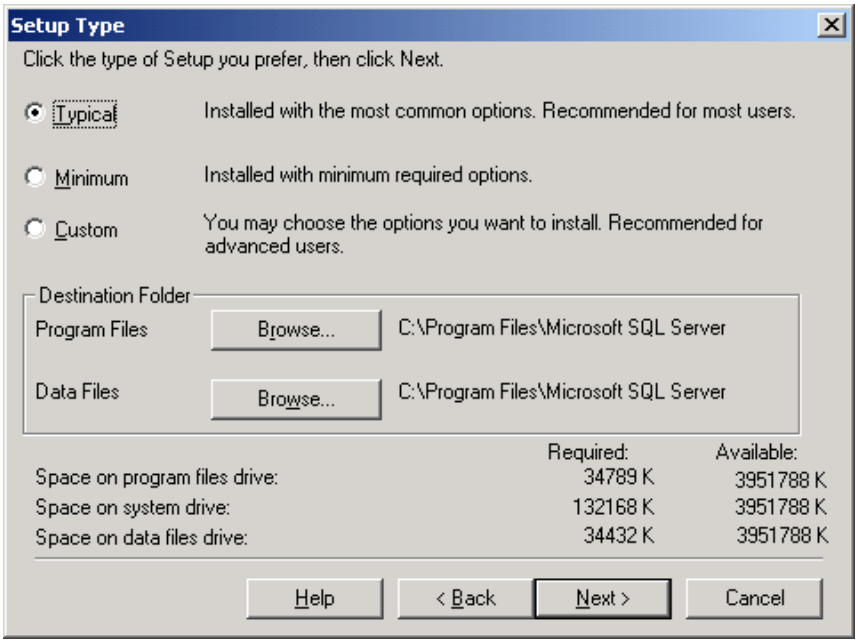
- Nhập tên của bạn, tên cơ quan, sau ấn Next, Yes.



Trong cửa sổ hiện lên 3 lựa chọn:

1. Cài đặt các công cụ truy vấn: Sử dụng cho các máy khách không lưu trữ dữ liệu nhưng có chức năng truy vấn dữ liệu đến SQL Server có CSDL
2. Cài đặt Server và các công cụ truy vấn: Cài đặt SQL Server có dữ liệu và các công cụ của máy khác truy vấn dữ liệu
3. Cài đặt kết nối: Dùng cho các máy chỉ sử dụng kết nối đến Server, thường dùng cài đặt cho các máy sử dụng các ứng dụng kết nối đến server

- Chọn lựa chọn 2, sau ấn Next



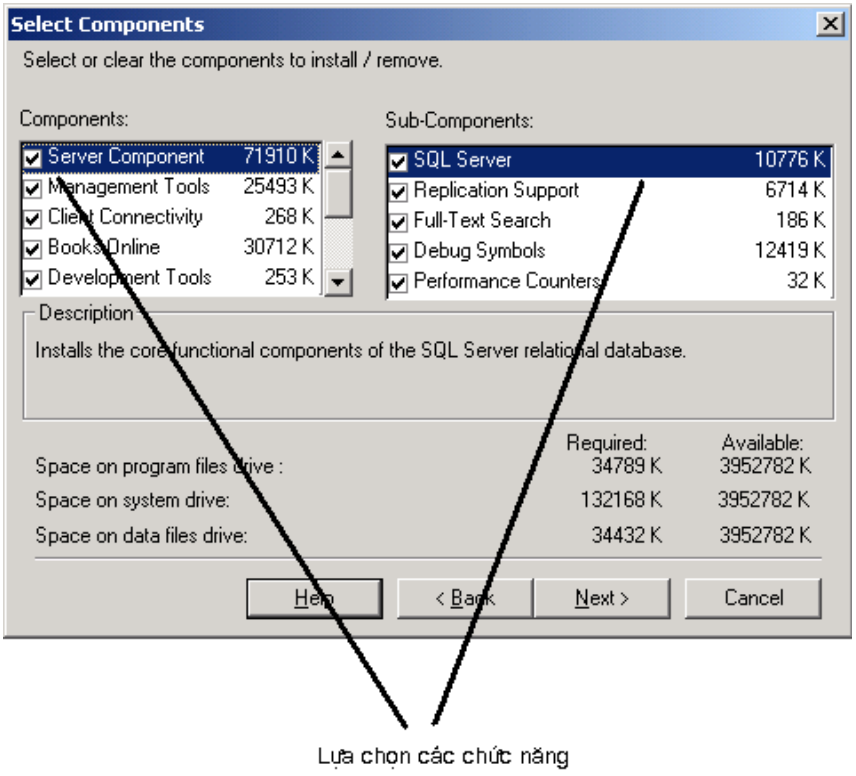
Dùng các lựa chọn để chọn kiểu cài đặt:

+ *Typical*: Cài đặt những chức năng cơ bản được hệ thống định sẵn (chức năng thông thường).

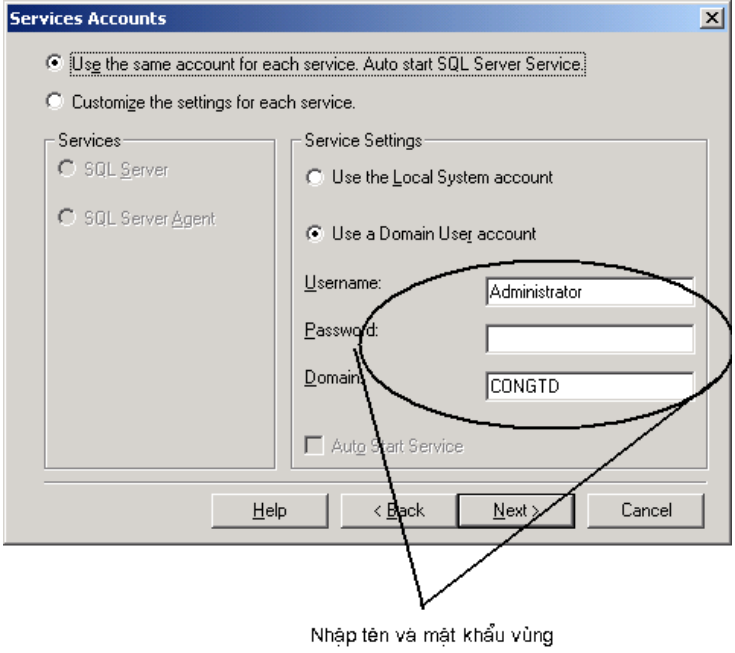
+ *Minimum*: Cài đặt những chức năng tối thiểu của hệ thống.

+ *Custom*: Lựa chọn những chức năng cần cài đặt theo yêu cầu của người dùng.

Trong cách lựa chọn Custom ta cần thêm bước chọn các chức năng như sau:



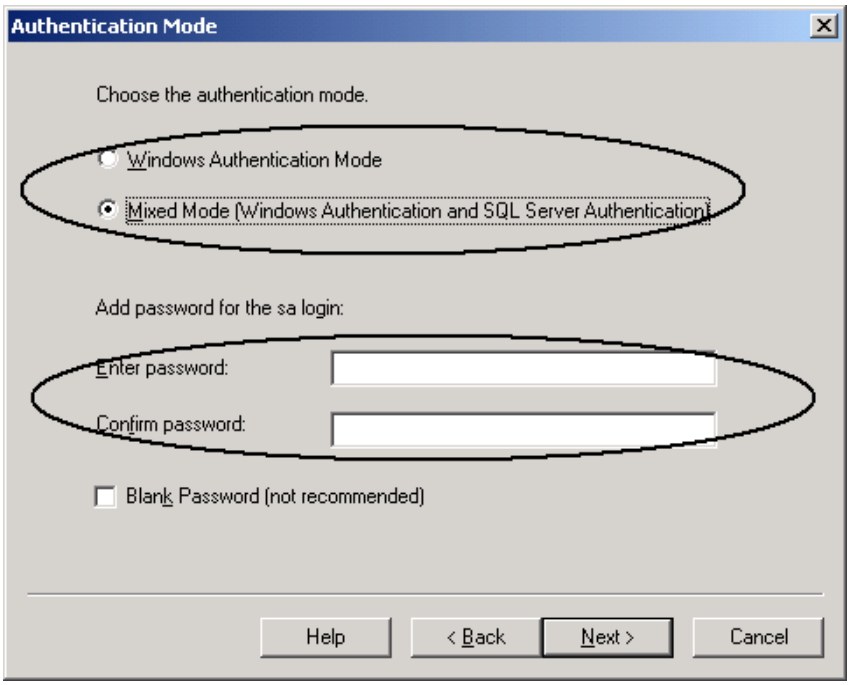
- Ấn nút Next để tiếp tục.



Nhập tên và mật khẩu vùng

Trong cửa sổ trên ta cần nhập tên, mật khẩu của user được đăng ký truy nhập vùng, thường SQL Server cài đặt được thực hiện theo quyền Administrator của máy tính chủ, khi đó bạn lựa chọn nút chọn Use a Domain User account.

- Ấn nút next để tiếp tục.



Trong cửa sổ trên cho phép ta sử dụng 2 lựa chọn:

+ *Lựa chọn thứ nhất*: Người dùng sử dụng hệ thống bảo mật của Windows (hệ điều hành của máy chủ cài đặt – thông thường khi cài đặt dùng lựa chọn này).

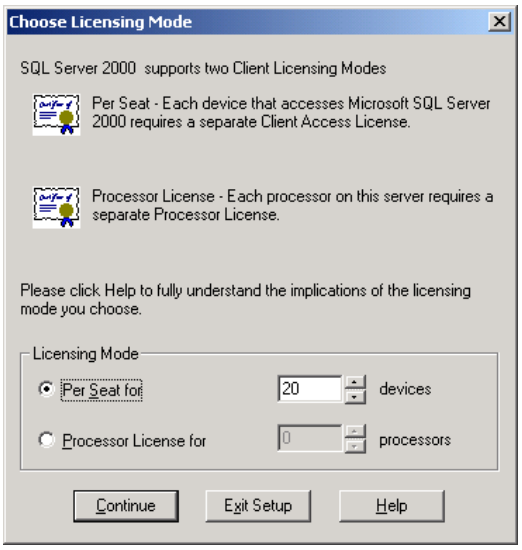
+ *Lựa chọn thứ hai*: Người dùng sử dụng hệ thống bảo mật của Windows và của hệ quản trị CSDL SQL Server.

Trong các trường hợp trên đều có thể sử dụng tên và mật khẩu được cung cấp theo vùng (domain) của hệ điều hành. Nếu sử dụng lựa chọn thứ 2 ta sử dụng tên và mật khẩu của người quản trị vùng (Administrator).

Đối với SQL Server ta có thể thay tên Administrator bằng tên sa (viết tắt của từ System Administrator).

Vấn đề thực hiện chọn chế độ bảo mật nào sẽ được bàn trong những bài sau.

- Ấn next để tiếp tục.

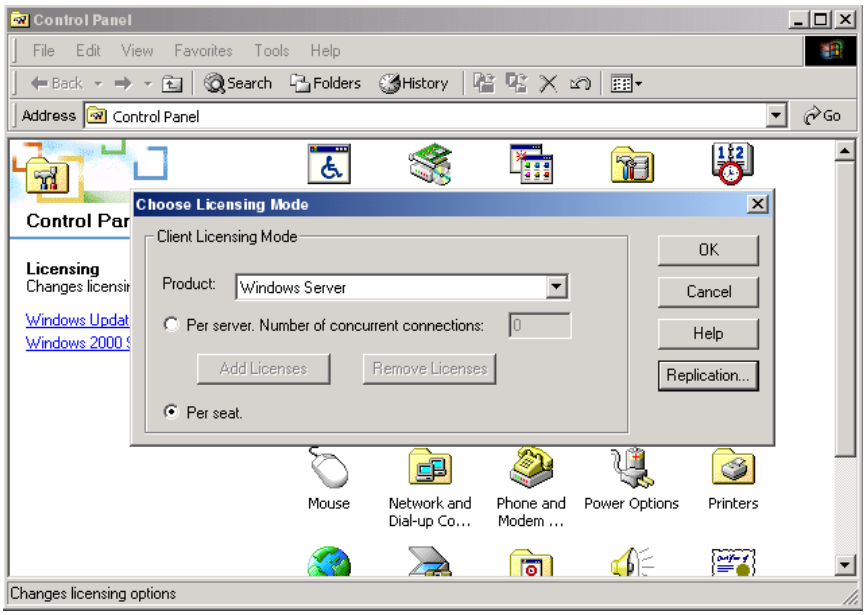


Trong cửa sổ trên ta có 2 lựa chọn:

+ *Per Seat for*: Lựa chọn cho phép xác định số thiết bị (khái niệm sẽ được giới thiệu sau) trên mỗi vị trí khai thác hệ thống theo bản quyền được phép của Microsoft.

+ *Processor License for*: Xác định số Processor cho phép sử dụng theo bản quyền được cung cấp bởi Microsoft.

Ngoài việc đăng kỹ bản quyền tại thời điểm này, ta có thể đăng ký bản quyền trong công cụ điều khiển của Control Panel.



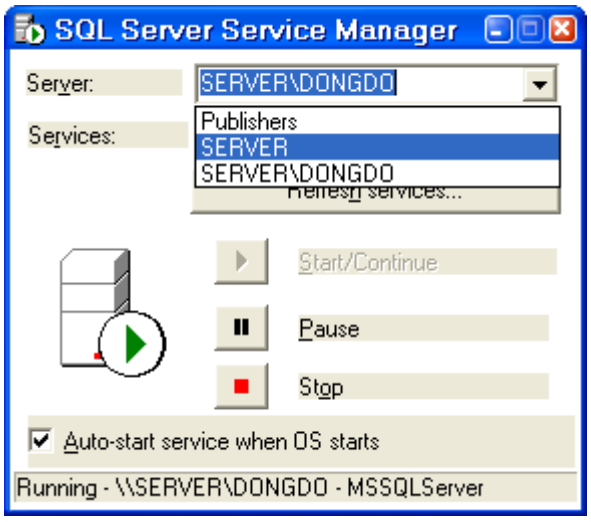
QUẢN TRỊ SERVER

INSTANCE

SQL Server hỗ trợ nhiều hoạt động trên mạng, như các mô hình đã xem xét trước ta có thể thiết lập nhiều máy tính cài đặt SQL Server, các máy tính có thể liên kết với nhau, trao đổi dữ liệu với nhau.

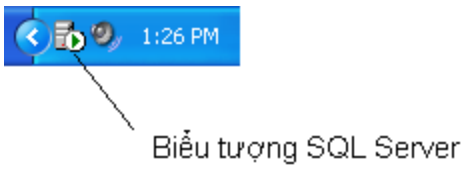
Tuy nhiên một máy tính cũng có thể thiết lập nhiều hệ thống SQL Server khác nhau, mỗi hệ thống đều có một tên quy định, mỗi hệ thống như vậy gọi là một Instance.

Mỗi Instance trên một máy tính được coi như một hệ thống SQL Server độc lập, tương tự như các hệ thống SQL Server cài đặt trên các máy tính khác nhau.

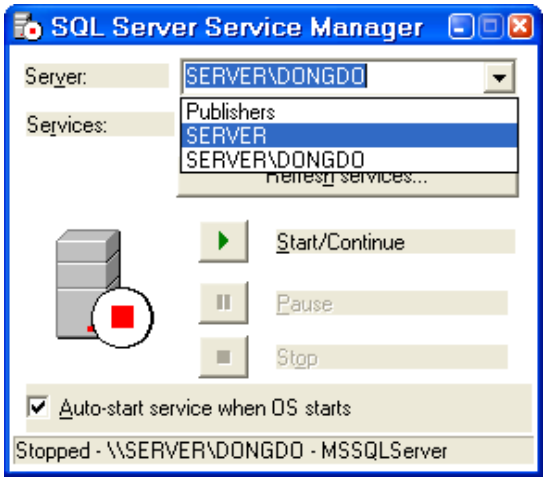


ĐIỀU KHIỂN CÁC DỊCH VỤ CỦA SQL SERVER.

SQL Server sau khi cài đặt xong, khởi động máy thông thường sẽ được thiết lập có biểu tượng ở góc dưới, trái màn hình như sau:



Biểu tượng mày chỉ có với máy tính cài đặt phiên bản SQL Server và là biểu tượng của trình quản lý dịch vụ Service Manager.



Gồm các dịch vụ cơ bản sau:

+ Distributed Transaction Coordinator - DTC.

+ Microsoft Search.

+ SQL Server.

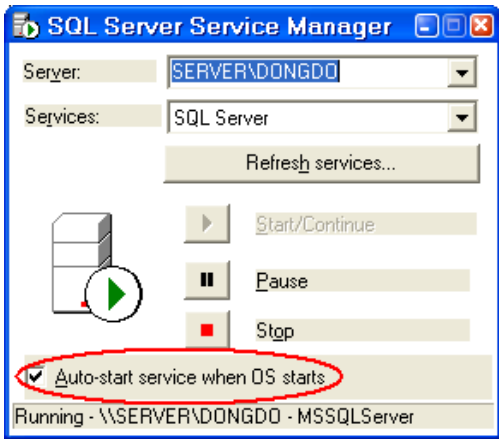
+ SQL Server Agent.

Các dịch vụ này ta có thể bắt đầu, tạm dựng hoặc kết thúc, mỗi dịch vụ đều điều khiển các ứng dụng, công cụ quản trị của SQL Server.

Để thực hiện điều khiển dịch vụ đầu tiên ta làm như sau:

Services -> Start/Continue (Pause, Stop)

Để dịch vụ khởi động tự động khi khởi động hệ điều hành hãy chọn vào nút chọn ***Auto-start service when OS starts***.

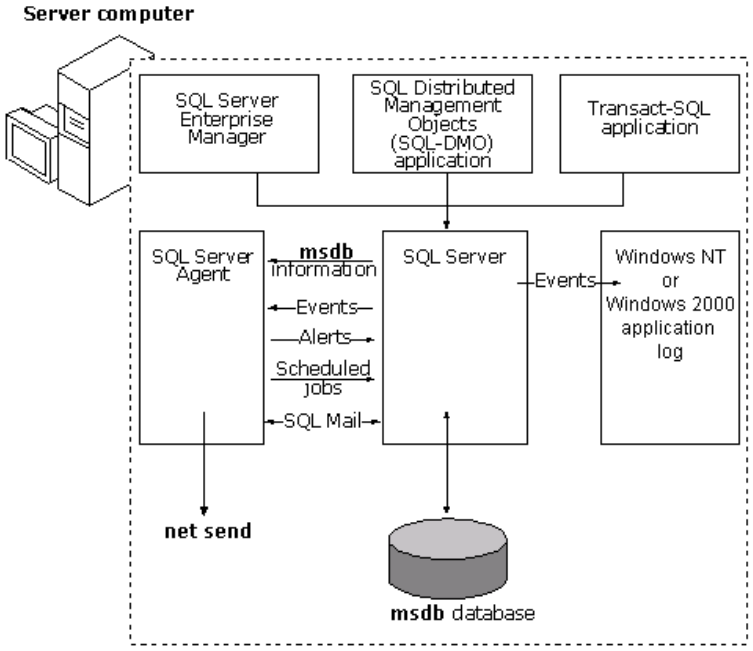


Dịch vụ MS SQLServer.

Dùng quản lý tất cả các file gồm các CSDL mà SQL Server quản lý, là thành phần xử lý tất cả các lệnh của Transact-SQL được gửi từ các trình ứng dụng client, phân phối các nguồn tài nguyên khi có nhiều user cùng truy nhập một lúc. Đây là dịch vụ quản trị cơ bản, khi ngắt dịch vụ này hệ thống sẽ ngưng tất cả các công việc khai thác dữ liệu.

Dịch vụ SQLServerAgent.

Hỗ trợ các tính năng cho phép lập thời biểu các hoạt dộng theo từng giai đoạn tren SQL Server, hoặc thông báo cho người quản lý hệ thống về những sự cố của hệ thống, bao gồm các thành phần Jobs, Alerts, Operator.

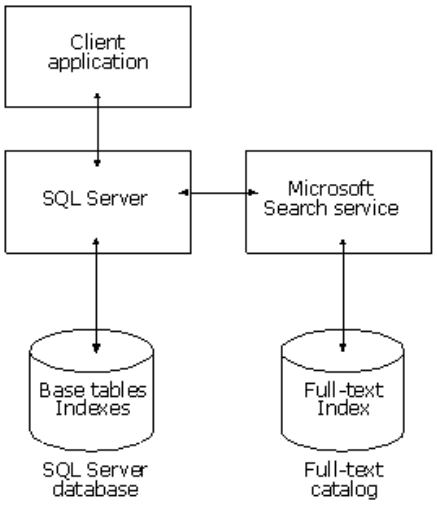


Dịch vụ Microsoft Search.

Cung cấp dịch vụ tìm kiếm và tìm kiếm văn bản với các phép toán cơ bản sau:

+ Ký tự (chuỗi): =, >, >=, <, <= được so sánh với một chuỗi hằng.

+ So sánh chuỗi nhỏ trong văn bản hoặc chuỗi có kích thước lớn, văn bản.



Dịch vụ MS DTC.

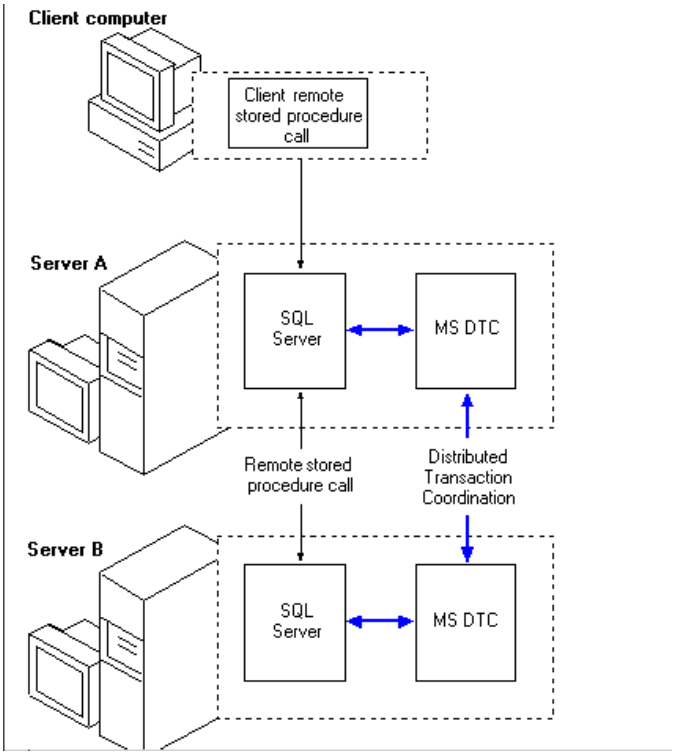
Là dịch vụ cho phép trong một phiên giao vận có thể sử dụng dữ liệu được phân phối trên nhiều server khác nhau, thực hiện theo các bước cơ bản sau:

+ Gọi các thủ tục lưu trữ trên các server xa sử dụng SQL Server

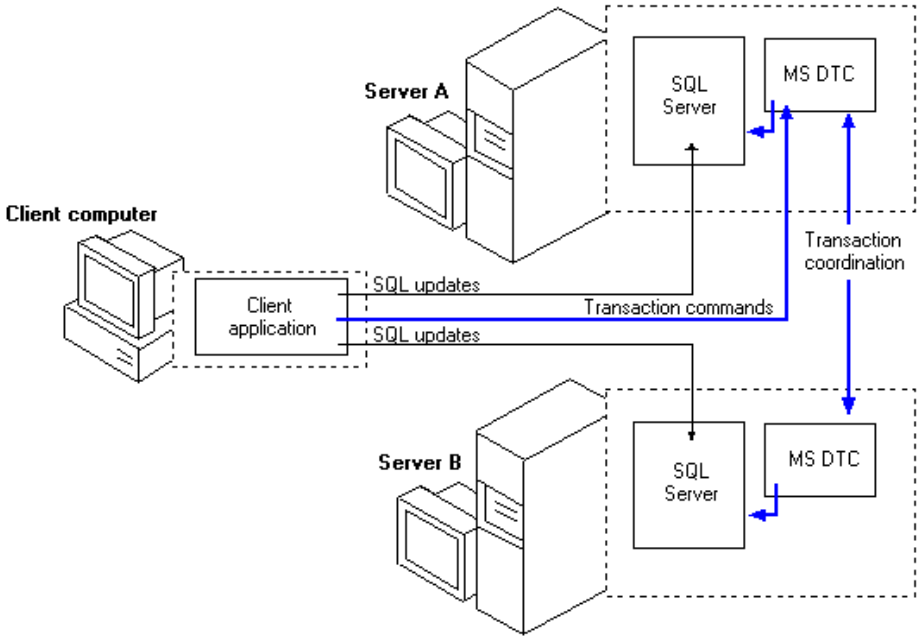
+ Tự động gọi hoặc tạo các phiên giao vận cục bộ và các giao vận với các máy chủ từ xa

+ Tạo bộ dữ liệu được cập nhật hoặc được phân phối bởi các server xa.

Xem xét sơ đồ hoạt động sau:



Như sơ đồ trên khi client triệu gọi một thủ tục có sẵn đến server cục bộ, khi có yêu cầu dữ liệu trên server khác, thông qua dịch vụ MS DTC server cục bộ sẽ triệu gọi các thủ tục từ server từ xa, kết quả có thể tạo được bộ dữ liệu được tập trung từ nhiều server khác nhau.



QUẢN TRỊ SERVER.

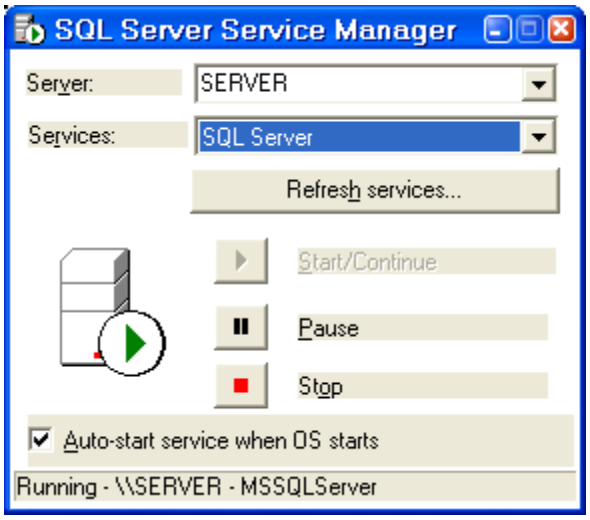
Như đã nếu trên mỗi Instance được coi là một hệ thống quản trị CSDL SQL

Server và có thể gọi tắt là Server. Server có chức năng quản trị toàn bộ hệ thống của SQL Server (dữ liệu, bảo mật, người dùng, tác vụ, các dịch vụ khác,...).

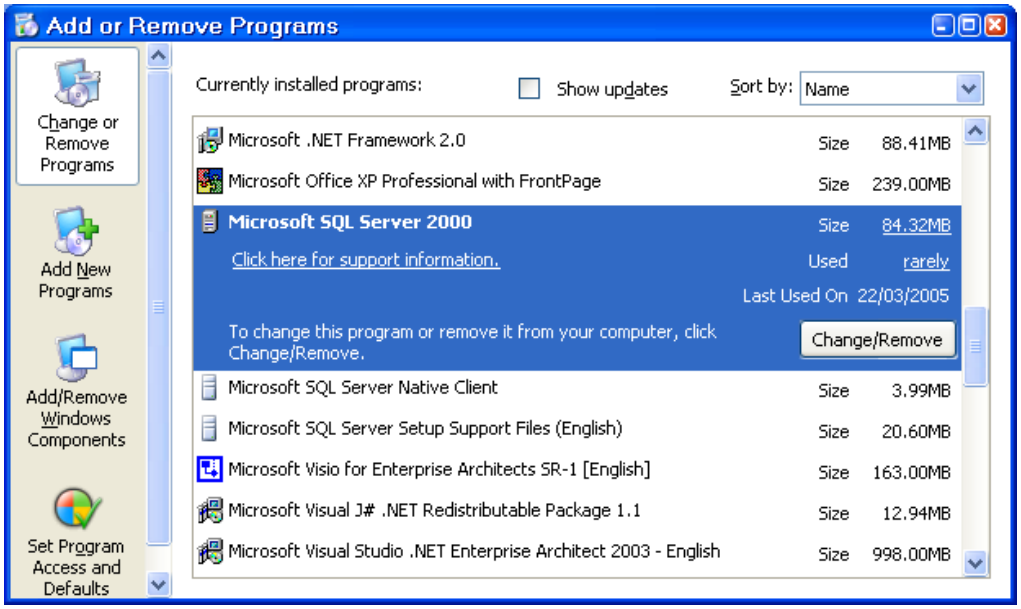
Các ứng dụng hoặc các công cụ khai thác dữ liệu (GUI) sẽ thực hiện khai thác dữ liệu do Server quản lý khi có kết nối đến Server. Tuy nhiên một vấn đề quan trọng là Server đó phải sẵn sàng phục vụ.

Thực hiện quản trị Server là việc thực hiện các công việc sau:

+ Bật/tắt/tạm dừng dịch vụ của SQL Server.



+ Gỡ bỏ hoặc cài đặt Server (Instance).



+ Thay đổi, nâng cấp phiên bản.

THIẾT LẬP KẾT NỐI ĐẾN SERVER.

Để khai thác được dữ liệu của hệ thống SQL Server ta phải thực hiện kết nối (connect) đến Server, việc kết nối có thể thực hiện từ các vị trí: Ứng dụng, công cụ khai thác của SQL Server là SQL Client. Trong phần này ta sẽ xem xét việc kết nối từ SQL Client đến Server.

Mỗi Server khi cài đặt đã có một tên là tên của Instance được đặt. Trên một mạng máy tính nếu có đủ quyền hạn ta hoàn toàn có thể thực hiện kết nối đến Server nói trên.

Từ một máy SQL Client có thể thực hiện đồng thời kết nối đến nhiều Server khác nhau, đây cũng chính là ưu điểm của SQL Server.

Nếu bạn cài đặt phiên bản SQL Server trên máy tính bạn cũng phải làm tiòan bộ các bước kết nối như SQL Client, phiên bản SQL Server được coi như gồm 2 phần: Hệ thống quản trị, công cụ khai thác SQL Client.

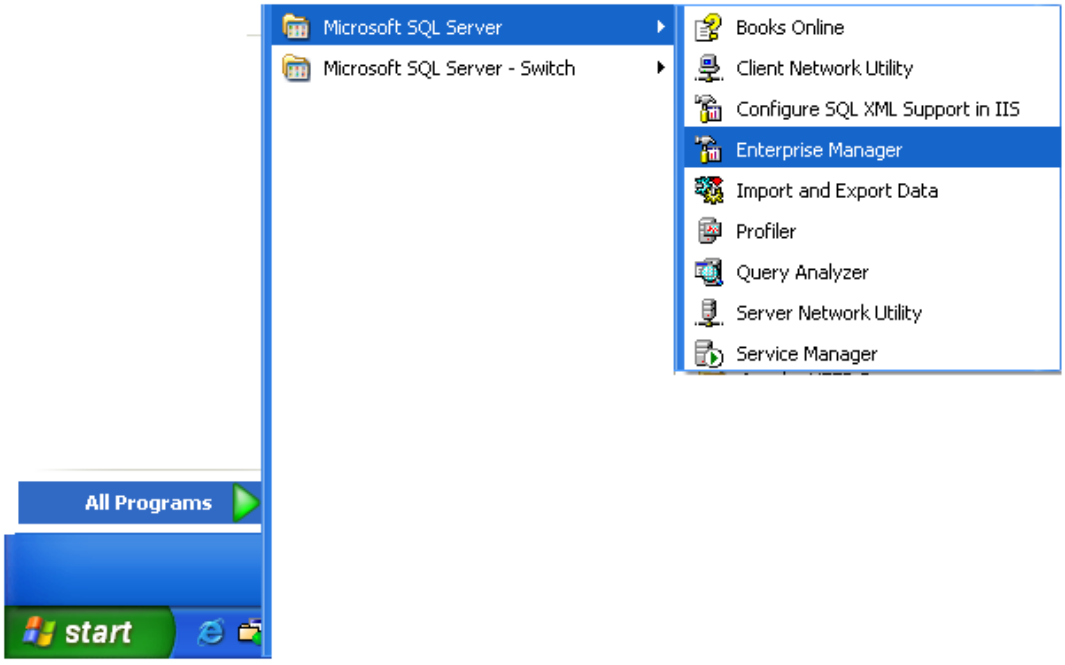
Dù sử dụng công cụ nào để khai thác đã được cài đặt trên máy tính của bạn, quyền hạn khai thác, quản trị phụ thuộc vào user thực hiện kết nối.

Quản trị Server Group.

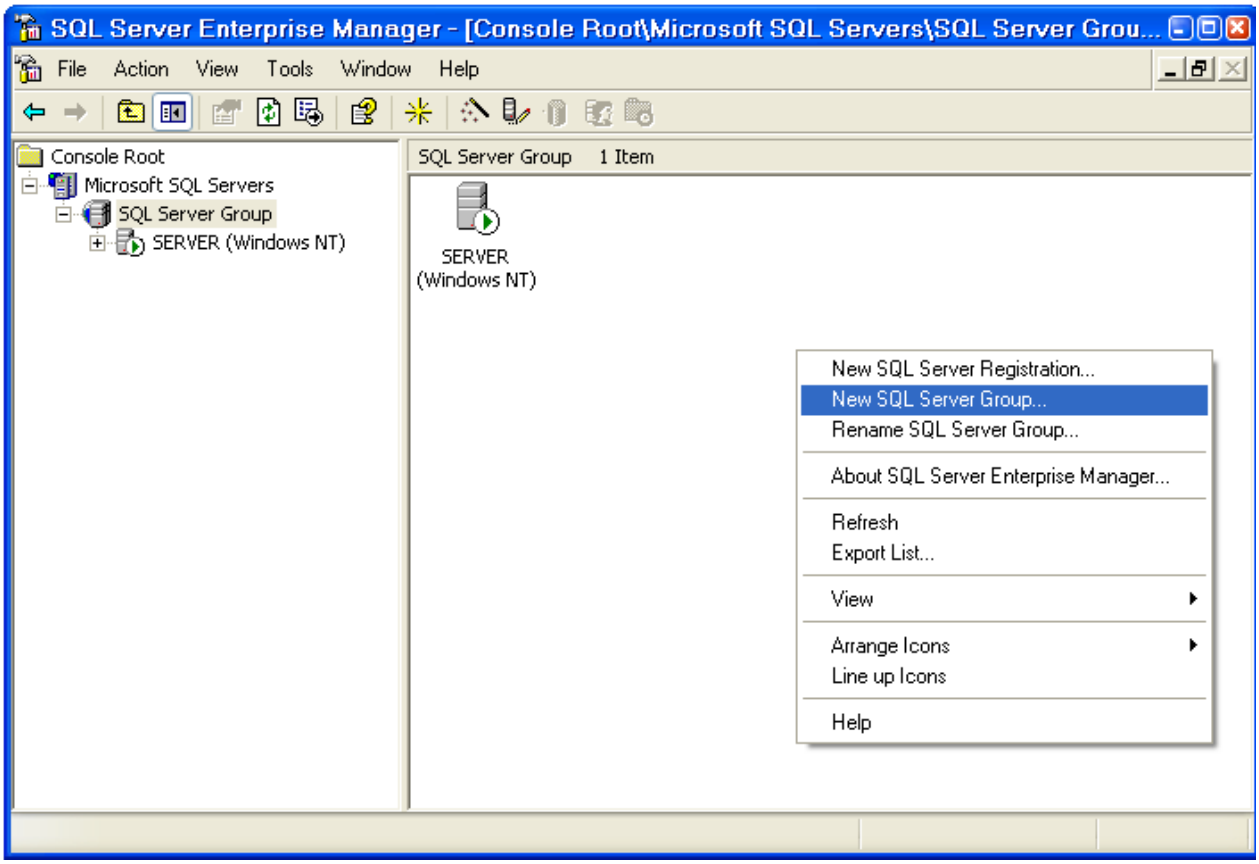
Server Group là công cụ dùng quản lý các kết nối (sẽ thực hành sau) tương tự như khái niệm thư mục trong hệ điều hành, trong các Server Group chứa các Server Group con hoặc các kết nối đến Server.

Các bước thực hiện như sau:

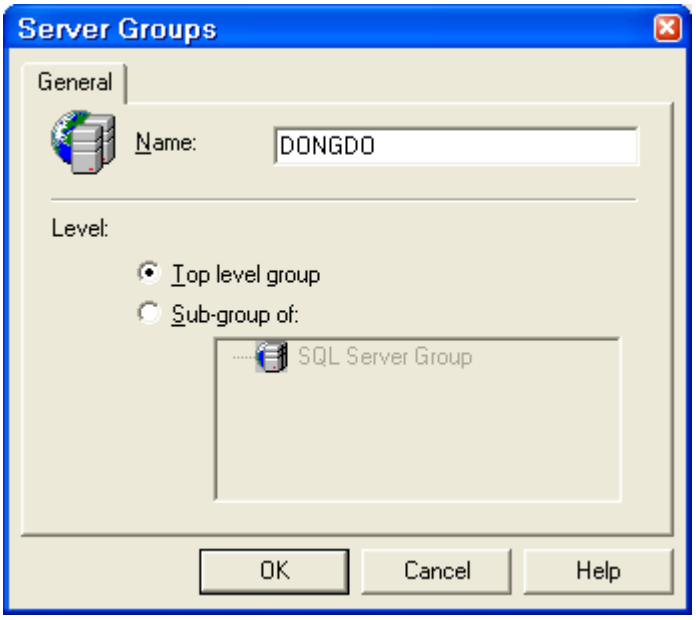
- Vào chức năng Enterprise manager như hình dưới



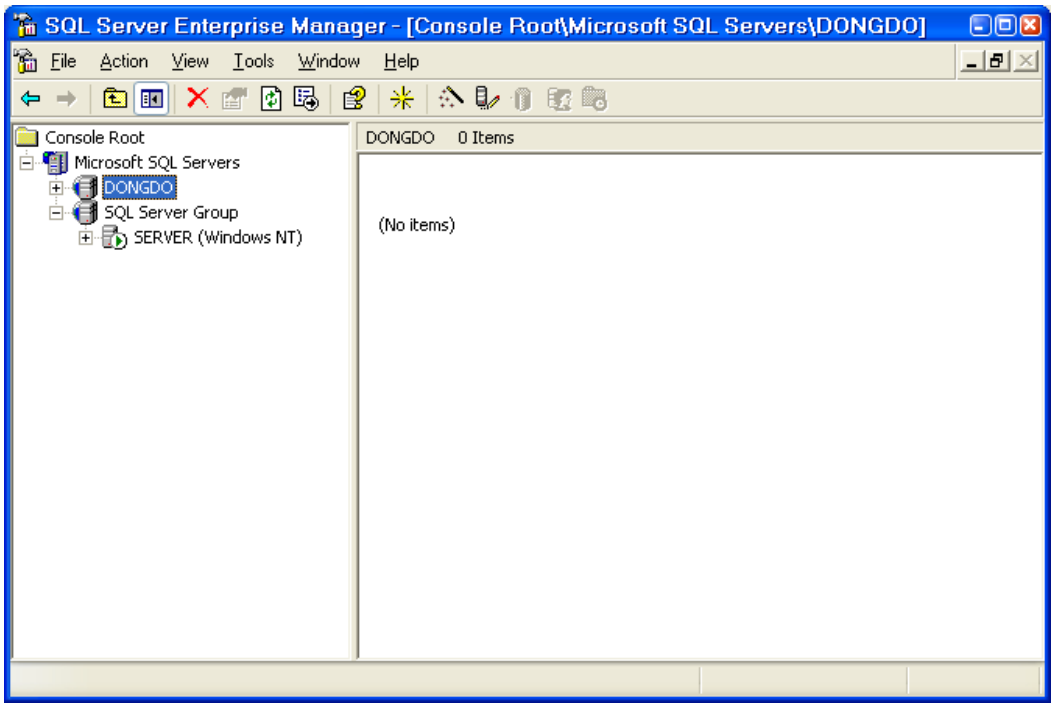
- Di chuyển vào mức trong bằng cách nhấn vào dấu + của cây các đối tượng.



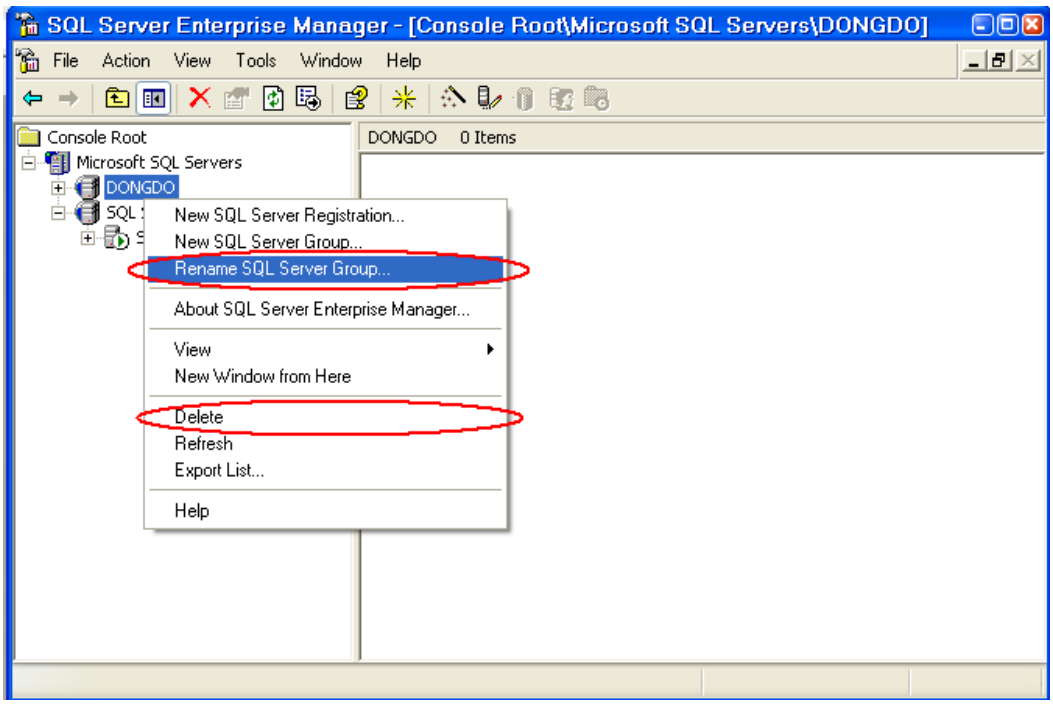
- Chọn New SQL Server group để tạo group mới



- Nhập tên group -> Ok



Các thao tác đổi tên, xóa được thực hiện bằng cách nhấn phải chuột vào group cần thao tác.



Thiết lập kết nối đến Server (thiết lập Server).

Là bước thiết lập kết nối đến Server từ Client, Server khác. Các kết nối được thể hiện bằng tên của Server kết nối đến (hay còn là tên của Instance), chính vì vậy nên tên các kết nối trên một Client là duy nhất, không trùng nhau trong toàn bộ client.

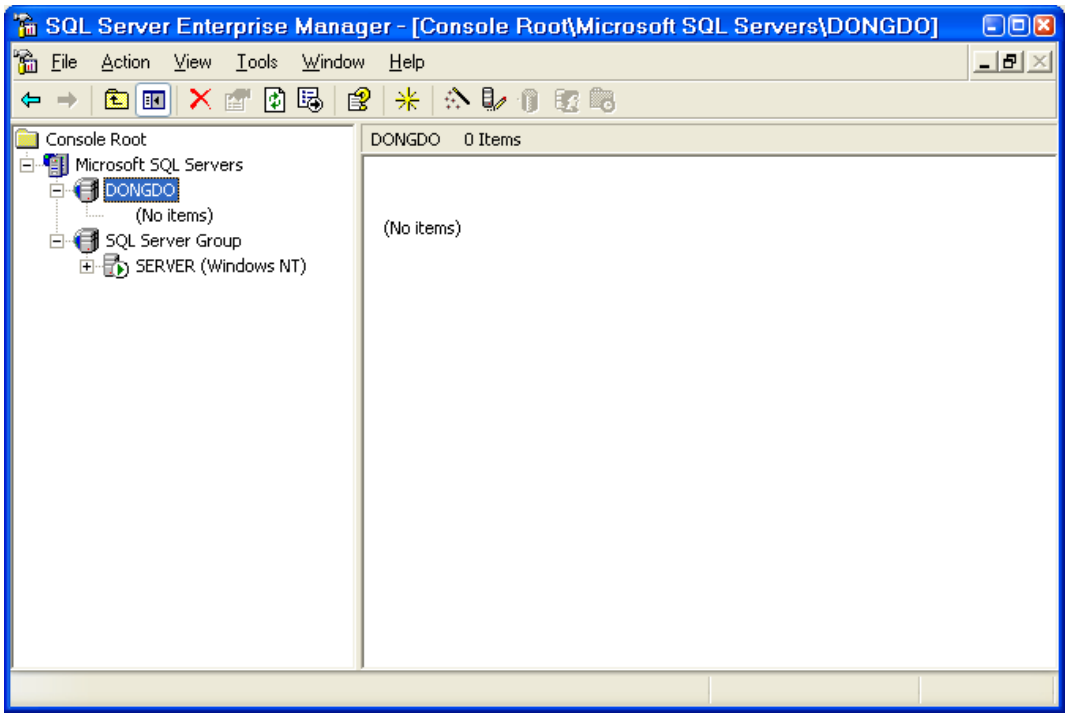
Trước khi thực hiện tạo kết nối ta phải chuẩn bị các tham số sau:

+ Tên Server (Instance) muốn kết nối đến

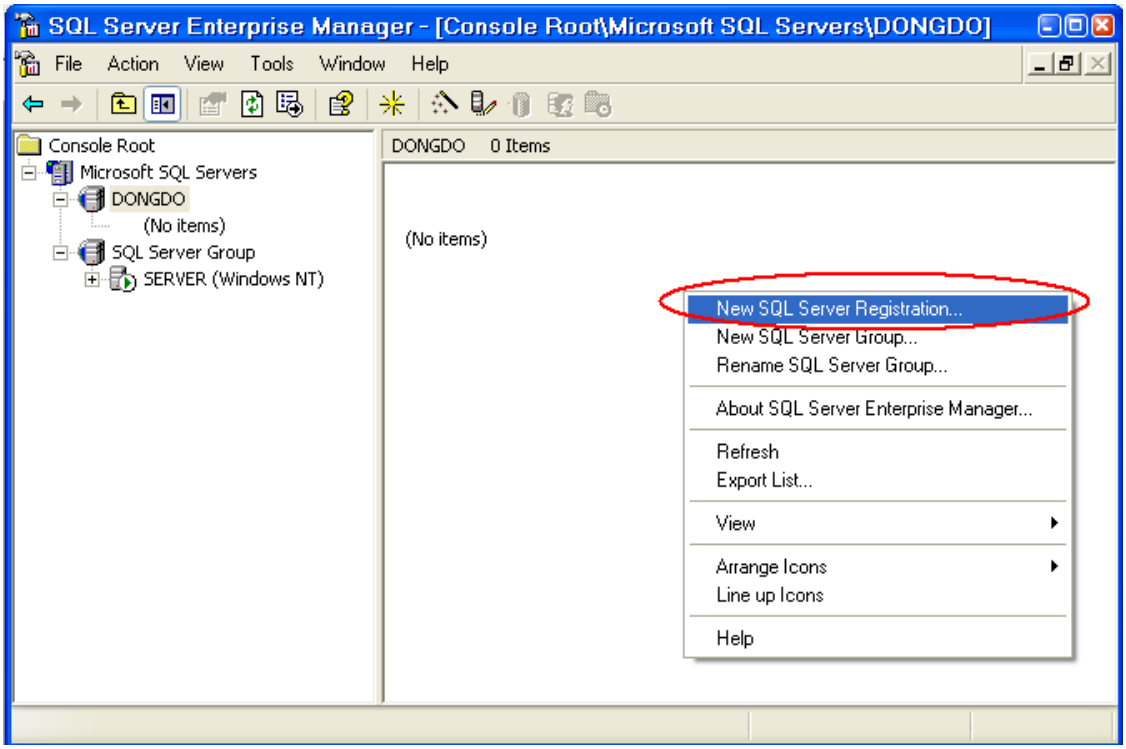
+ User name và Password của Server ta cần kết nối đến (tham số này do người quản trị Server cấp).

Cách làm như sau:

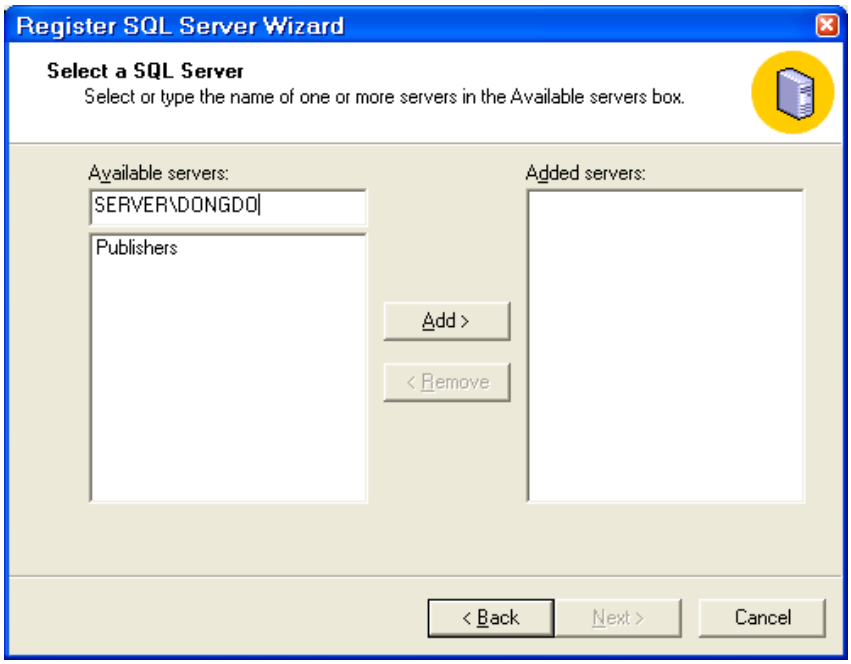
- Vào Enterprise và chọn Server group



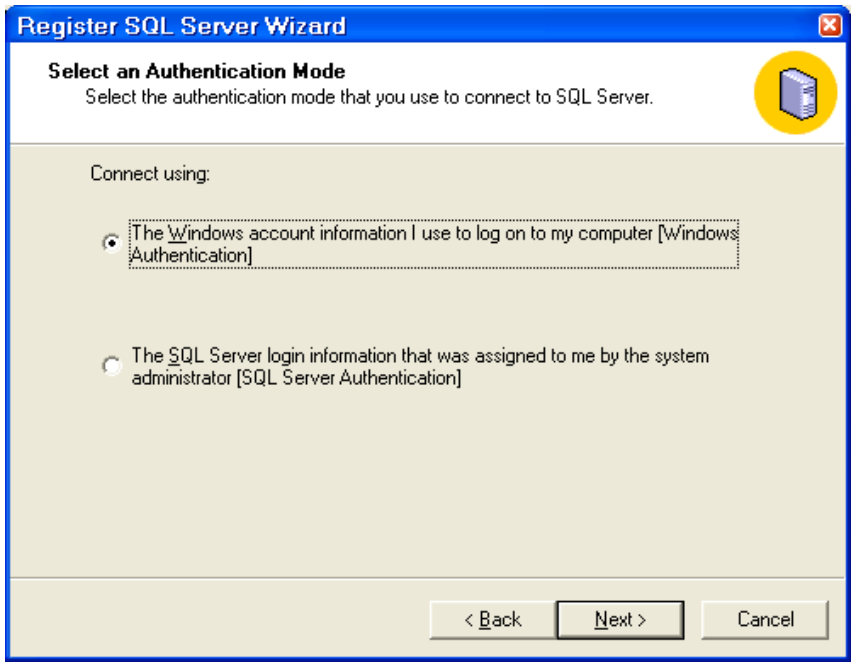
- Nhấn nút phải chuột vào cửa sổ bên phải, chọn New SQL Server Registration.



- Nhập tên Server.

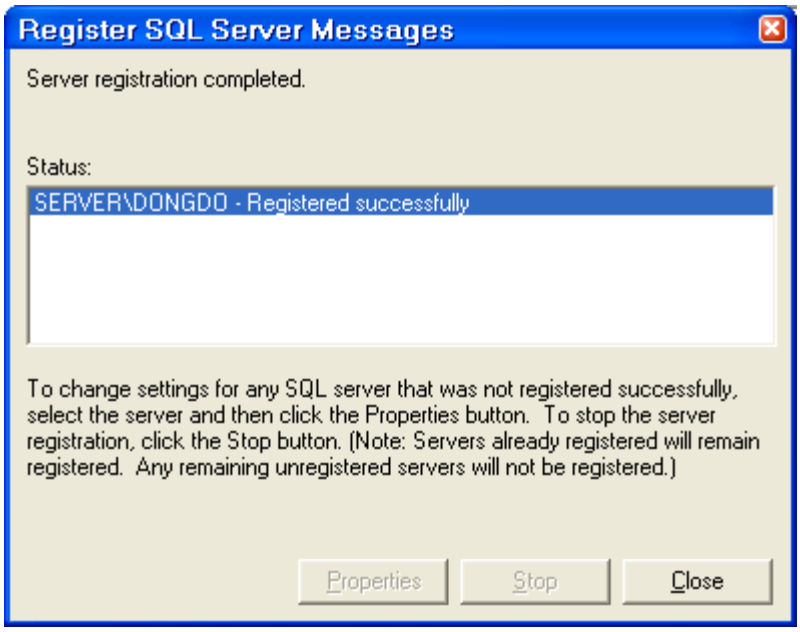


- Nhấn Add -> Next



- Chọn chế độ bào mật (thông thường chọn The Windows account information để chọn chế độ bảo mật của Widnows, phần này sẽ xem xét kỹ trong bài sau) -> Next -> Chọn Server Group.

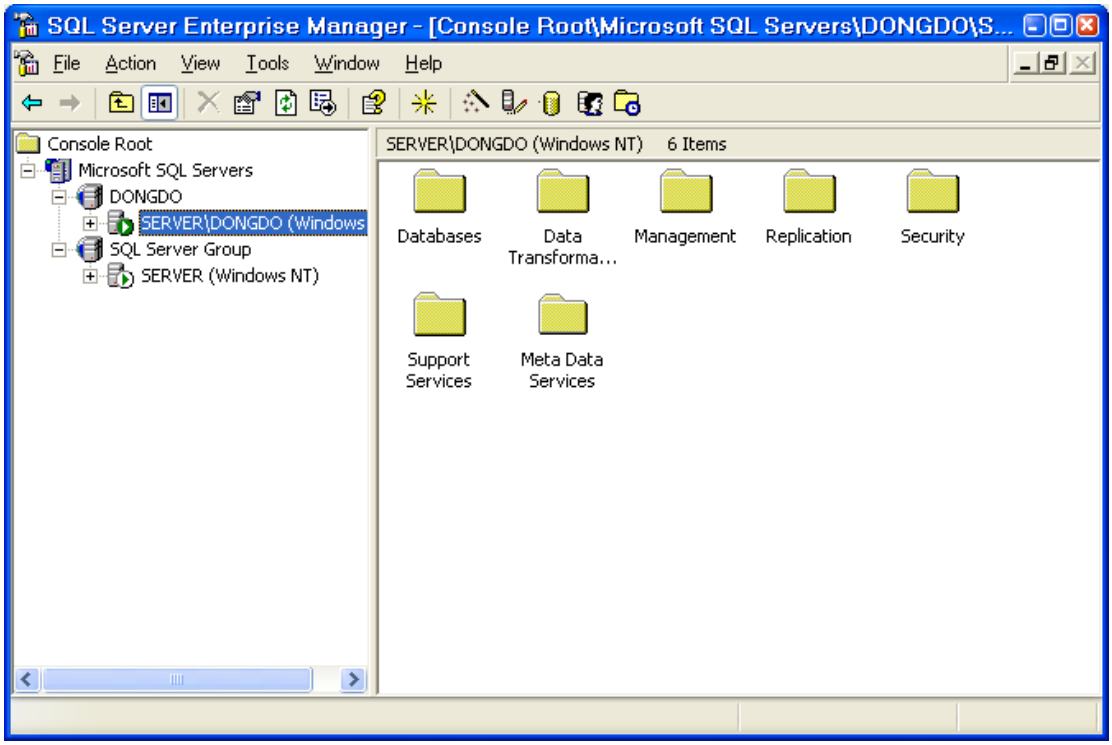
- Nhấn Finish.



Khi màn hình xuất hiện thông báo Registered successfully là việc thiết lập đã thành công.

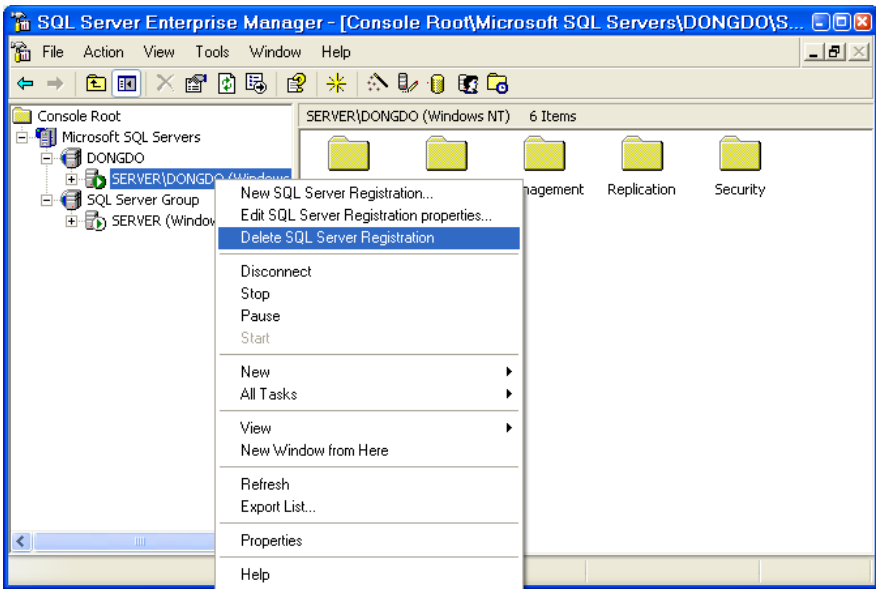
- Nhấn Close.

Sau khi thiết lập xong kết nối xuất hiện trên danh sách các kết nối.



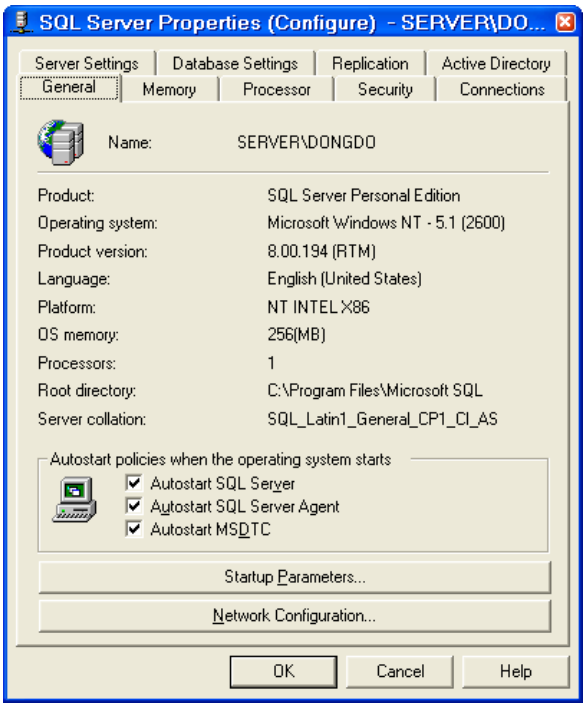
Kết nối như đã thiết lập có quyền hạn khai thác phụ thuộc vào user kết nối, trong ví dụ trên quyền hạn phụ thuộc vào user đã truy nhập vào Windows, tuy nhiên trong những bài sau sẽ giới thiệu cách thứac tạo user, sử dụng user của SQL Server để thực hiện tạo kết nối và khai thác.

Để xóa hoặc sửa thông tin cho kết nối chọn Delete hoặc Edit khi nhấn nút phải chuột vào tên kết nối cần thao tác.



Xem và thay đổi tham số cho Server.

Để thay đổi tham số cho Server, hãy chọn tên kết nối -> nhấn nút phải chuột -> chọn Properties.



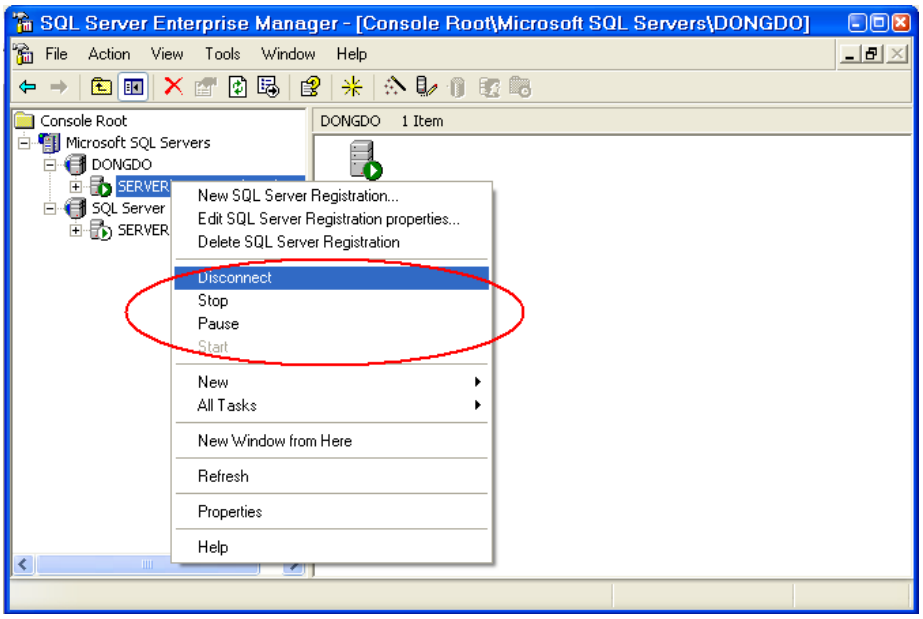
Tuy nhiên các tham số trên có thể bạn chưa xem xét, nên trong bài này chỉ giới thiệu mã sẽ xem xét một số tham số cơ bản trong như bài liên quan.

Bật/tắt/tạm dừng/kết nối/ngắt kết nối Server.

Ta có thể thực hiện tạm điều khiển dịch vụ MS SQL Server từ kết nối.

- Chọn tên kết nố

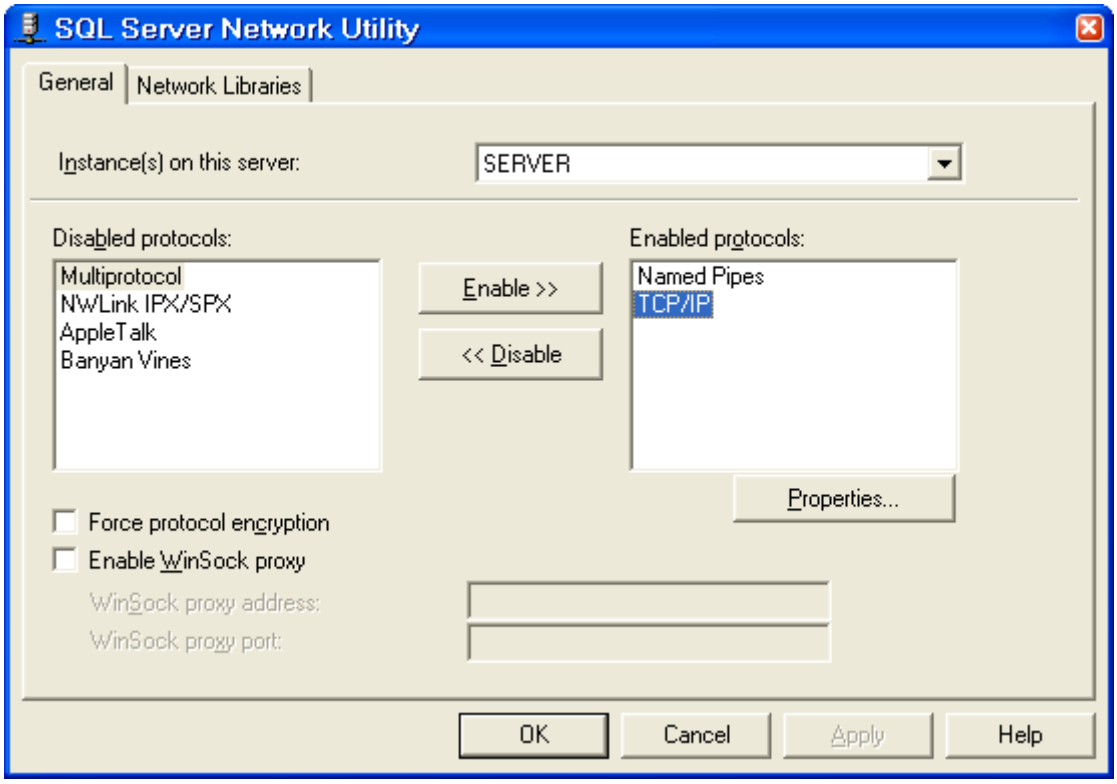
- Nhấn nút phải chuột



CẤU HÌNH KẾT NỐI MẠNG CỦA SERVER.

Để các Client hoặc các ứng dụng kết nối được đến Server, ta phải cấu hình các phương thức kết nối phù hợp với kết nối mạng. Kết nối mạng có thể sử dụng kết nối thông qua Proxy, thông qua mạng Internet.

Khi sử dụng kết nối nào đi nưa thì trước hết ta phải chọn giao thức phù hợp với giao thức mạng đạng sử dụng. Thực hiện bằng cách chọn Server network utility



- Chọn giao thức đưa vào danh sách enabled để sử dụng và đưa vò danh sách Disabled để không sử dụng.

- Chọn Properties để chọn cổng, tham số của giao thức.

- Chọn Enable WinSock proxy để thực hiện kết nối qua Proxy.

- Chọn Force protocol encryption để sử dụng kết nối qua Internet không dùng Fire Wall với SQL Server.

QUẢN TRỊ CÁC CLIENT.

Khi Server đã sẵn sàng cho kết nối, việc tiếp theo là xem xét đến các client kết nối đến server. Trong phần này ta sẽ xem xét cấu hình client kết nối đến server.

Các client kết nối đến server đều thực hiện trên cở sở hệ thống truyền tin của mạng máy tính, tuy nhiên các ứng dụng client kết nối đến server để thực hiện khai thác dữ liệu trên server thông qua một số phương thức kết nối sau:

- OLE DB: Có 2 kiểu Microsoft OLE DB Provider for SQL Server và Microsoft OLE DB Provider for ODBC.

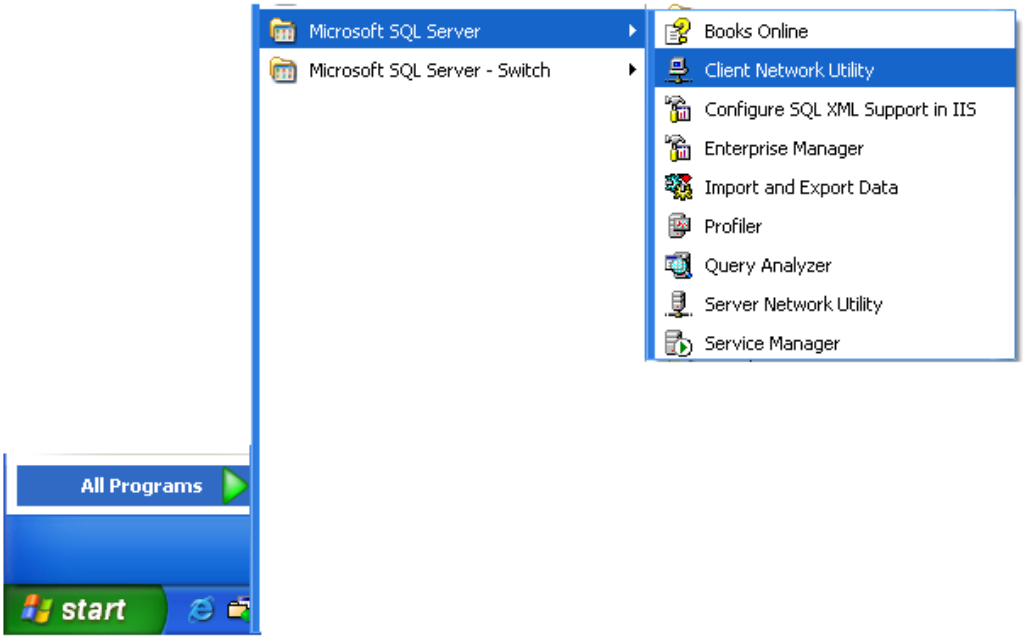
- ODBC: Kết nối thông qua SQL Server Enterprise Manager và SQL Query Analyzer sử dụng SQL Server ODBC.

- DB-Library: Sử dụng lệnh SQL Server **isql**.

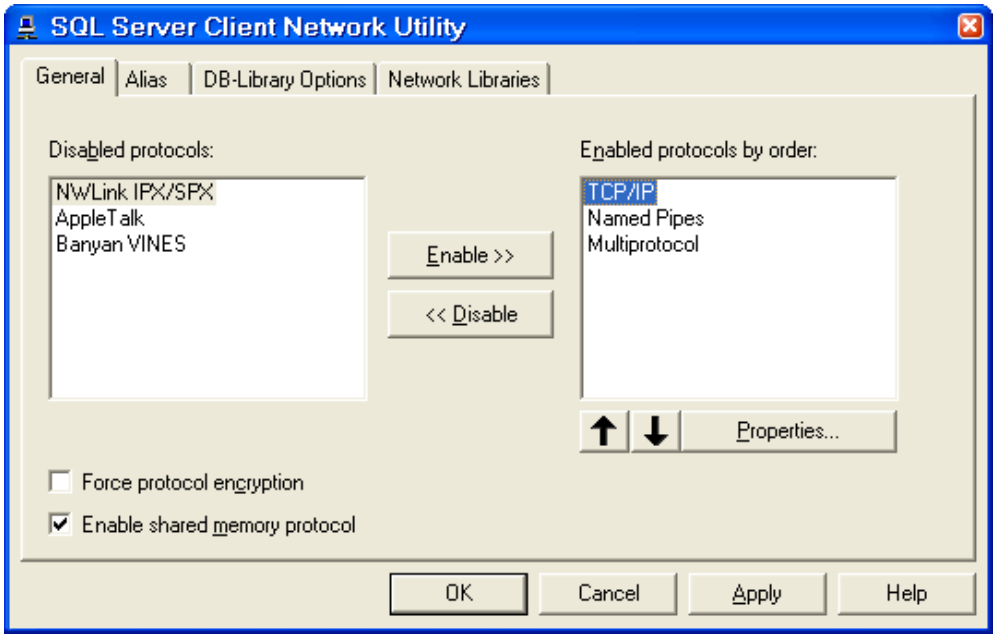
Cấu hình Net-Library.

Như đã xem xét trước mỗi Instance khi cấu hình xác định một địa chỉ và số hiệu cổng riêng, nên việc kết nối thông qua Net-Library là kết nối thông qua địa chỉ và như vậy kết nối đã xácđịnh được đến Instance.

Trên Server thông thường được cấu hình theo TCP/IP Sockets và Named Pipes Net-Libraries, trên client thông thường cấu hình theo Thực hiện cấu hình ta sử dụng Client network utility.



- Chọn Client network utility.



- Chọn giao thức và các tham số liên quan tương ứng với server, có thể thực hiện định tên Server sang tên mới trên bảng Alias.

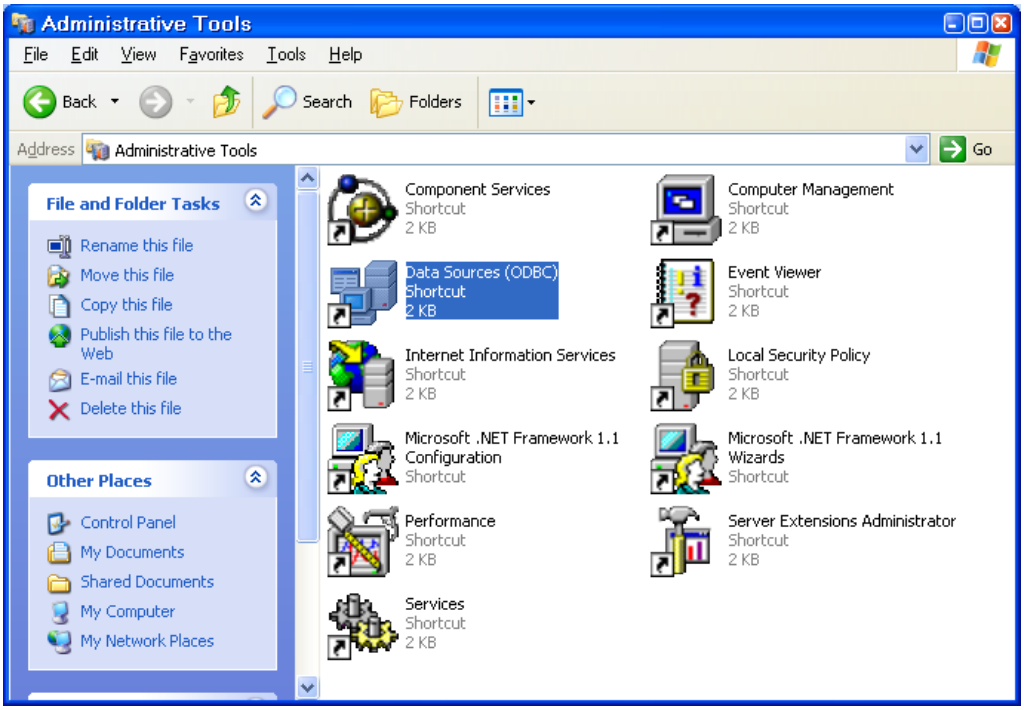
Cấu hình ODBC.

ODBC viết tắt của từ Open DataBase Connectivity, là công cụ kết nối mở. ODBC được Windows cũng cấp dẫn khi cài đặt, được sử dụng làm kết nối trung gian giữa ứng dụng và các hệ quản trị CSDL (Dbase, Access, SQL Server, Oracle,…).

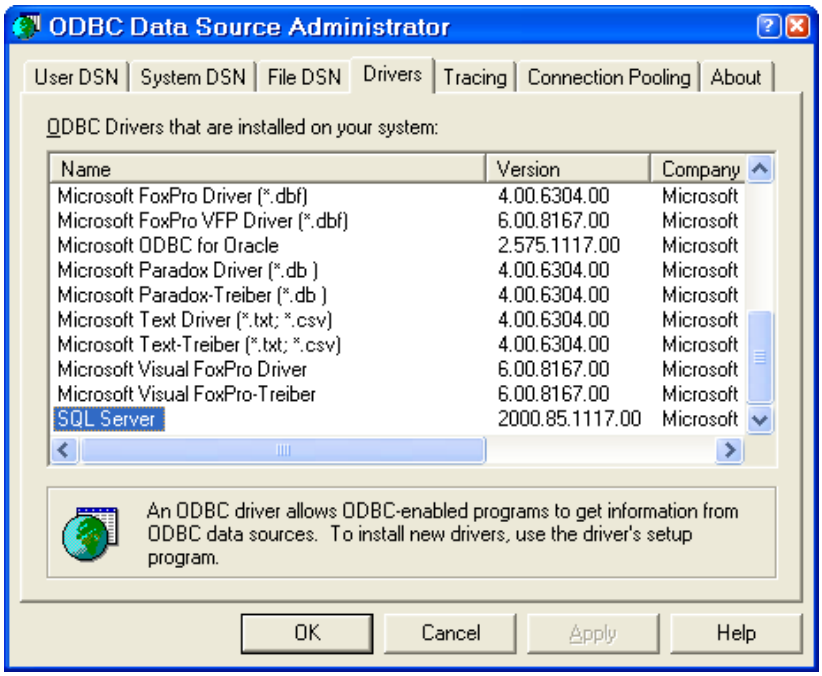
Thông qua ODBC ứng dụng chỉ cần xác định tên của nguồn trong ODBC (gọi là Data Source) và tài khoản khi truy nhập để thực hiện quyền mà không cần quan tâm đến cơ sở dữ liệu đang nằm ở đâu.

Thông thường khi cài đặt hệ quản rị CSDL mới thì Windows sẽ tự cập nhật vào danh sách các Driver điều khiển ODBC của hệ quản trị CSDL đó. Thực hiện tạo ODBC cho SQL Server như sau:

- Chọn ODBC trong Administrative tools (Control panel).

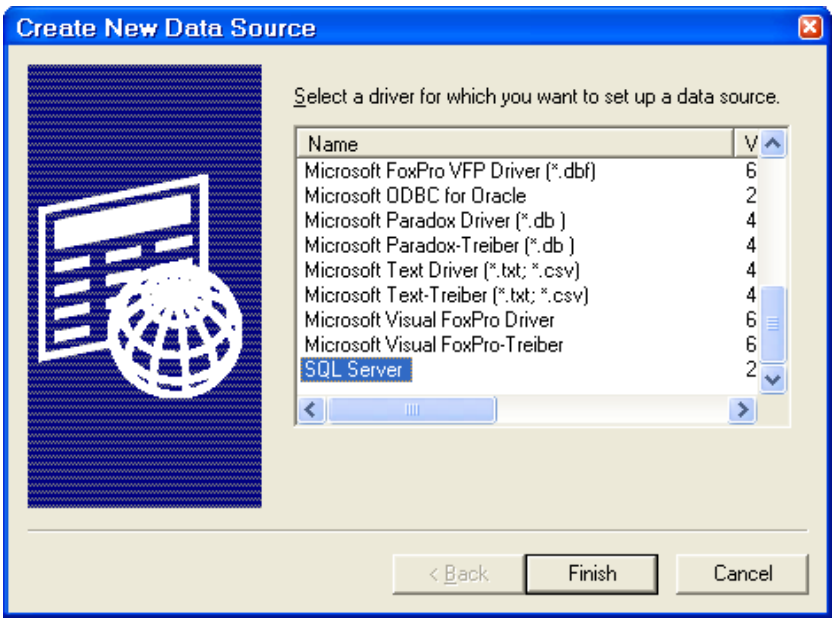


- Chọn bảng Drivers, trong danh sách kiểm tra xem đã có SQL Server chưa

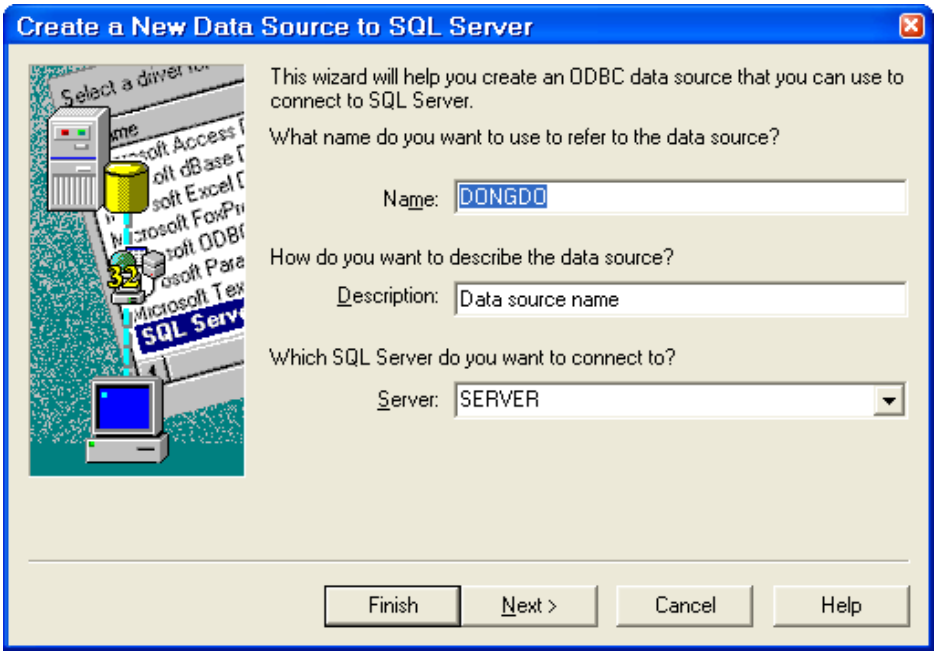


Nếu chưa có kiểm tra cách cài đặt SQL Server (thông thường Windows tự cập nhật).

- Chọn bảng User DSN (Data Source Name) -> Add.

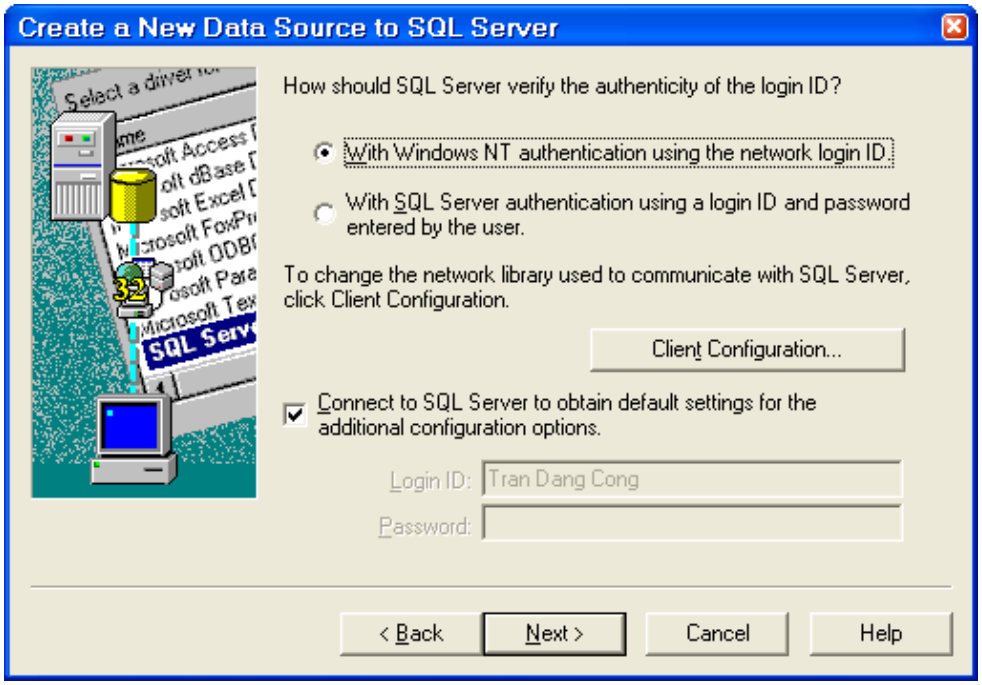


- Nhấn Finish.



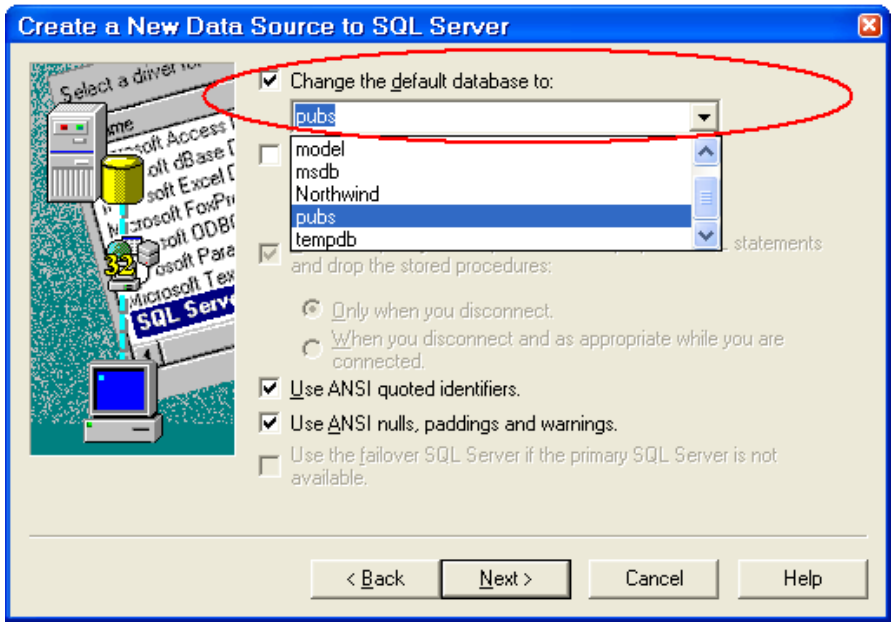
- Nhập tên DSN (đây là tên sẽ được sử dụng cho ứng dụng), thông tin mô tả, tên Server (Instance).

- Nhấn Next.

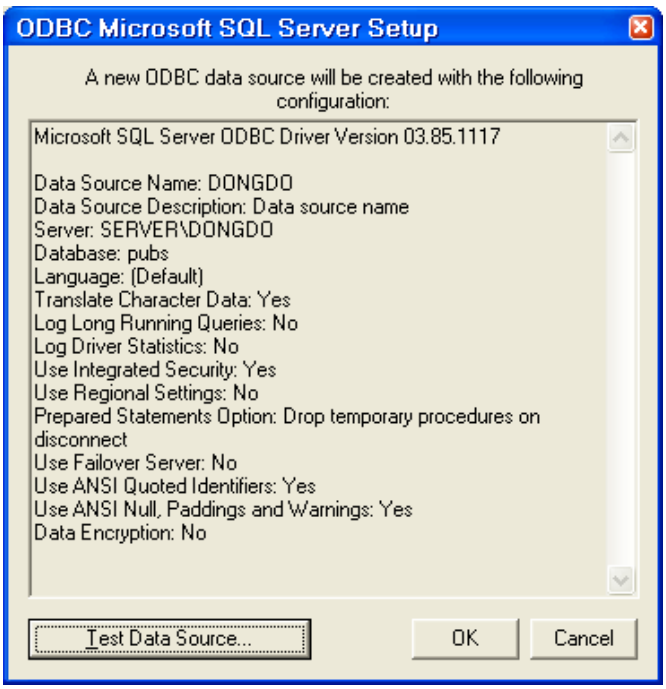


Trong cửa sổ có 2 lựa chọn: Sử dụng chế độ bảo mật kế thừa của Windows NT hoặc của SQL Server (sẽ xem xét sau), trước hết bạn hãy chọn lựa chọn kế thừa của Windows NT (lúc này quyền khai thác là quyền của người truy nhập vào Windows).

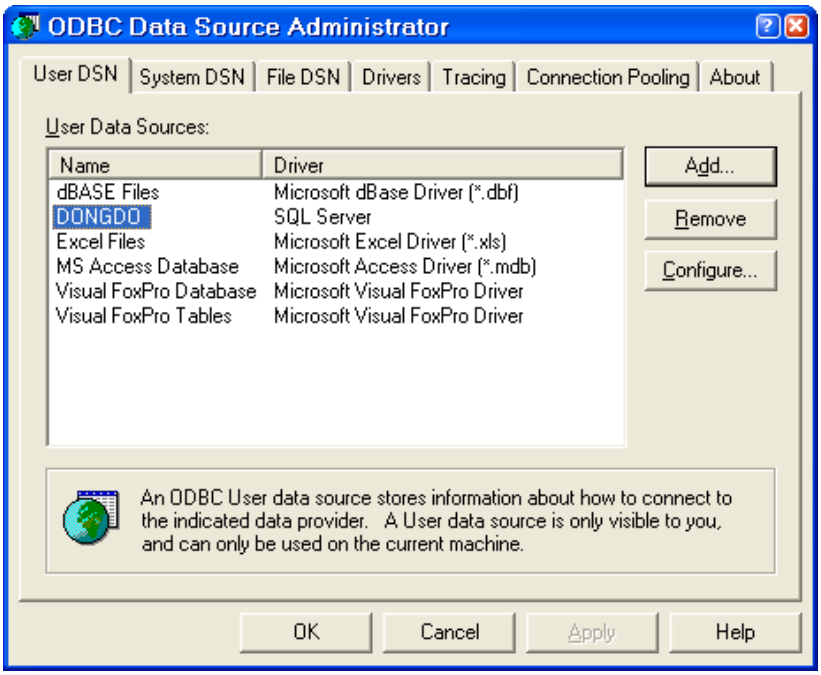
- Nhấn Next, chọn hộp chọn Change the default database to -> chọn cơ sở dữ liệu (việc tạo cơ sở dữ liệu sẽ xem xét bài sau, tại bước này bạn hãy chọn một cơ sở dữ liệu ví dụ có tên là Pubs, đây là cơ sở dữ liệu ví dụ do SQL Server tự thiết lập để làm mẫu).



- Nhấn Next -> Finish.



Để kiểm tra kết nối có thành công không bạn nhấn Test Data Source sau đo nhấn Ok để kết thúc và thu được màn hình sau:



Trên danh sách các Data Source có tên DONGDO vừa được tạo, Data Source tạo ra sẽ được sử dụng trong ứng dụng client.

Cấu hình OLE DB.

OLE DB là phương thức khá quen thuộc đối với người lập tình CSDL (lập trình trên Desktop hoặc trên Internet). OLE DB sử dụng với nhiều hệ quản trị CSDL khác nhau, mỗi hệ quản trị có cú pháp riêng và chỉ định driver điều khiển cho nó. Với SQL Server thông thường sử dụng 2 phương thức kết nối sử dụng OLE DB:

- Microsoft OLE DB Provider for SQL Server (SQLOLEDB): Không sử dụng ODBC, xác định driver cho SQL Server.

- Microsoft OLE DB Provider for ODBC: Sử dụng ODBC đã tạo (trong phần trước).

QUẢN TRỊ CƠ SỞ DỮ LIỆU

Trong chương này ta sẽ xem xét cấu trúc vật lý, tạo, xóa, sửa tham số của cơ sở dữ liệu.

CẤU TRÚC CƠ SỞ DỮ LIỆU.

Chắc hẳn khi nghiên cứu đến hệ quản trị CSDL SQL Server bạn đã xem xét đến các hệ quản trị CSDL như DBase hoặc Access, với hệ quản trị CSDL như trên mỗi cơ sở dữ liệu khi sử dụng (thực hiện mở CSDL) sẽ mở trực tiếp từ tập tin chứa CSDL, tập tin chứa CSDL sẽ có một tập tin chính (ví dụ *.dbf hoặc *.mdb) và tập tin phụ nhưng khi ta thao tác ta chỉ cần quan tâm đến tập tin chính. Nên trong các ứng dụng thông thường ta thường dùng các thao tác mở (open) để mở tập tin chính chứa CSDL và đóng (close) đóng tập tin chính chứa CSDL mà không cần quan tâm đến việc đã kết nối đến CSDL chưa (không có phương thức kết nối).

SQL Server quản lý trực tiếp các CSDL, danh sách mỗi Server sẽ gồm danh sách các tên CSDL, tên các CSDL là duy nhất, không trùng nhau. Mỗi CSDL SQL Server sẽ quản lý các cấu trúc vật lý của nó. Chính từ cách thức quản lý như trên mà việc quản trị cơ sở dữ liệu có một số đặc điểm sau:

+ Để Client khai thác CSDL trước hết phải thực hiện kết nối đến Server quản trị CSDL đó.

+ Chỉ thực hiện khai thác với các CSDL có tên trong danh sách các CSDL mà Server quản lý.

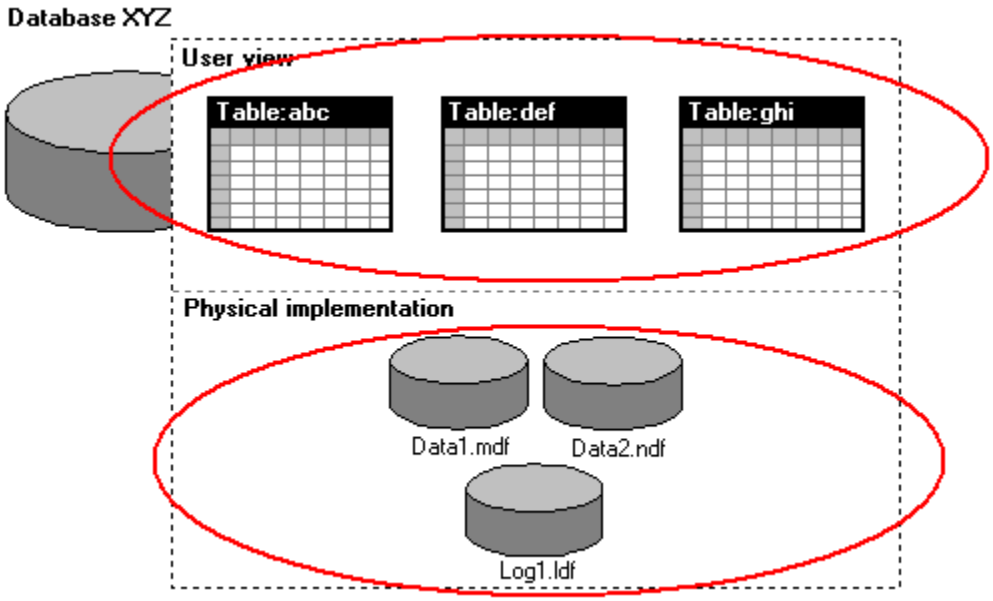
+ Không có các phương thức mở CSDL trực tiếp từ tập tin như Dbase hoặc Access.

+ Khi đã kết nối đến Server, Client chỉ thực hiện được quyền khai thác theo quy định đã định sẵn trong CSDL (phân quyền trong CSDL).

Cấu trúc cơ sở dữ liệu.

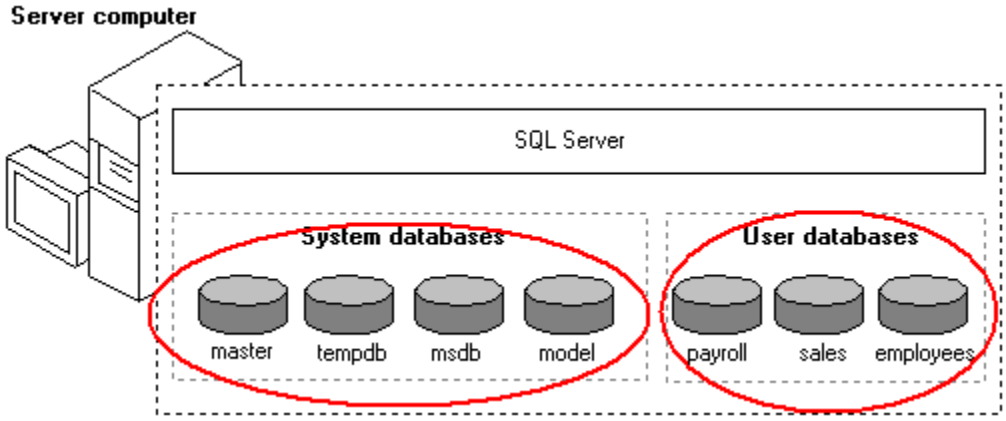
Cơ sở dữ liệu trong SQL Server lưu trữ theo 2 phần: phần dữ liệu (gồm một tập tin bắt buộc *.mdf và các tập tin phụ *.ndf) và phần nhật ký (*.ldf). Như vậy một cơ sở dữ liệu có ít nhất 2 tập tin.

Cấu trúc logic trong CSDL gồm các table, view và các object khác. Sau đây là cấu trúc một CSDL.



Sơ đồ quản trị cơ sở dữ liệu của SQL Server.

Cơ sở dữ liệu trong SQL Server chia thành 2 loại: Cơ sở dữ liệu hệ thống (do SQL Server sinh ra khi cài đặt) và cở sở dữ liệu người dùng (do người dùng



tạo ta).

Cơ sở dữ liệu hệ thống gồm:

- Master: Lưu trữ các thông tin login account, cấu hình hệ thống, thông tin quản trị các CSDL, là CSDL quan trọng nên thường được sao lưu để bảo đảm an toàn cho hệ thống.

- Tempdb: Chứa các table tạm thời và các thủ tục được lưu trữ tạm thời. Các table và thủ tục nói trên được lưu trữ trong CSDL này phục vụ cho các user.
- Model: Được sử dụng khi template được sử dụng cho các CSDL được tạo trên một hệ thống.
- Msdb: Sử dụng bởi SQL Agent.

Tập tin của các CSDL nói trên như sau:

Tập tin CSDL	Tên tập tin vật lý	Kích thước ngầm định
master primary data	Master.mdf	11.0 MB
master log	Mastlog.ldf	1.25 MB
tempdb primary data	Tempdb.mdf	8.0 MB
tempdb log	Templog.ldf	0.5 MB
model primary data	Model.mdf	0.75 MB
model log	Modellog.ldf	0.75 MB
msdb primary data	Msdbdata.mdf	12.0 MB
msdb log	Msdblog.ldf	2.25 MB

Cấu trúc vật lý của CSDL.

Như các cấu trúc các CSDL hệ quản trị CSDL thông thường (Dbase, Access), SQL Server cũng quản lý tập tin dữ liệu của CSDL ở dạng vật lý theo trang (page) và phân đoạn (extent).

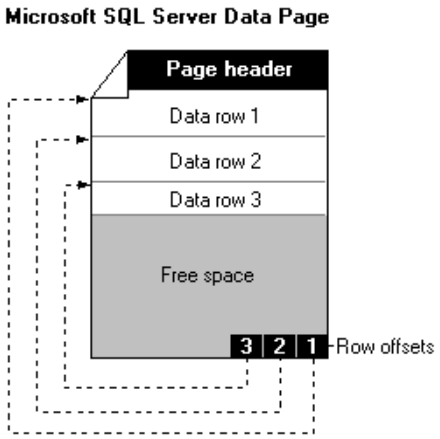
Page.

SQL Server quản lý một page có kích thước là 8KB, như vậy 1MB có 128 page, trong mỗi trang có 96 byte chứa thông tin của trang. Có 8 kiểu page như sau:

Tên	Nội dung
Data	Chứa tất cả các kiểu dữ liệu loại trừ **text**, **ntext** và **image**
Index	Các khóa Index.
Text/Image	**Text**, **ntext**, and **image** data.
Global Allocation Map, Secondary Global Allocation Map	Chứa các thông tin định vị của các extent.
Page Free Space	Chứa thông tin khoảng trống của page.

Index Allocation Map	Chứa các thông tin về Extent đã sử dụng cho Index và Page.
Bulk Changed Map	Chứa thông thông tin về các lệnh BACKUP LOG.
Differential Changed Map	Chứa các thông tin lệnh BACKUP DATABASE.

Đối với các tập tin nhật ký (*.ldf), các bản ghi được ghi lại liên tục, không phân trang.



Dữ liệu trong một trang sẽ bắt đầu lưu trữ từ sau phần thông tin Header, và lưu trữ liên tiếp, mỗi hàng có kích thước tối đa là 8060byte. Riêng đối với dữ liệu kiểu text, ntext, image đây là kiểu dữ liệu phức tạp và có kích thước lơn, SQL Server sẽ có chiến lược quản lý khác, phân tran riêng nhằm tăng hiệu quả truy vấn dữ liệu.

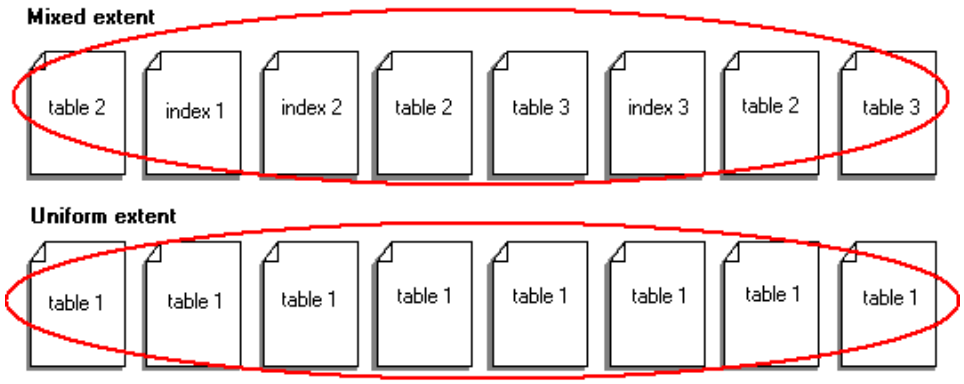
Dữ liệu trong SQL Server được lưu trữ trên đĩa và tạo chỉ mục Index theo cấu trúc dữ liệu kiểu B-Tree Plus (có thể tham khảo thêm trong những nội dung cấu trúc dữ liệu nâng cao).

Extent.

Extent là đơn vị dùng chứa các table và index, mỗi extent có 8 page hay 64KB. SQL Server có 2 kiểu extent:

- Uniform: Chỉ dùng lưu trữ cho một đối tượng,.
- Mixform: Có thể dùng lưu trữ 8 đối tượng.

Cấu trúc Extent như sau:



File.

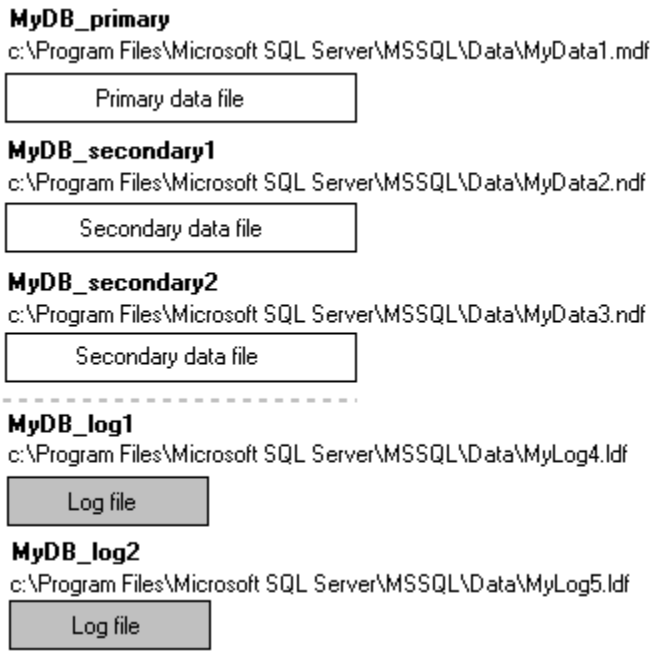
Tập tin lưu trữ một CSDL trong SQL Server có 3 loại.

Primary data file: Là file chính lưu trữ dữ liệu (*.mdf = Master Data File), mỗi CSDL có một file primary, lưu trữ điểm bắt đầu của một CSDL và các điểm kết nối đến các file lưu trữ tiếp theo (sencondary).

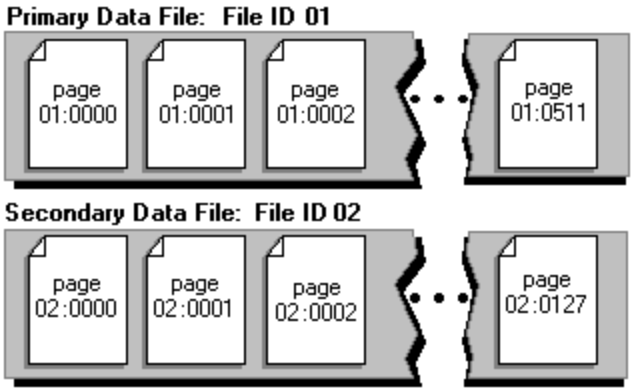
Secondary data file: Là tập tin lưu trữ dữ liệu sau Primary data file, một CSDL có thể có nhiều tập tin sencondary. Loại tập tin này cho phép một CSDL có thể phân tán dữ liệu ở nhiều nơi trên máy tính hoặc trên mạng.

Log file: Là loại tập tin lưu trữ thông tin nhật ký của CSDL.

Giả sử tạo một CSDL có tên MyDB, thông thường hệ thống ngầm định các tập tin như sau:



Các tập tin lưu trữ dữ liệu phân thành từng trang, các trang đánh số id liên tiếp theo từng file:



File group.

SQL Server sử dụng công cụ file group để giúp người dùng dễ dàng quản lý file, các file lưu trữ dữ liệu của một CSDL có thể nhóm thành từng nhóm, gồm 2 kiểu nhóm chính:

- Primary: Là nhóm bắt buộc có, dùng xác định cho file primary (*.mdf) và những file khác.
- User-defined: Nhóm do người dùng tạo ta, tự đặt tên để dễ quản lý.

QUẢN LÝ CƠ SỞ DỮ LIỆU.

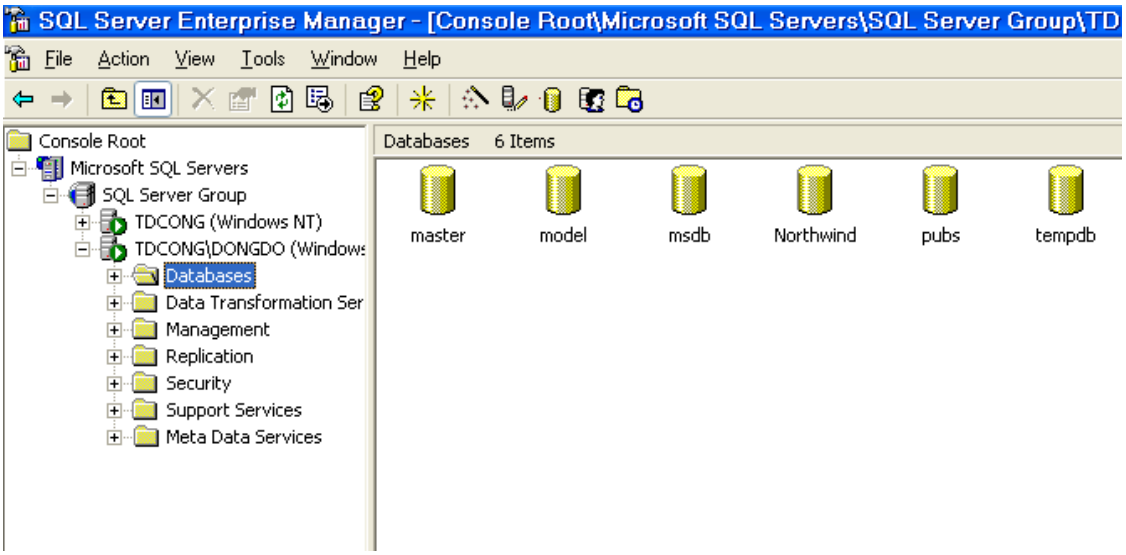
Tạo cơ sở dữ liệu.

Theo lý thuyết cơ sở dữ liệu, trước khi tạo CSDL ta phải thực hiện phân tích các thông tin liên quan mục đích sử dụng CSDL cho ài toán của mình: Tên CSDL, các table, ràng buộc,… tuân theo các chuẩn CSDL (phần này sẽ bàn kỹ trong bài sau)

Trong các thao tác với CSDL và đối tượng khác sẽ gồm 2 phần: Phần thao tác theo công cụ wizard và câu lệnh T-SQL.

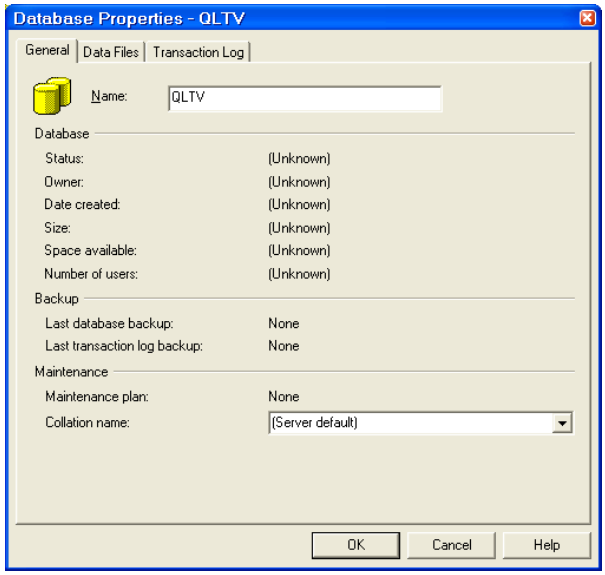
Tạo theo công cụ:

- Vào Enterprise Manager -> Databases.

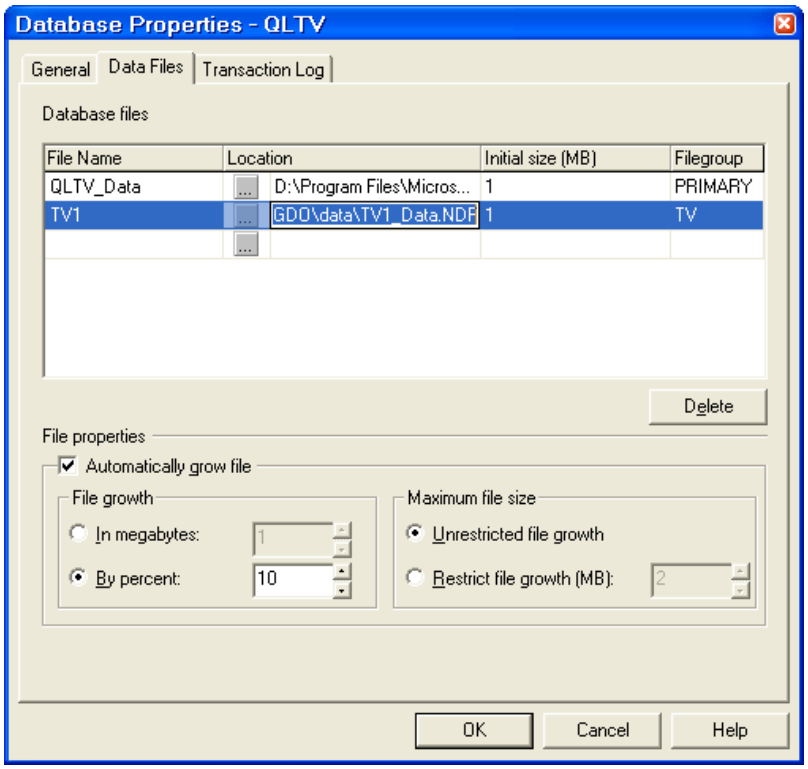


- Nhấn nút phải chuột/hoặc menu Action -> New Database…

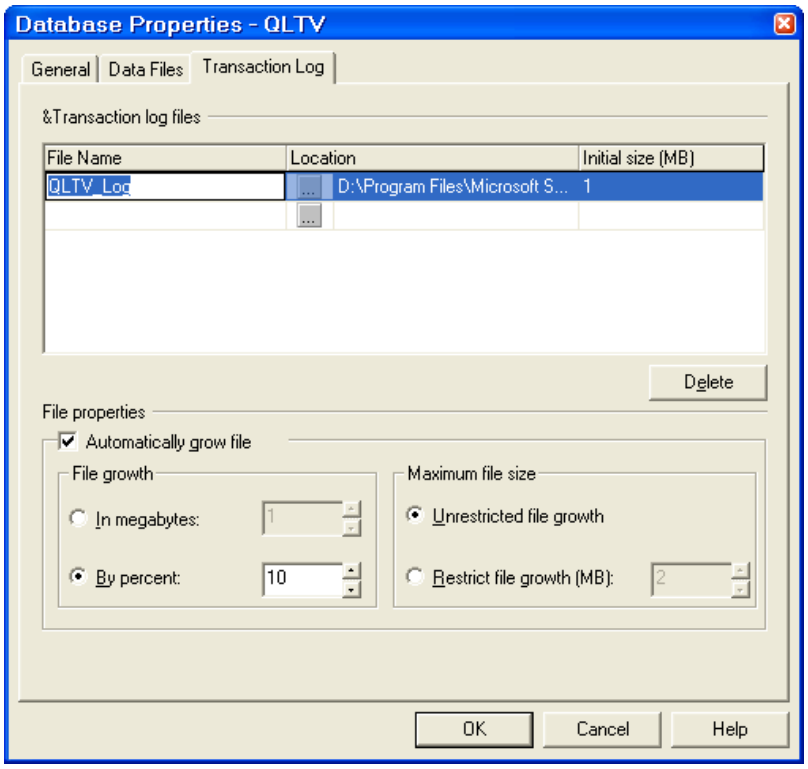
- Nhập tên CSDL.



- Xác định tên logic, tên vật lý, tên nhóm của tập tin và các tham số khác.



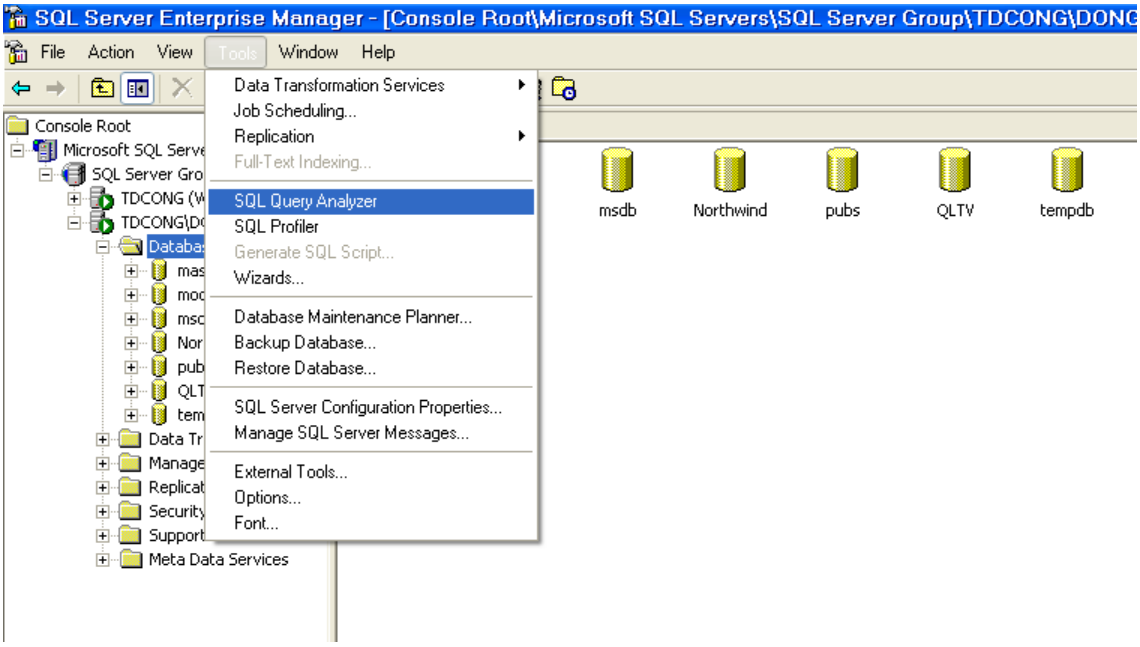
- Xác định tên logic, vật lý, tham số khác tập tin nhật ký.



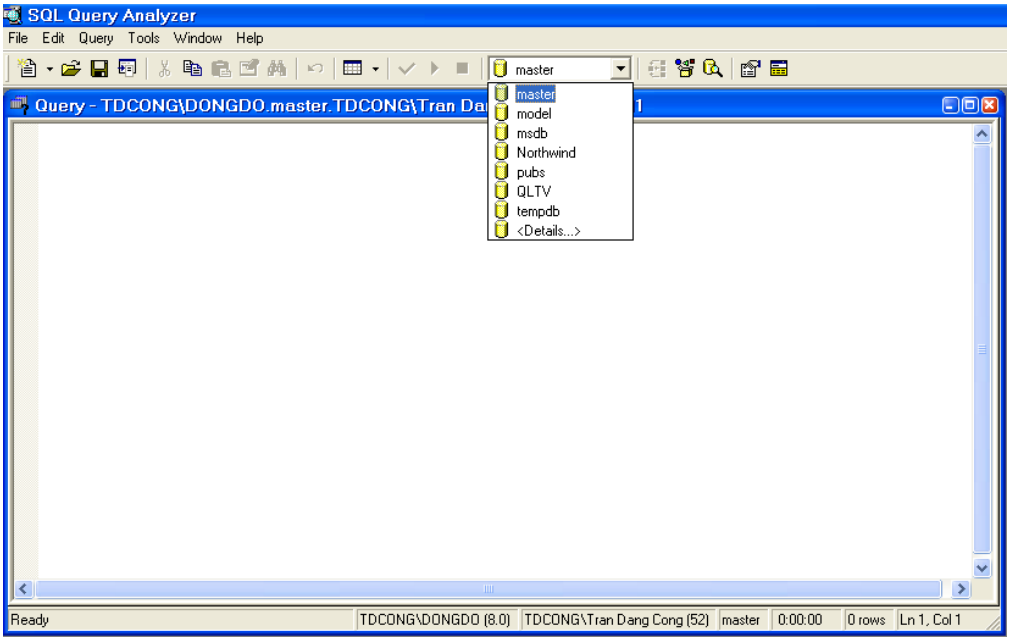
Tạo theo câu lệnh.

Sử dụng câu lệnh Create Database tạo CSDL, công cụ thực hiện lệnh:

- Trong Enterprise Manager -> Databases -> Tools -> SQL Query Analyzer



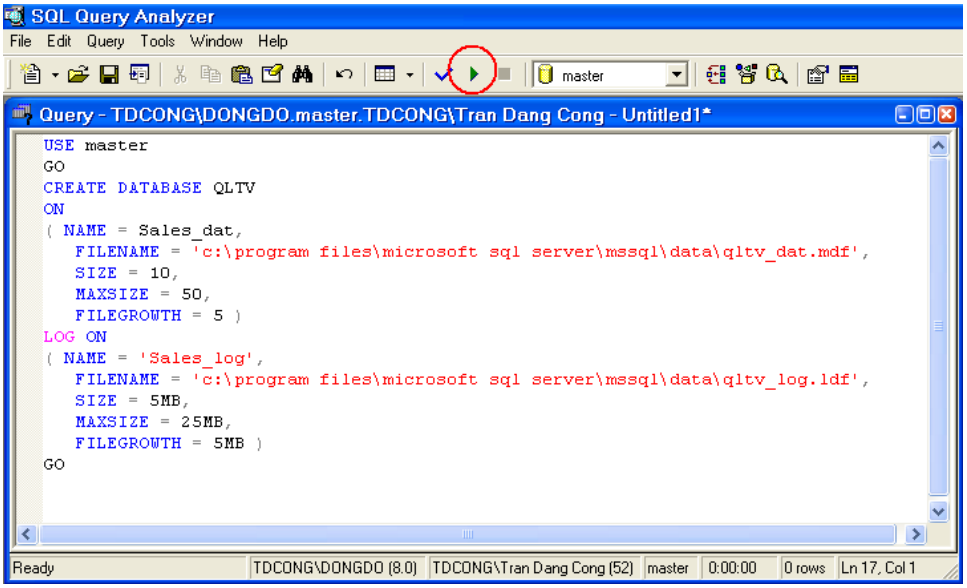
- Chọn CSDL Master.



- Soạn lệnh trong cửa sổ lệnh:

```
USE master
GO
CREATE DATABASE QLTV
ON
( NAME = Sales_dat,
            FILENAME    =    'c:\program    files\microsoft    sql
server\mssql\data\qltv_dat.mdf',
   SIZE = 10,
   MAXSIZE = 50,
   FILEGROWTH = 5 )
LOG ON
( NAME = 'Sales_log',
            FILENAME    =    'c:\program    files\microsoft    sql
server\mssql\data\qltv_log.ldf',
   SIZE = 5MB,
   MAXSIZE = 25MB,
   FILEGROWTH = 5MB )
GO
```

- Nhấn F5 hoặc nút thực hiện.



Công cụ SQL Query Analyzer cho phép bạn thực hiện từng câu lệnh bằng cách bôi đen vào đoạn lệnh cần thực hiện sau đó nhấn F5 hoặc nút thực hiện.

Khi tạo CSDL mới thì ạn phải đứng ở vị trí CSDL Master, khi muốn thực hiện lệnh với một CSDL cụ thể đã có nào đó bạn phải chọn vào CSDL đó và thực hiện lệnh.

Trong đoạn lệnh trên lệnh User Master thực hiện chọn CSDL Master bằng câu lệnh; lệnh Use xác định CSDL thực hiện.

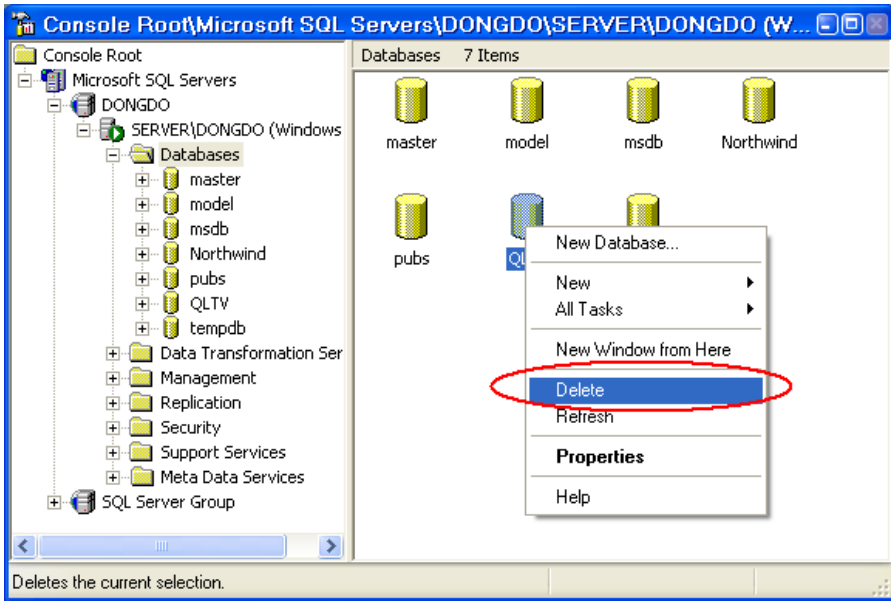
Lệnh Go xác định câu lệnh kết thúc và bắt đầu câu lệnh khác, câu lệnh được hiểu là dòng lệnh. Trong lệnh T-SQL một số lệnh khác nhau vẫn có thể nằm trên một dòng lệnh nên trong một số tình huống kich bản câu lệnh không cần sử dụng lệnh Go.

Xóa cơ sở dữ liệu.

Xóa theo công cụ.

- Chọn vào CSDL.

- Nhấn nút phải chuột -> Delete.



- Chọn Yes.

Xóa theo câu lệnh.

- Sử dụng lệnh Drop Database

```
Drop Database QLTV
```

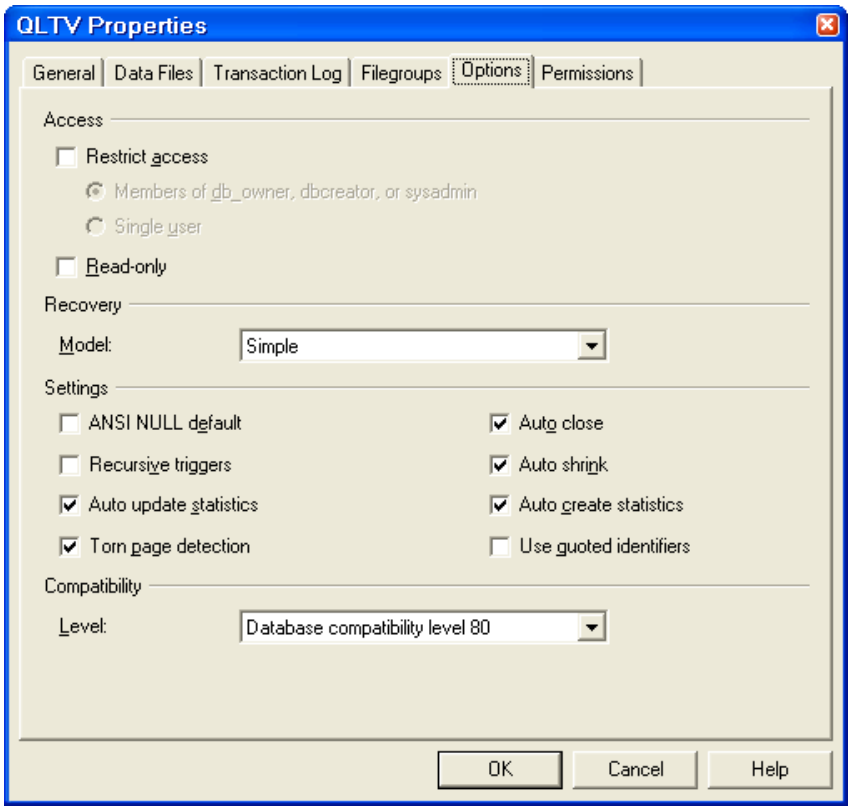
Sửa tham số.

Sửa theo công cụ.

- Chọn CSDL.

- Nhấn nút phải chuột

- Chọn Properties.



- Thay đổi tham số khi cần thiết:

+ Restrict access: Ngăn truy nhập.

+ Read only: Đặt thuộc tính chỉ đọc.

Sửa theo câu lệnh.

- Sử dụng câu lệnh Alter Database, ví dụ sau thực hiện thêm tập tin secondary vào CSDL.

```
ALTER DATABASE QLTV
ADD FILE
(
 NAME = QLTV_newfile,
 FILENAME = 'c:\Program Files\Microsoft SQL
Server\MSSQL\Data\newf.ndf',
 SIZE = 5MB,
 MAXSIZE = 100MB,
 FILEGROWTH = 5MB
)
GO
```

- Sửa tham số dựa vào thủ tục hệ thống sp_dboption:

+ Sửa đổi thuộc tính read only:

```
USE master
EXEC sp_dboption 'qltv', 'read only', 'TRUE'
```

+ Sửa thuộc tính autoshring

:

```
USE master
EXEC sp_dboption 'qltv', autoshring, TRUE
```

+ Sửa thuộc tính single user:

```
USE master
EXEC sp_dboption 'qltv', single_user
```

Mọi câu lệnh liên quan bạn có thể tra cứu, tham khảo trong Book Onlines.

BẢNG DỮ LIỆU – TABLE

CÁC CHUẨN TẮC.

Trong thiết kế cơ sở dữ liệu, việc tuân thủ ngặt ngèo những chuẩn là việc hết sức quan trọng, nó giúp cho việc quản trị dữ liệu có hiệu quả, khắc phục dư thừa, thuận lợi trong quản trị dữ liệu lớn, hiệu quả với dữ liệu phức tạp.

Gồm 3 chuẩn cơ bản:

Chuẩn thứ nhất.

Chuẩn thứ nhất xác định cấu trúc của một bảng không thể chứa các trường lặp lại.

Ta có thể lấy ví dụ như sau giả sử muốn lưu trữ thông tin một quyển sách, mỗi quyển sách có thể có một hoặc nhiều tác giả tham gia biên soạn, nếu không tuân theo chuẩn thứ nhất như nếu trên thì trong một bảng dữ liệu sách có thể có nhiều trường dữ liệu xác định thông tin tác giả.

ID	Tên sách	NXB	Tác giả 1	Tác giả 2

Trong ví dụ trên bạn nhận thấy thống tin Tác giả được lặp lại 2 lần, khắ phục bằng cách tạo ra một bảng lưu trữ danh sách tác giả của sách (sẽ bàn trong chuẩn sau).

Chuẩn thứ hai.

Chuẩn thứ hai xác định trong các hàng dữ liệu, mỗi cột đều phụ thuộc vào cột khóa chính. Ta xem xét một trường hợp vi phạm chuẩn thứ hai như sau:

Giả sử xét tình huống sinh viên mượn sách trong một thư viện, việc mượn sách được nhật ký theo bảng như sau:

Id_sach	Id_Sinhvien	Ngày mượn	Sức khỏe sinh viên

Xem xét trong bảng trên ta thấy mỗi hàng phụ thuộc vào khóa id_sach và id_sinhvien, nhưng thông tin Sức khỏe sinh viên không phụ thuộc vào id_sach, nên thông tin này cần chuyển sang bảng về thông tin của sinh viên.

Chuẩn thứ ba.

Chuẩn thứ ba xác định bản ghi tuân thủ theo chuẩn thứ hai và không có bất kỳ phần phụ thuộc chuyển tiếp nào. Phần phụ thuộc chuyển tiếp tồn tại khi một bảng chứa một cột đặc trưng. Cột này không phải là khóa nhưng vẫn xác định các cột khác.

Ta xem xét một ví dụ vi phạm chuẩn như sau:

Giả sử trong thư viện có một bảng liệt kê sách tồn trong kho, khi sinh viên mượn sách số lượng sách mà sinh viên mượn sẽ tăng, nếu nhật ký mượn sách được thực hiện theo bảng sau:

Id_sach	Id_Sinhvien	Ngày mượn	Số lượng đã mượn

Bảng trên bạn thấy mỗi lần sinh viên mượn sách số lượng sách có mã id_sach mà sinh viên có mã id_sinhvien sẽ tăng lên và tổng số là Số lượng đã mượn, thông tin này là thông tin tích lũy theo id_sach, id_sinhvien, ngày mượn.

Theo bảng trên ta thấy không vi phạm chuẩn thứ hai nhưng vi phạm chuẩn thứ ba vì cột Số sách đã mượn là cột phụ thuộc chuyển tiếp, cột này cần phải được chuyển sang bảng khác là bảng Sinh viên mượn sách:

Id_sach	Id_Sinhvien	Số lượng đã mượn

Khi nào cần chuẩn tắc.

Một cơ sở dữ liệu cần được chuẩn tắc khi:

- Dữ liệu lớn, phân tán.

- Không xác định rõ nhóm dữ liệu.

- Dữ liệu phức tạp.

- Bước đầu tiên khi xây dựng ứng dụng.

Khi nào không cần chuẩn tắc hóa.

Một số tình huống sẽ không cần chuẩn tắc hóa, nếu theo như thiết kế theo chuẩn thì việc đưa ra một mẫu tin truy vấn có thể phải thực hiện truy xuất từ nhiều bảng với nhau, điều này có nghĩa ta phải thực hiện kết hợp các bảng với nhau (tuy theo luật) nên thời gian truy xuất có thể rất lớn mà yêu cầu thực tế đặt ra trong tình huống này là phải nhanh, thì truy xuất theo một bảng đã có sẵn

là nhanh hơn, sau đây là một số trường hợp không cần chuẩn tác hóa (tùy theo tình huống):

- Thông tin tính toán.

- Thông tin sự kiện.

- Sự phân hoạch.

THIẾT KẾ BẢNG DỮ LIỆU.

Table (bảng dữ liệu) là một thành phần cơ bản của CSDL, một CSDL được thiết kế từ một hoặc nhiều bảng dữ liệu, mỗi bảng dữ liệu được cấu trúc từ các hàng và cột dữ liệu, mỗi hàng dùng mô tả một đối tượng, vấn đề, sự kiện,... cột thể hiện thuộc tính của các đối tượng, sự kiện,... của hàng. Dữ liệu cùng cột có cùng kiểu (data type). Ngoài các hàng, cột bảng còn có các khóa, liên kết, ràng buộc,...

Trước khi bắt tay vào thiết lập bảng dữ liệu trước hết ta phải xác định xem bảng sẽ xây dựng như thế nào, dựa trên một số thông tin sau:

- Kiểu dữ liệu trong bảng.

- Các cột, kiểu dữ liệu tương ứng (và độ dài nếu cần thiết).

- Cột nào cho phép giá trị NULL (là giá trị mà phần dữ liệu thuộc hàng, cột xác định không được gán giá trị nào, vì vậy nên 2 phần tử có cùng giá trị NULL là không bằng nhau).

- Giá trị ngầm định (là giá trị mà khi chưa nhập vào nó nhận giá trị này).

- Chỉ số Index, khóa chính, khóa ngoài.

Kiểu dữ liệu.

SQL Server gồm những kiểu dữ liệu sau:

Binary: Là kiểu dữ liệu chứa dạng số ở hệ hexa, gồm 3 kiểu dữ liệu Binary, Varbinary, Image.

Text: Là kiểu ký tự, chứa chữ cái, ký hiệu, số, gồm những kiểu dữ liệu sau:

- Char: Kiểu ký tự, khi xác định độ dài thì độ dài trong CSDL sẽ xác định theo độ dài đặt trước mà không theo độ dài dữ liệu thực có, không sử dụng với ký tự dạng Unicode, độ dài tối đa là 8000.

- Nchar: Tương tự như Char nhưng sử dụng với ký tự Unicode, độ dài tối đa 4000.

- Nvarchar: Tương tự như NChar nhưng kích thước trong CSDL sẽ là kích thước thực dữ liệu hiện có, không tính theo kích thước đặt trước, kích thước tối đa là 4000.

- Varchar: Tương tự như Nvarchar nhưng không hỗ trợ Unicode.

- Text: Kiểu văn bản, chứa cả ký tự xuống dòng, lưu trữ theo dạng văn bản, có kích thước lớn, có thể lên đến vài Gb, cơ chế quản lý kiểu dữ liệu theo dạng con trỏ và cách thức chèn và cập nhật sẽ khác, kiểu dữ liệu này không hỗ trợ cho Unicode.

- Ntext: Tương tự như Text nhưng có hỗ trợ Unicode.

Data/Time: Kiểu dữ liệu ngày, thời gian, ngày và thời gian, gồm 2 kiểu:

- DateTime: Đầy đủ cả ngày và thời gian.

- SmallDataTime: Chỉ ngày hoặc thời gian.

Numeric: Dữ liệu kiểu số, gồm các kiểu dữ liệu sau:

- Int, smallint, tinyint, bigint: Số nguyên

- Float, real, decimal, numeric: Số thực.

Monetary: Tiền tệ:

- Money, Smallmoney.

Bit: Kiểu số 0, 1.

Sql_variant: Là kiểu dữ liệu xác định theo kiểu dữ liệu khác, một cột dữ liệu được định nghĩa dữ liệu kiểu này có thể lưu trữ nhiều dữ liệu có kiểu khác nhau trong cùng một bảng. Ví dụ có thể lưu trữ nhiều kiểu dữ liệu **int**, **binary**, **char**, nhưng không chứa dữ liệu kiểu **text**, **ntext**, **image**, **timestamp**, **sql_variant**.

Timestamp: Là kiểu dữ liệu có kích thước 8 byte, lưu trữ dạng số nhị phân do hệ thống tự sinh ra, mỗi giá trị timestamp trong CSDL là duy nhất.

Table: Là kiểu dữ liệu đặc biệt lưu trữ tập hợp các hàng (dạng bảng), mục đích sử dụng chính là lưu trữ tạm thời tập hợp các hàng sau truy vấn.

Text in row.

Như xem xét trước, dữ liệu kiểu char, varchar có độ dài tối đa là 8000byte, dữ liệu kiểu text, ntext có 2 kiểu lưu trữ: lưu trữ trực tiếp, lưu trữ quản lý theo kiểu con trỏ.

- Đối với lưu trữ theo kiểu trực tiếp, kích thước tối đa đối với text là 8000, đối với ntext là 4000 (kích thước 1 ký tự ở mã Unicode là 2 byte, mã không Unicode là 1 byte).

- Lưu trữ, quản lý theo con trỏ kích thước lên đến GB.

Để lưu trữ dữ liệu theo kiểu con trỏ đầu tiên ta phải đặt chức năng **Text in row** về trạng thái **On**, thuộc tính này hiệu ứng cả với kiểu dữ liệu image.

Sử dụng thủ tục sp_tableoption để thay đổi thuộc tính, thuộc tính thay đổi theo bảng dữ liệu.

Giả sử bật chức năng text in row như sau:

```
Sp_tableoption N'TacGia', 'text in row', 'ON'
```

Tắt chức năng text in row như sau:

```
Sp_tableoption N'TacGia', 'text in row', 'OFF'
```

Để cập nhật dữ liệu khi thuộc tính được bật, ta phải dùng lệnh READTEXT, UPDATETEXT, WRITETEXT (sẽ bàn kỹ câu lệnh này sau).

Auto number.

Đặt cột dữ liệu kiểu số, tăng tự động khi một hàng được thêm, cột kiểu này không sửa dữ liệu. Dữ liệu kiểu này tương ứng với việc khi thêm hàng dữ liệu chèn thêm giá trị dạng số theo hàm NewID().

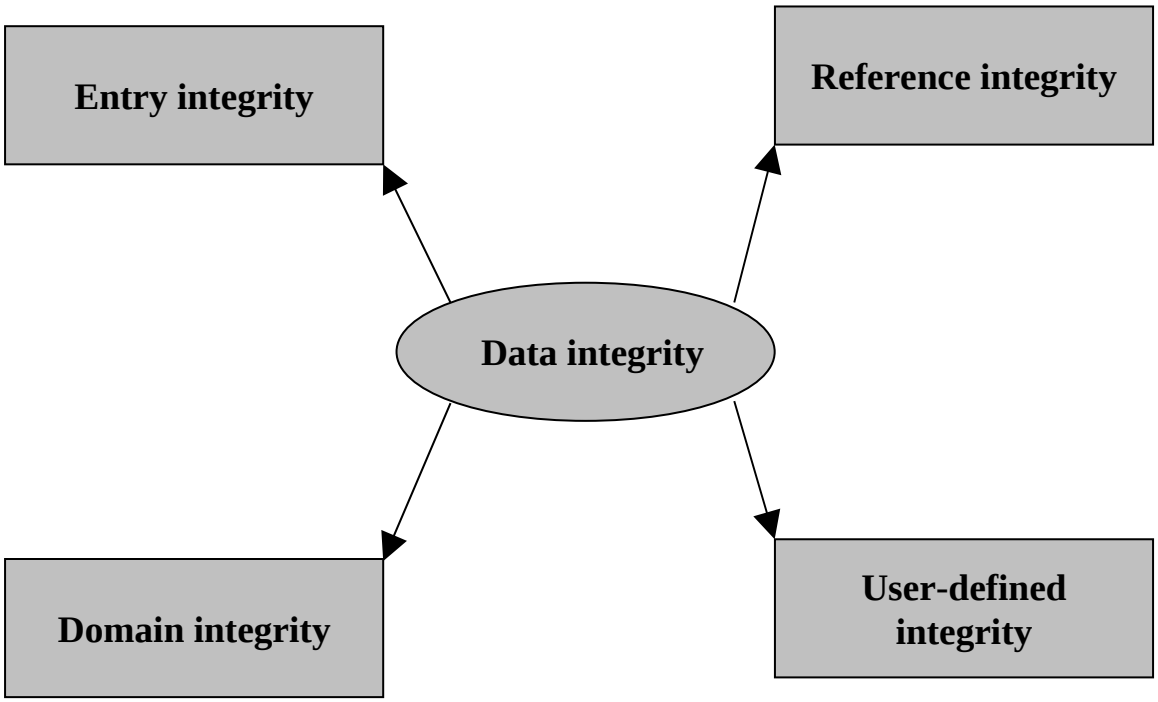
Ràng buộc dữ liệu.

Để có một CSDL khi lưu trữ dữ liệu có độ tin cậy, độ chính xác cao, nhanh và thuận tiện trong khai thác dữ liệu thì toàn vẹn dữ liệu là vấn đề hết sức quan trọng. Khi ràng buộc được thiết lập, dữ liệu khi nhập vào CSDL sẽ được kiểm soát, độ tin cậy thông tin sẽ được bảo đảm.

Có nhiều kiểu ràng buộc dữ liệu, một CSDL có thể gồm một hoặc nhiều ràng buộc, ràng buộc có thể trên một bảng, trên nhiều bảng.

Toàn vẹn dữ liệu chia thành 4 loại:

- Toàn vẹn thực thể (Entry integrity): Mỗi thực thể đều được xác định theo một khóa, khi biết khóa ta hoàn toàn có thể xác định được thực thể tương ứng. Khóa như vậy coi là khóa chính.
- Toàn vẹn theo miền (Domain integrity): Là loại toàn vẹn có hiệu ứng với các cột dữ liệu trong một phạm vi nào đó, ví dụ kiểu dữ liệu cũng là một dạng của toàn vẹn miền, ràng buộc theo khóa check cũng là toàn vẹn theo miền.
- Toàn vẹn dạng tham chiếu (Referential integrity): Khi một bảng có quan hệ với một bảng khác theo một mối quan hệ, trong mối quan hệ đó sẽ có một khóa chính (như phần toạn vẹn thực thể) và một khóa ngoài, khóa ngoài sẽ là khóa tham chiếu của khóa chính, giá trị của khóa ngoài sẽ thuộc tập các giá trị của khóa chính hoặc giá trị NULL. Ràng buộc kiểu quan hệ (Relationship) gọi là toàn vẹn kiểu tham chiếu.
- Toàn vẹn do người dùng định nghĩa (User-defined integrity): Là toàn vẹn do người dùng định nghĩa, quy định dữ liệu nhập vào theo quy cách, giá trị được kiểm soát chặt chẽ, toàn vẹn kiểu này cũng có thể xây dựng trên cơ sở các toàn vẹn trước.



Bốn loại toàn vẹn nói trên ta có thể thống kê tương ứng với các khóa, quy tắc, ràng buộc trong SQL Server như sau:

Kiểu toàn vẹn	Công cụ trong SQL Server
Entry integrity	1. Ràng buộc Primary key 2. Ràng buộc Unique 3. Cột Identity
Domain integrity	1. Giá trị ngầm định Default 2. Ràng buộc khóa ngoài Foreign Key 3. Ràng buộc Check 4. Thuộc tính NOT NULL
Referential integrity	1. Ràng buộc Forelgn Key 2. Ràng buộc Check
User-defined integrity	1. Rules 2. Stored procedures 3. Triggers

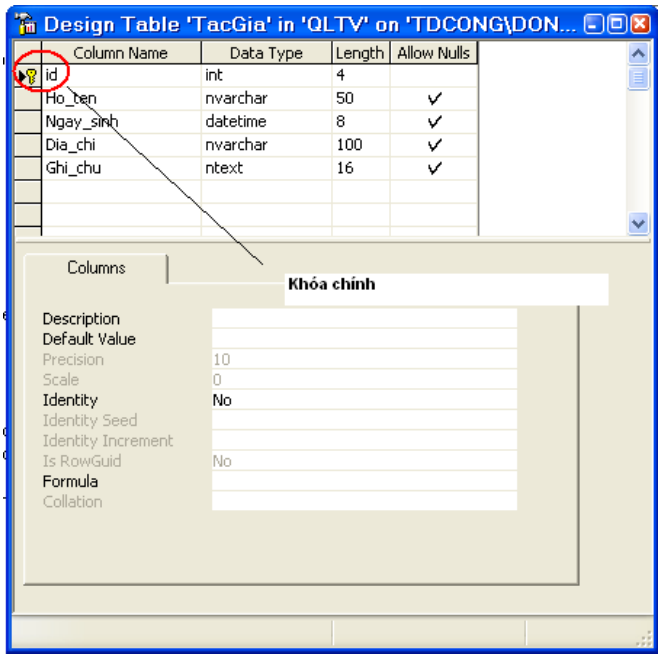
Các khóa.

Khóa chính – Primary Key.

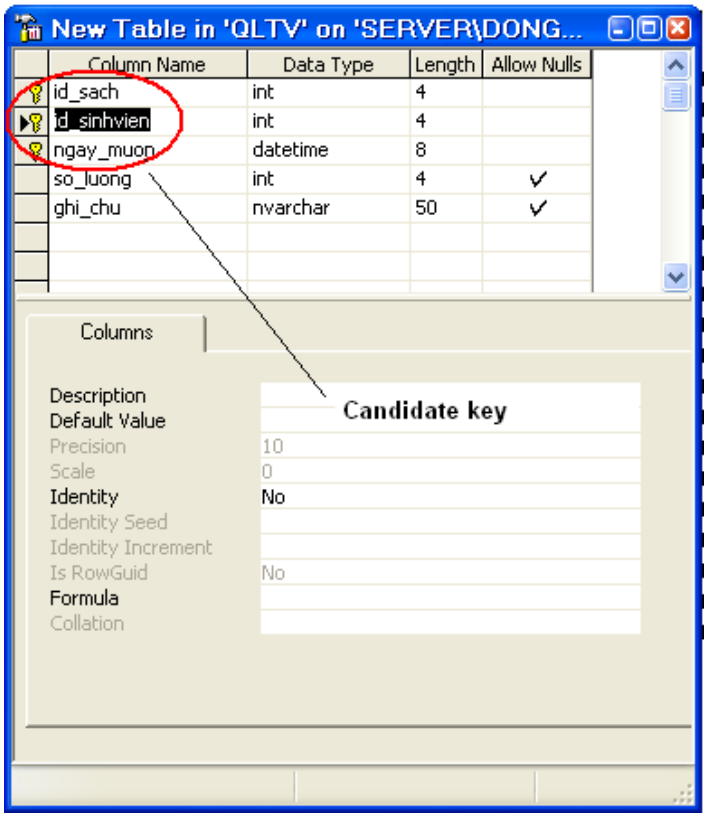
Là một hoặc tổ hợp nhiều cột dữ liệu xác định duy nhất trong một bảng, giá trị khóa chính luôn khác NULL.

Ví dụ: Bảng danh sách tác giả viết sách.

Trong ví dụ trên bảng dữ liệu có khóa chính là một cột dữ liệu id, khi cột xác định là khóa chính bên cạnh xuất hiện biểu tượng chìa khóa, thuộc tính Allow Nulls không được đanh dấu.


Design Table 'TacGia' in 'QLTV' on 'TDCONG\DON...
Column Name
Data Type
Length
Allow Nulls
id
int
4
Ho_ten
nvarchar
50
Ngay_sinh
datetime
8
Dia_chi
nvarchar
100
Ghi_chu
ntext
16
Columns
Khóa chính
Description
Default Value
Precision
10
Scale
0
Identity
No
Identity Seed
Identity Increment
Is RowGuid
No
Formula
Collation

Ví dụ: Bảng dữ liệu lưu trữ thông tin nhật ký mượn sách.



Trong ví dụ trên bảng dữ liệu có khóa chính được tổ hợp từ 3 cột dữ liệu id_sach, id_sinhvien, ngay_muon, ba cột trên xác định duy nhất một sinh viên được mượn một loại sách trong một ngày (giả sử quy chế xác định như vậy), các cột tham gia khóa chính gọi là candidate key.

Khóa ngoài.

Theo chuẩn thiết kế CSDL, khi lưu trữ thông tin sách phải có một cột chứa thông tin nhà xuất bản. Một nhà xuất bản có thể xuất bản nhiều quyển sách và một quyển sách chỉ xuất bản ở một nhà xuất bản. Nên trong thiết kế ta phải có:

+ Bảng dữ liệu lưu trữ danh sách các nàh xuất bản: Có khóa chính đại diện cho nhà xuất bản.

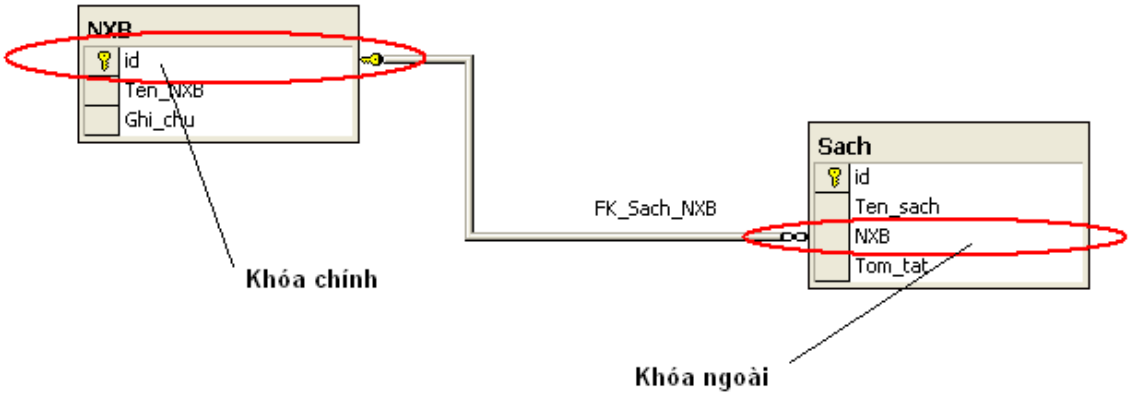
+ Bảng dữ liệu lưu trữ sách: Có chứa thông tin nhà xuất bản.

+ Quan hệ giữa nhà xuất bản và sách: Mã khóa nhà xuất bản thuộc bảng nhà xuất bản và thông tin nhà xuất bản thuộc bảng sách, cột thông tin nhà xuất bản thuộc bảng sách tham gia quan hệ trên gọi là khóa ngoài (Foreign key).

Cột dữ liệu là khóa ngoài có thể có quan hệ với nhiều khóa chỉnh ở nhiều bảng, một bảng có thể có nhiều khóa ngoài, khóa ngoài có thể có giá trị NULL,

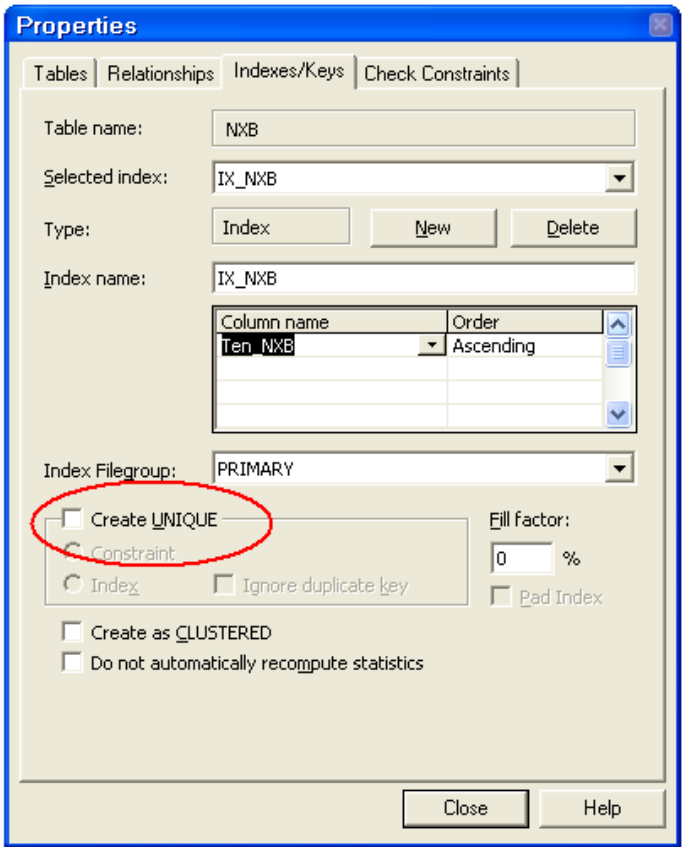
giá trị của khóa ngoài luôn nằm trong tập giá trị của khóa chính trong mối quan hệ đã thiết lập.

Khóa ngoài và khóa chính phải có cùng kiểu dữ liệu, cùng kích thước.



Ràng buộc Unique.

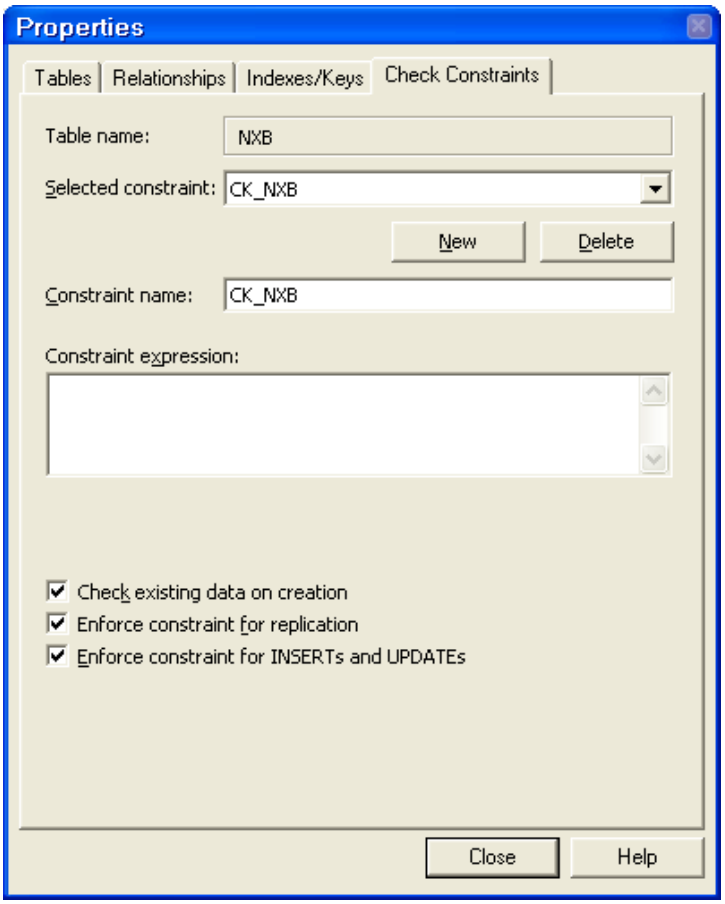
Unique là ràng buộc xác định trên một hoặc tổ hợp cột dữ liệu, cột hoặc tổ hợp cột dữ liệu được xác định ràng buộc loại này là duy nhất.



Một bảng dữ liệu có thể có nhiều ràng buộc duy nhất, một cột trongmrang buộc loại này cho phép nhận giá trị NULL, rang buộc duy nhất có thể sử dụng làm tham chiếu cho khóa ngoài.

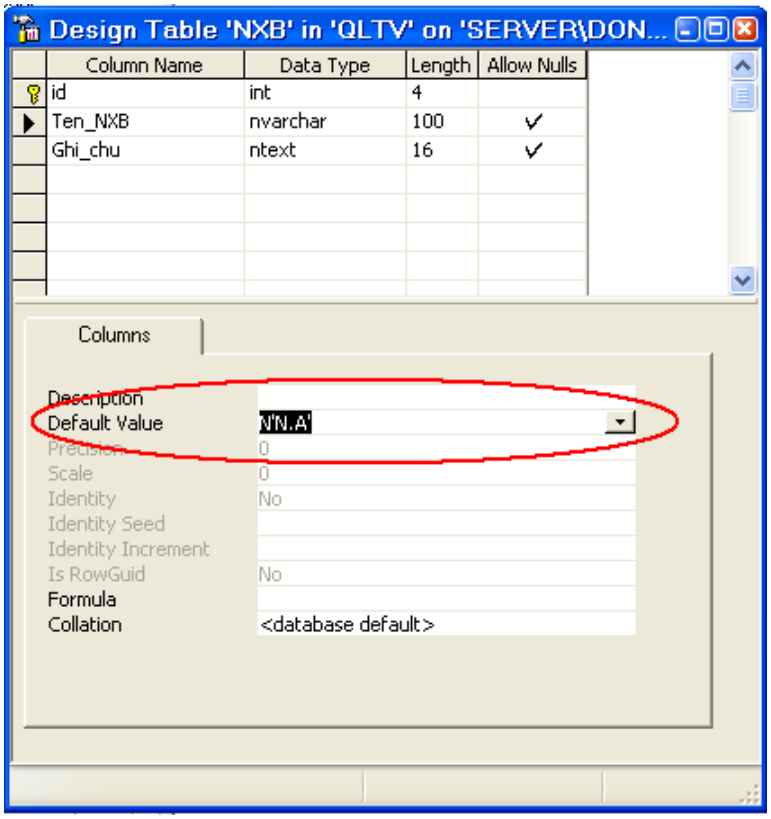
Ràng buộc Check.

Là ràng buộc khống chế dữ liệu nằm trong một phạm vi nào đó. Ràng buộc này sẽ kiểm tra dữ liệu khi nhập vào.



Giá trị ngầm định – Default.

Giá trị gán cho cột dữ liệu khi thêm bản ghi và chứa nhập dữ liệu vào cột này.



TẠO BẢNG DỮ LIỆU.

Sau khi đã xác định đầy đủ các thông tin thiết kế CSDL, bước tiếp theo là thực hiện tạo cấu trúc CSDL. Để tạo cấu trúc CSDL bước quan trọng là tạo bảng dữ liệu.

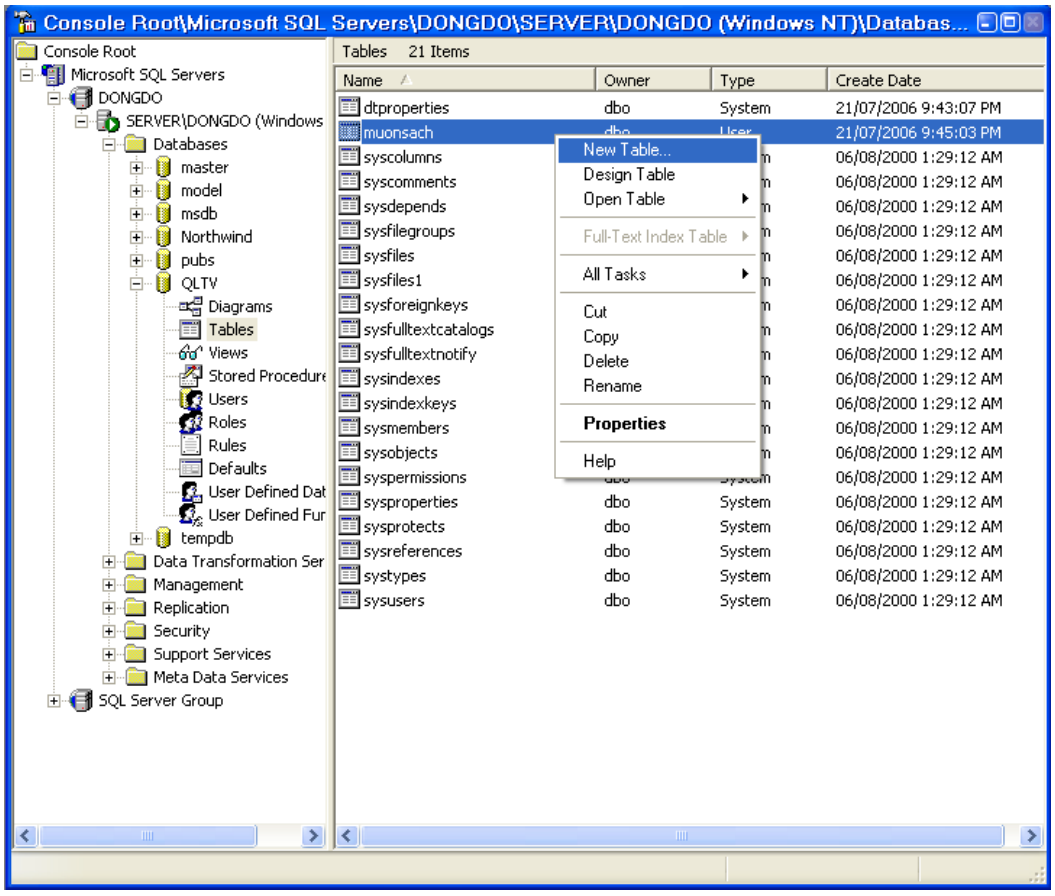
Khi tạo CSDL hệ thống tự động tạo ra một số bảng dữ liệu ngầm định, các bảng dữ liệu này sẽ cung cấp, quản lý thông tin quản trị của CSDL, cung cấp một số hàm hệ thống trợ giúp bgười dùng.

Tạo bằng công cụ.

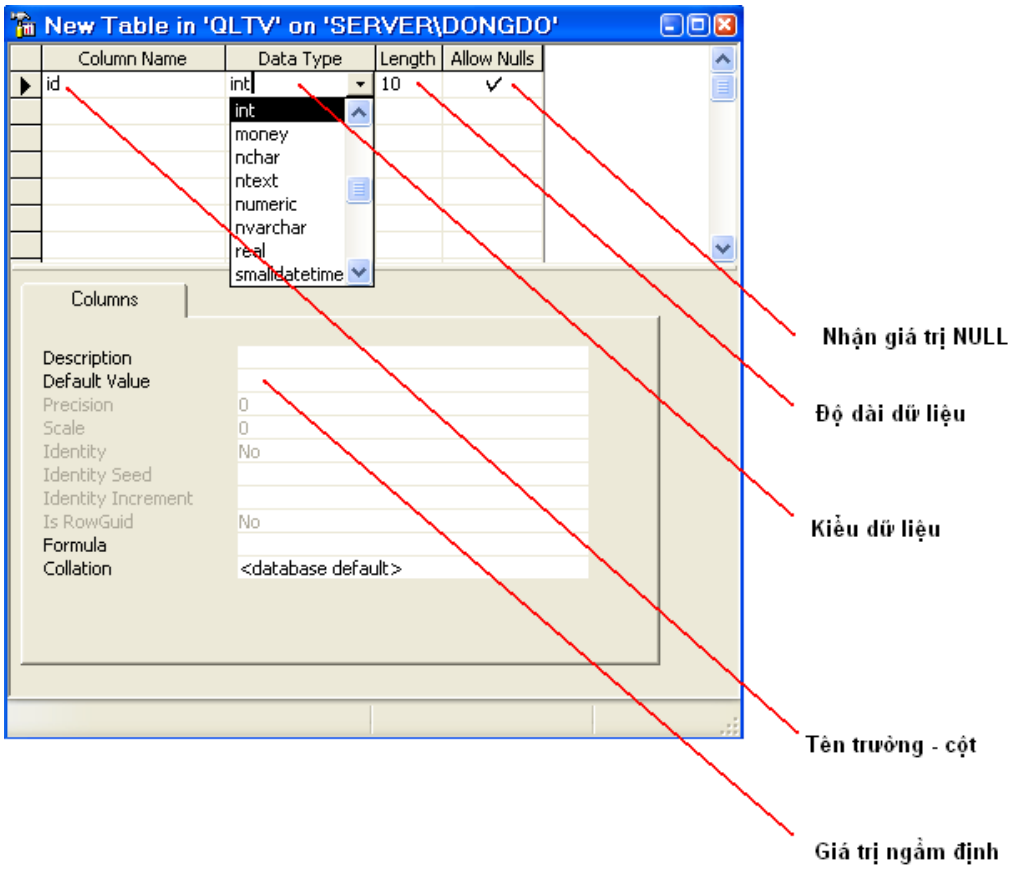
- Chọn CSDL

- Chọn Tables

- Nhấn phải chuột ở cửa sổ bên phải

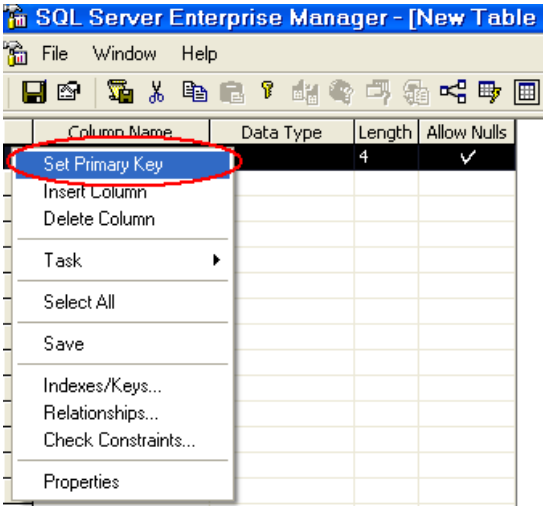


- Chọn New Table.



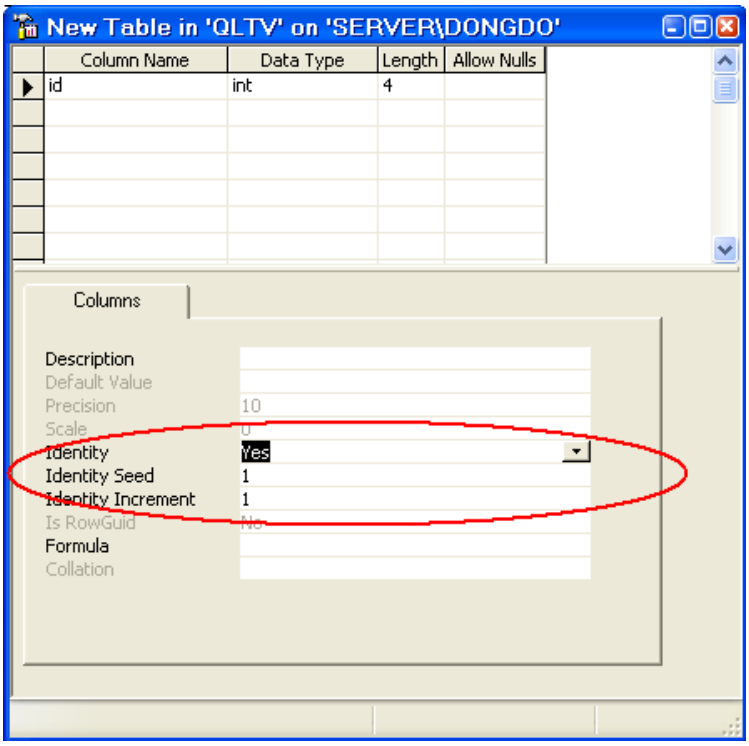
Đặt khóa chính.

Để xác định khóa chính ta thực hiện chọn những cột tham gia khóa bằng cách giữ phím shift và chọn chuột -> nhấn chuột phải -> chọn Set primary key.



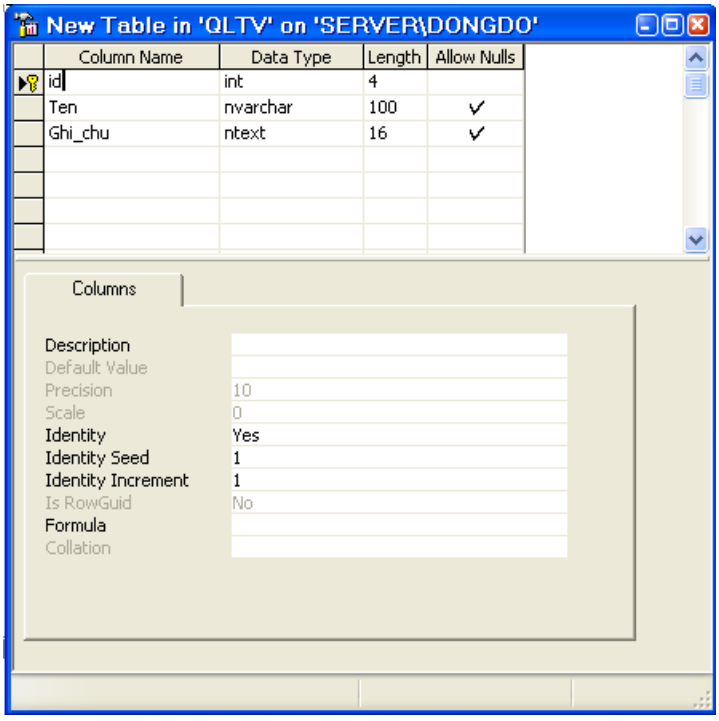
Xác định Identity.

- Chọn cột dữ liệu -> Chọn yes trong mục Identity -> đặt seed (giá trị khởi đầu) -> đặt increment (bước tự động tăng).



Tạo bảng bằng câu lệnh.

Giả sử cần tạo bảng tên NXB có cấu trúc như sau:



Sử dụng lệnh Create table, kịch bản câu lệnh như sau:

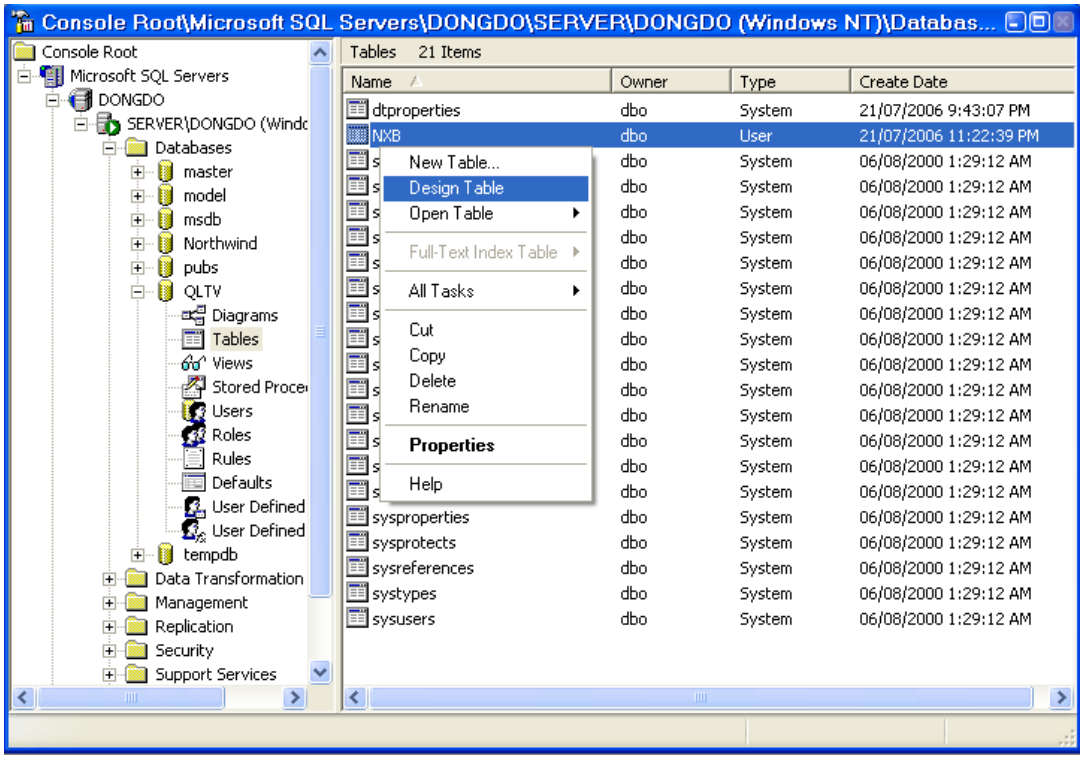
```
Create table NXB(id int not null primary key identity(1,1), Ten Nvarchar(100), Ghi_chu Ntext)
```

Sửa cấu trúc bảng.

Sử dụng công cụ.

- Chọn bảng cần sửa đổi của CSDL.

- Nhất phải chuột -> chọn Design Table.



- Thực hiện sửa cấu trúc bảng.

Sửa dụng câu lệnh.

Để sửa cấu trúc bảng dữ liệu ta sử dụng câu lệnh Alter table.

- Thêm một cột vào bảng đã có:

```
ALTER TABLE NXB ADD Dia_chi NVARCHAR(100) NULL
```

- Xóa cột từ bảng đã có.

```
ALTER TABLE NXB_Drop column_Dia_chi
```

Xóa bảng.

Sử dụng công cụ.

- Chọn bảng
- Nhất chuột phải
- Chọn Delete -> Yes.

Sử dụng lênh. (Drop Table)

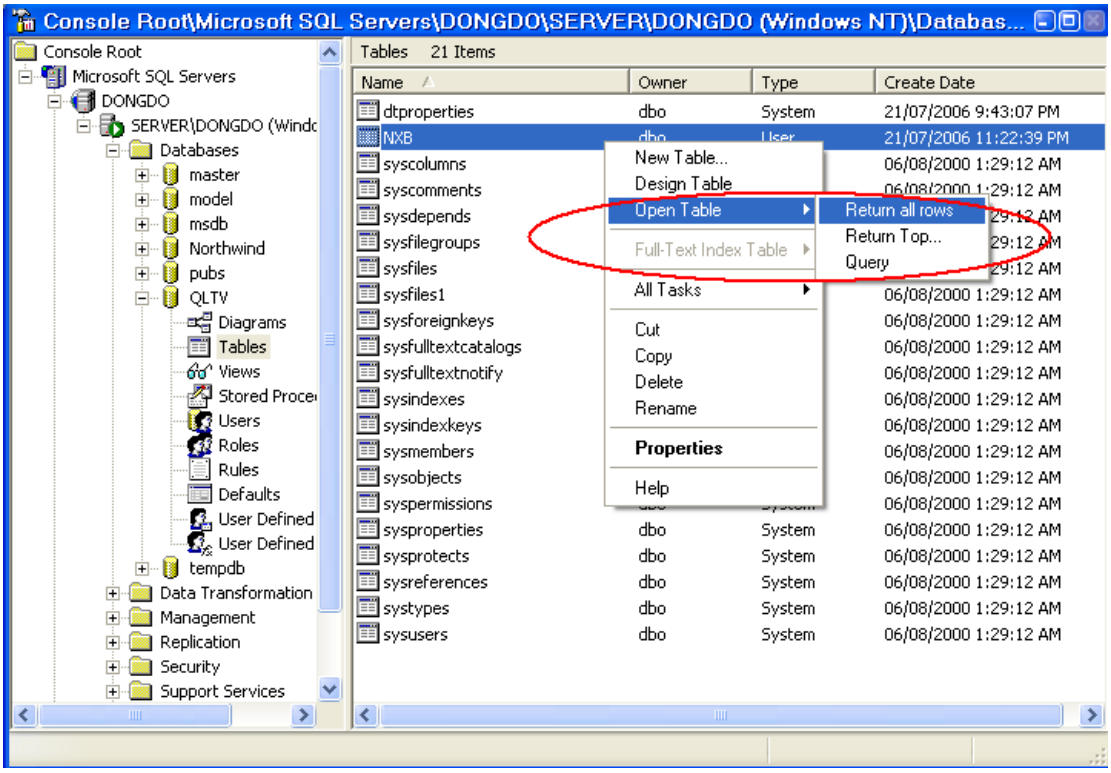
```
Drop Table NXB
```

Bảng dữ liệu có tham gia mối quan hệ Relationship khi xóa bạn cần chú ý: Nếu bảng chứa khóa ngoài thì việc xóa thực hiện bình thường, nếu bảng chứa khóa chính của mối quan hệ thì không xóa được.

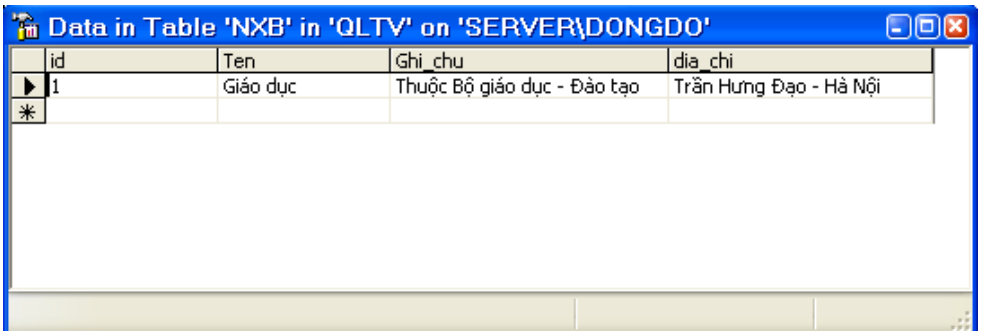
Nhập dữ liệu vào bảng.

Sử dụng công cụ.

- Chọn bảng dữ liệu
- Nhất chuột phải -> Open Table -> Return all rows



- Nhập dữ liệu theo đúng quy cách kiểu dữ liệu, ràng buộc nếu có.



Việc sửa, xóa được thực hiện trực tiếp. Đối với các cột là dạng số, tăng tự động không cần nhập dữ liệu. Để lưu lại dữ liệu đã nhập bạn di chuyển con trỏ sang hàng khác.

Sử dụng câu lệnh.

Sử dụng lệnh Inert into.

Insert into NXB(Ten, Dia_chi) values(N'Kim Đồng', N'hà Nội')

Nếu cột dữ liệu hỗ trợ Unicode thì trước giá trị nhập vào bạn phải thêm kèm ký tự N (như ví dụ trên).

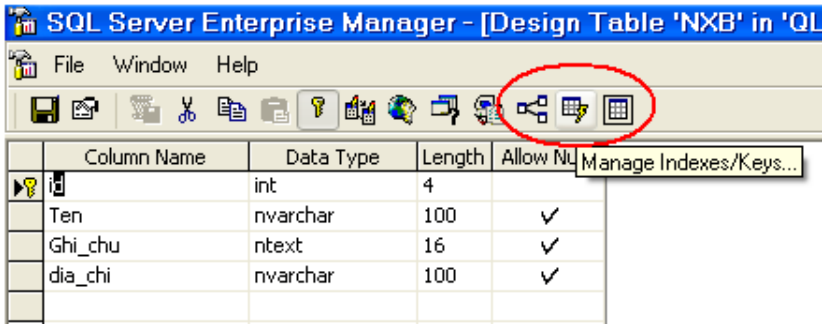
Tạo, sửa ràng buộc, khóa.

Phần này nhằm thực hiện thao tác với các rang buộc, khóa: ralationship, check, unique,...

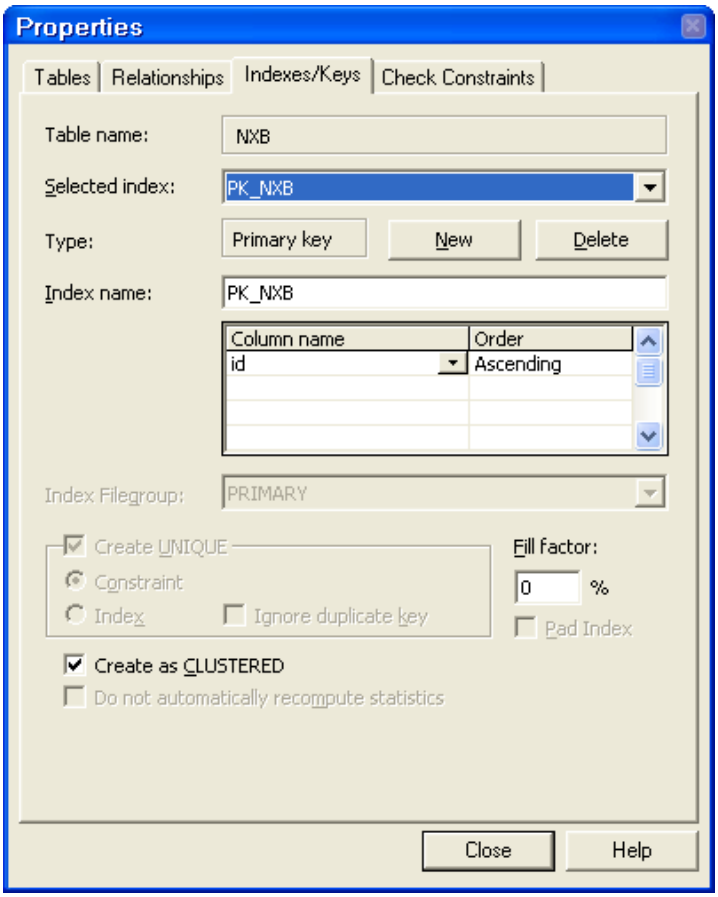
Sử dụng công cụ.

- Chọn chức năng Design table.

- Chọn biểu nút Manage Indexes/Keys...

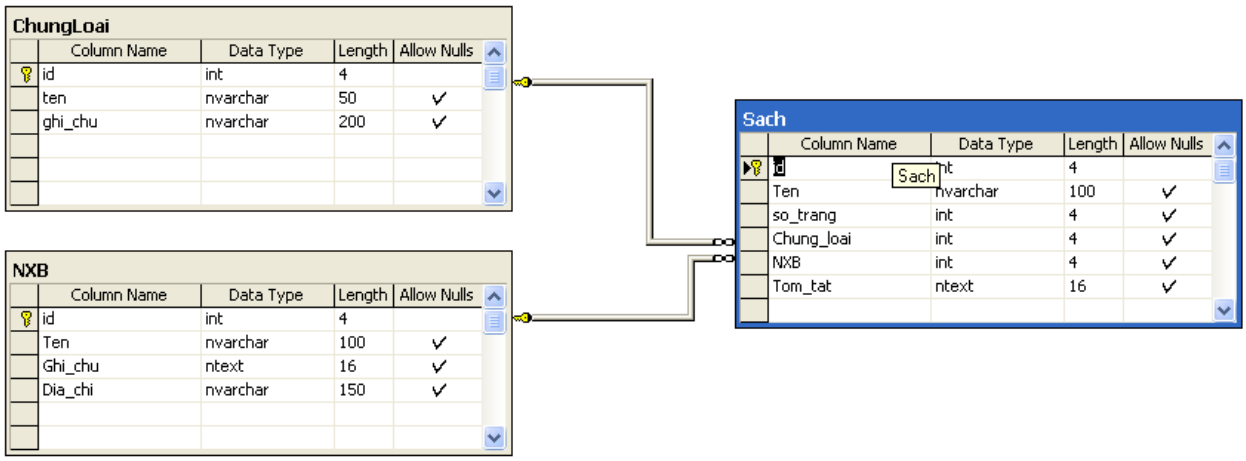


- Chọn bảng tương ứng.



Sử dụng câu lệnh.

Để cụ thể hơn ta thực hiện theo ví dụ có sơ đồ cấu trúc sau:

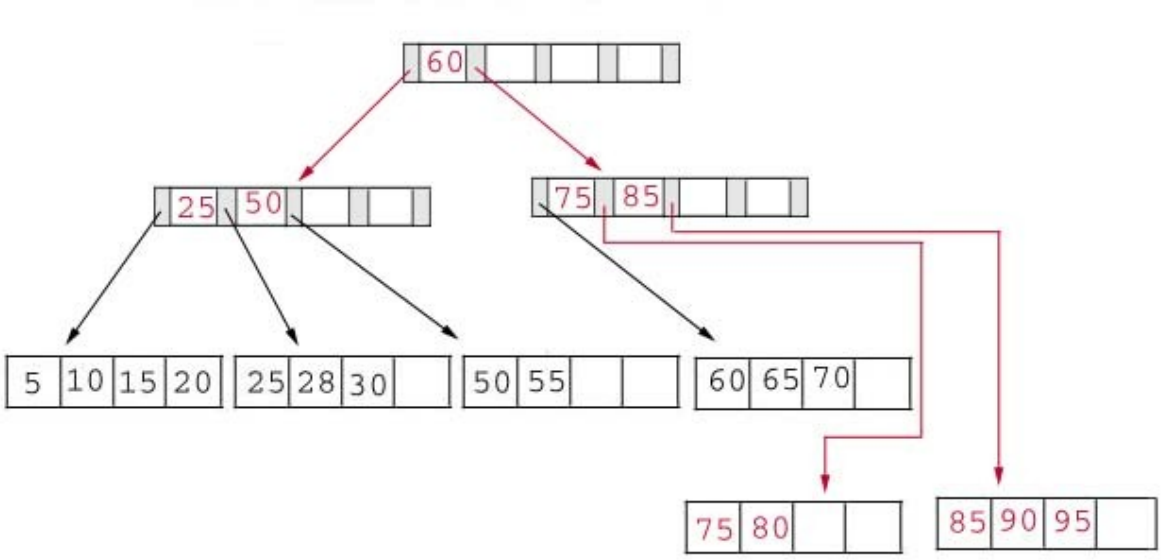


```
Create Table NXB(id int not null primary key identity(1,1), Ten Nvarchar(100), Ghi_chu Ntext, Dia_chi nvarchar(150))

Go

Create Table ChungLoai(id int not null primary key identity, ten nvarchar(50), ghi_chu nvarchar(200))

Go

Create Table Sach(id int not null primary key identity, Ten nvarchar(100), so_trang int default(0), Chung_loai int references Chungloai(id), NXB int references NXB(id), Tom_tat ntext)
```

KHÓA INDEX

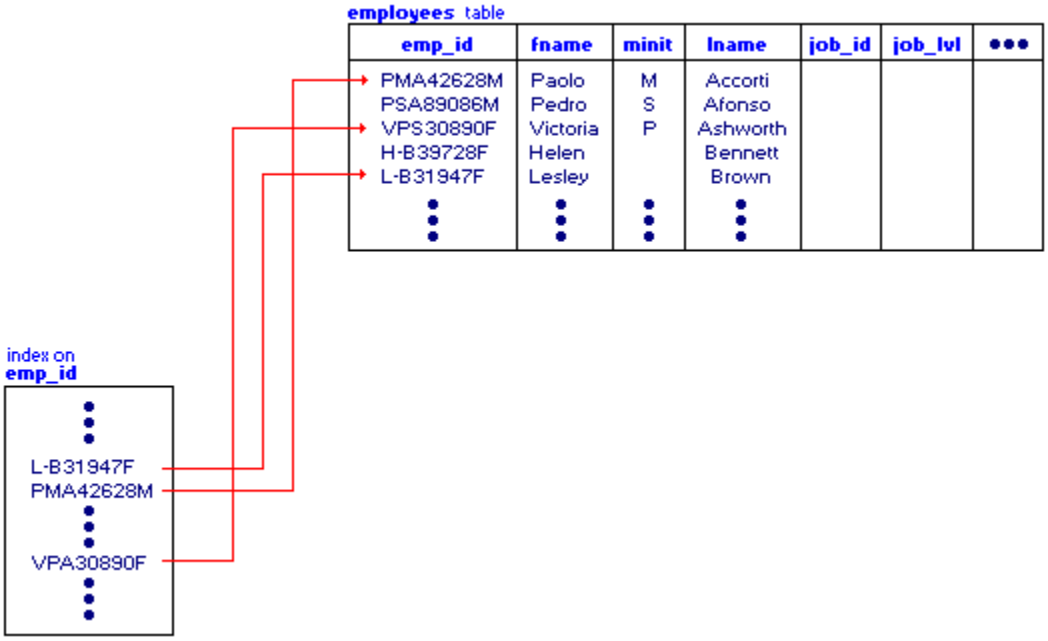
THIẾT KẾ KHÓA INDEX.

Index là một khóa quan trọng đối với CSDL đặc biệt là cơ sở dữ liệu lớn. Index được thiết lập từ một hoặc nhiều cột dữ liệu của bảng dữ liệu. các giá trị của khóa Index sẽ được sắp xếp và lưu trữ theo một danh sách (bảng khác). Mỗi giá trị trong khóa Index là duy nhất trong danh sách, mỗi giá trị khóa Index sẽ liên kết đến giá trị trong bảng dữ liệu (liên kết dạng con trỏ). Việc lưu trữ dữ liệu của bảng có khóa Index được thực hiện theo cấu trúc cấy B-Tree nhằm tăng tốc độ truy xuất dữ liệu đối với ổ đĩa (thiết bị thứ cấp).



Khi tìm kiểm một giá trị trong cột dữ liệu, mà cột này tham gia tạo khóa Index, đầu tiên câu lệnh xác định vị trí của giá trị nằm trong khóa Index bằng phép duyệt cây, sau đó thực hiện tìm theo liên kết đến bản ghi chứa giá trị tương ứng với khóa trong bảng.

Sơ đồ ví dụ dưới đây gồm khóa Index được tạo từ cột emp_id của bảng employees.



Việc thiết kế khóa Index dựa trên nhu cầu truy vấn, chèn dữ liệu trên một bảng, xác định dựa vào một số tham số sau:

+ Cột thường được sử dụng làm khóa truy vấn dữ liệu (xác định cột tham gia khóa Index).

+ Tập lệnh thường sử dụng try vấn cần tốc độ cao (xác định tập cột tham gia truy vấn).

+ Dữ liệu nhập vào bảng có khóa Index cần nhanh hơn hay truy vấn cần nhanh hơn (xác định đặt clustered hoặc nonclustered).

+ Lượng dữ liệu nhập đồng loạt nhiều hay ít (xác định tham số fillfactor).

Clustered Index.

Khi khóa đặt thuộc tính Clustered, dữ liệu của bảng sẽ được sắp xếp vật lý trên đĩa, như vậy khi thiết kế khóa dạng này dữ liệu được chèn và sẽ tìm đúng vị trí trên địa để lưu trữ (vùng đĩa dành cho bảng dữ liệu), chính vì vậy mà có thể xảy ra trường hợp phải dịch chuyển danh sách các giá trị đã có ở đĩa. Những việc tạo khóa Index dạng này sẽ không cần sắp xếp giá trị ở dạng logic mà khi truy nhập đĩa đã bảo đảm dữ liệu được sắo xếp.

Bảng dữ liệu chỉ có thể tạo tối đa một khóa Lustered Index.

Nonclustered Index.

Dữ liệu Index không sắp xếp ở dạng vật lý mà chỉ sắp xếp logic, dữ liệu của bảng lưu trữ giá trị khóa Index được sắp xếp, nhanh trong nhập dữ liệu.

Unique Index.

Xác định dữ liệu của cột tham gia khóa Index không lặp lại.

Fill Factor.

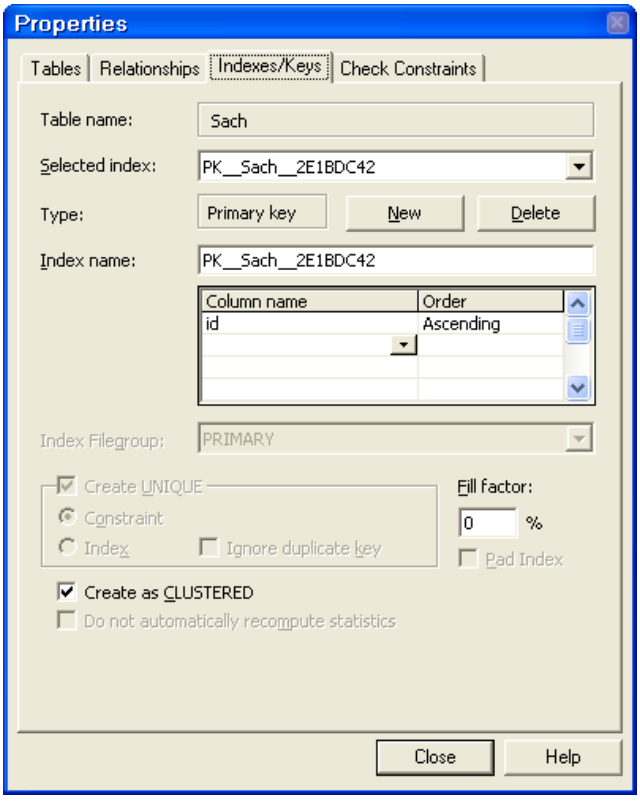
Khi tạo khóa Index, dữ liệu tham giá tạo khóa Index sẽ được phân theo mức của B-Tree, các mức được phần theo page dữ liệu, giá trị Fill factor xác định phần khoảng trống tối đa của page theo tỷ lệ phần trăm. Nhờ khoảng trống này mà tốc độ bố trí cấu trúc Index, tốc độ truy lục thông tin trong cây được cải thiện.

TẠO KHÓA INDEX.

Tạo theo công cụ.

- Chọn chức năng Design table

- Vào bảng Index manager.



- New

- Chọn các cột tham gia tạo khóa Index
- Đặt tham số.

Tạo theo câu lệnh.

- Sử dụng trong câu lệnh Create Table, Alter Table.
- Sử dụng lệnh Create Index.

CREATE [UNIQUE] [CLUSTERED | NONCLUSTERED] INDEX *index_name*

ON { *table* | *view* } **(** *column* [ASC | DESC] [**,**...*n*] **)**

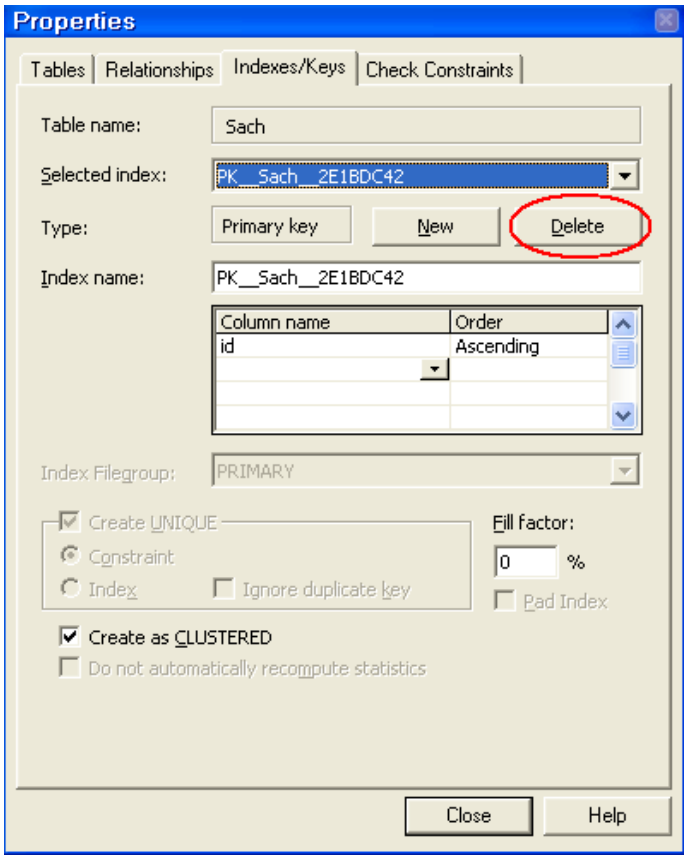
Ví dụ:

```
CREATE INDEX sach_idx  ON sach (id)
```

XÓA INDEX.

Sử dụng công cụ.

- Vào Index amnager
- Chọn khóa Index -> Delete



Sử dụng câu lệnh.

Sử dụng lệnh Drop Index.

Drop Index Sach(sach_idx)

KHUNG NHÌN – VIEW

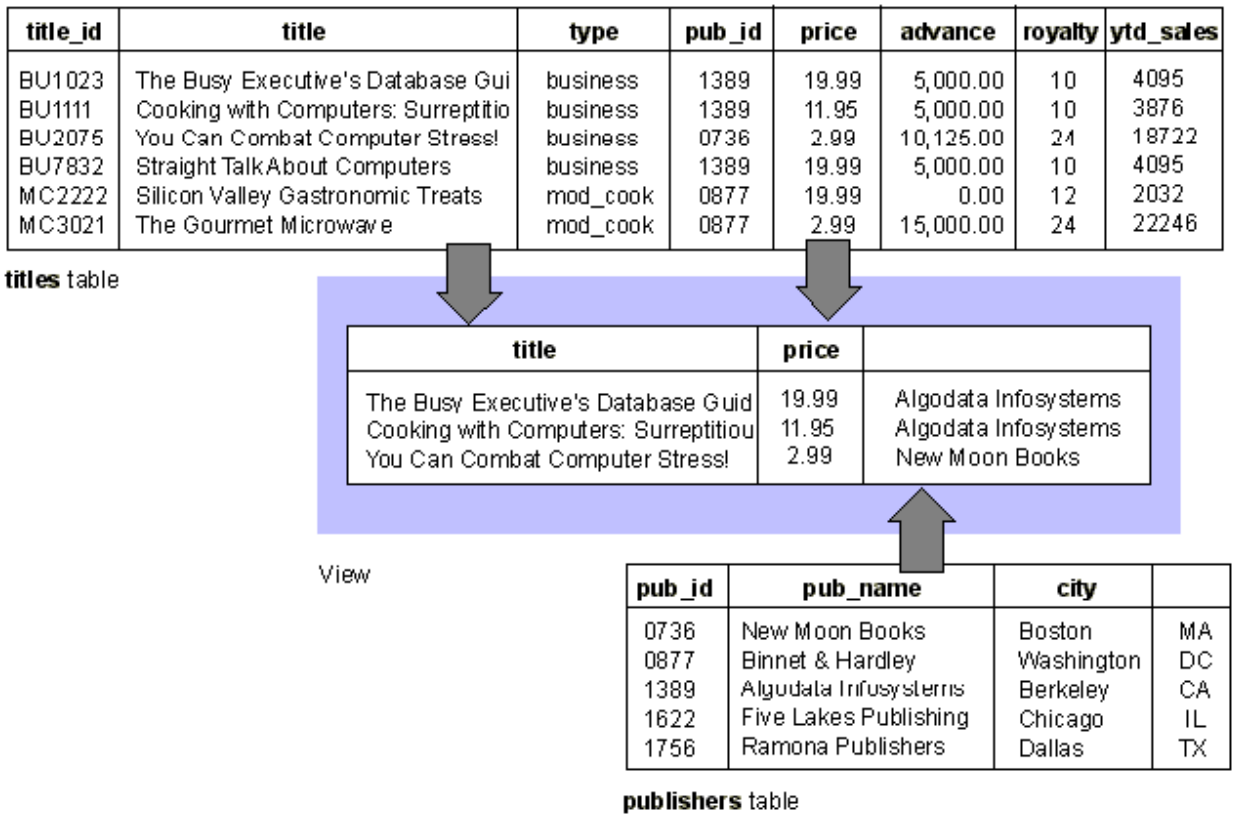
KHÁI NIỆM KHUNG NHÌN.

Khung nhìn (View) là một bảng tạm thời, có cấu trúc như một bảng, khung nhìn không lưu trữ dữ liệu mà nó được tạo ra khi sử dụng, khung nhìn là đối tượng thuộc CSDL.

Khung nhìn được tạo ra từ câu lệnh truy vấn dữ liệu (lệnh Select), truy vấn từ một hoặc nhiều bảng dữ liệu.

Khung nhìn được sử dụng khai thác dữ liệu như một bảng dữ liệu, chia sẻ nhiều người dùng, an toàn trong khai thác, không ảnh hưởng dữ liệu gốc.

Có thể thực hiện truy vấn dữ liệu trên cấu trúc của khung nhìn.

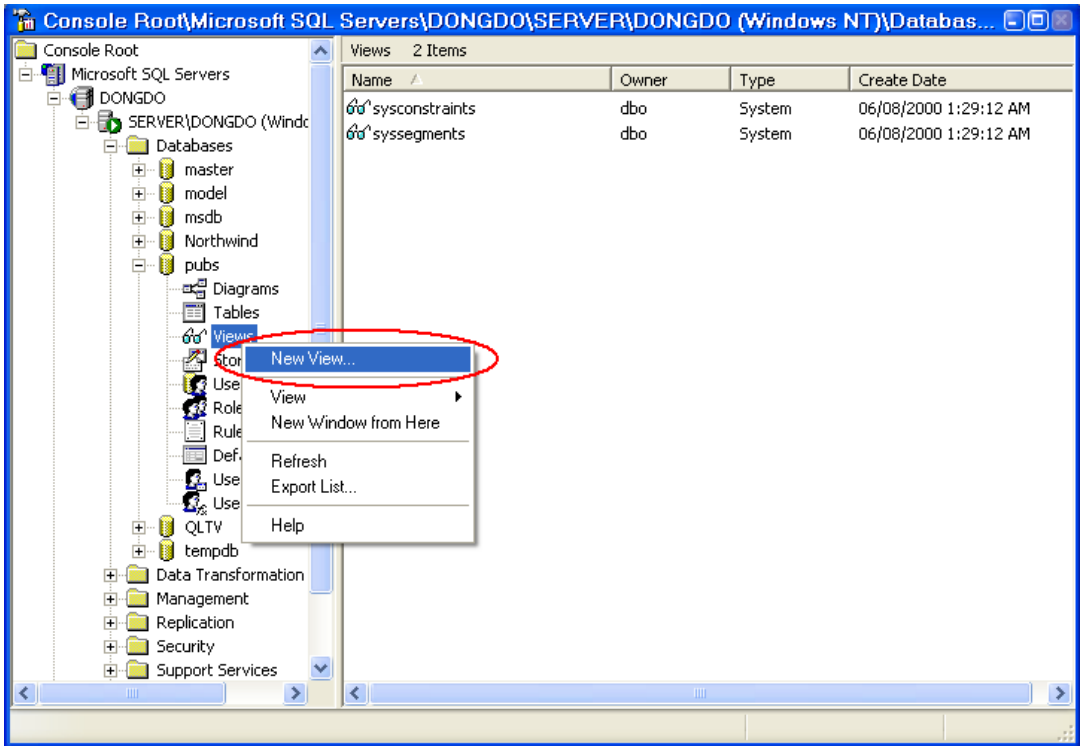


TẠO KHUNG NHÌN.

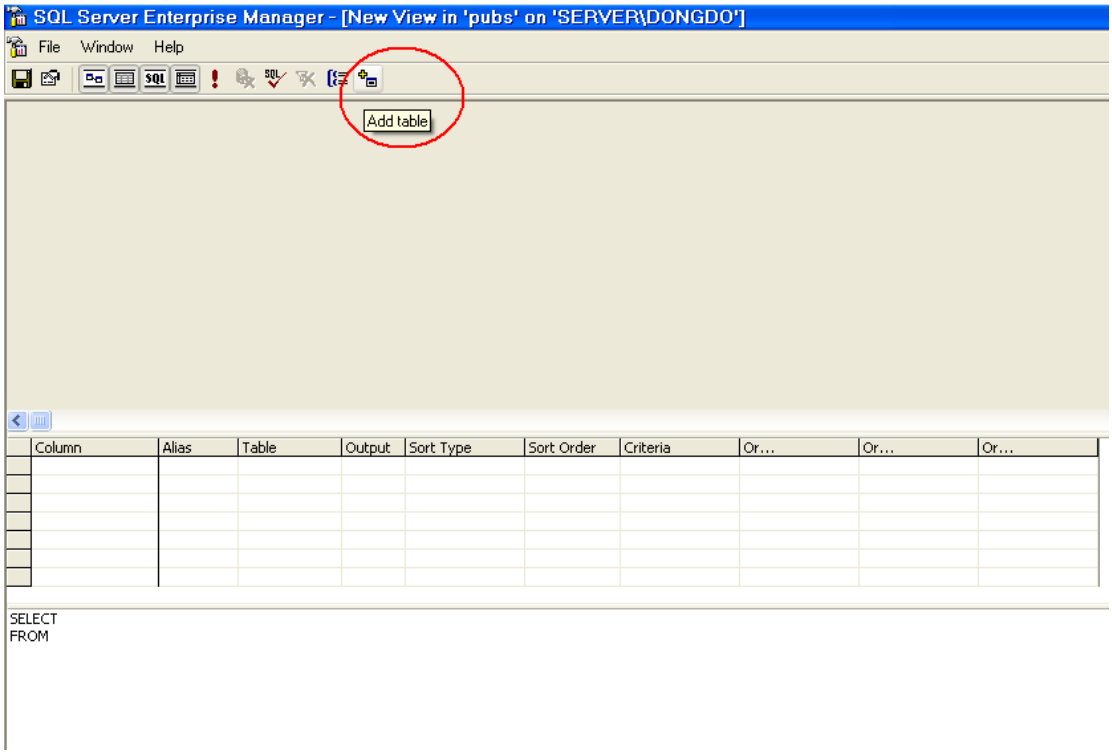
Sử dụng công cụ.

- Chọn chức năng Views của CSDL.

- Nhấn phải chuột.

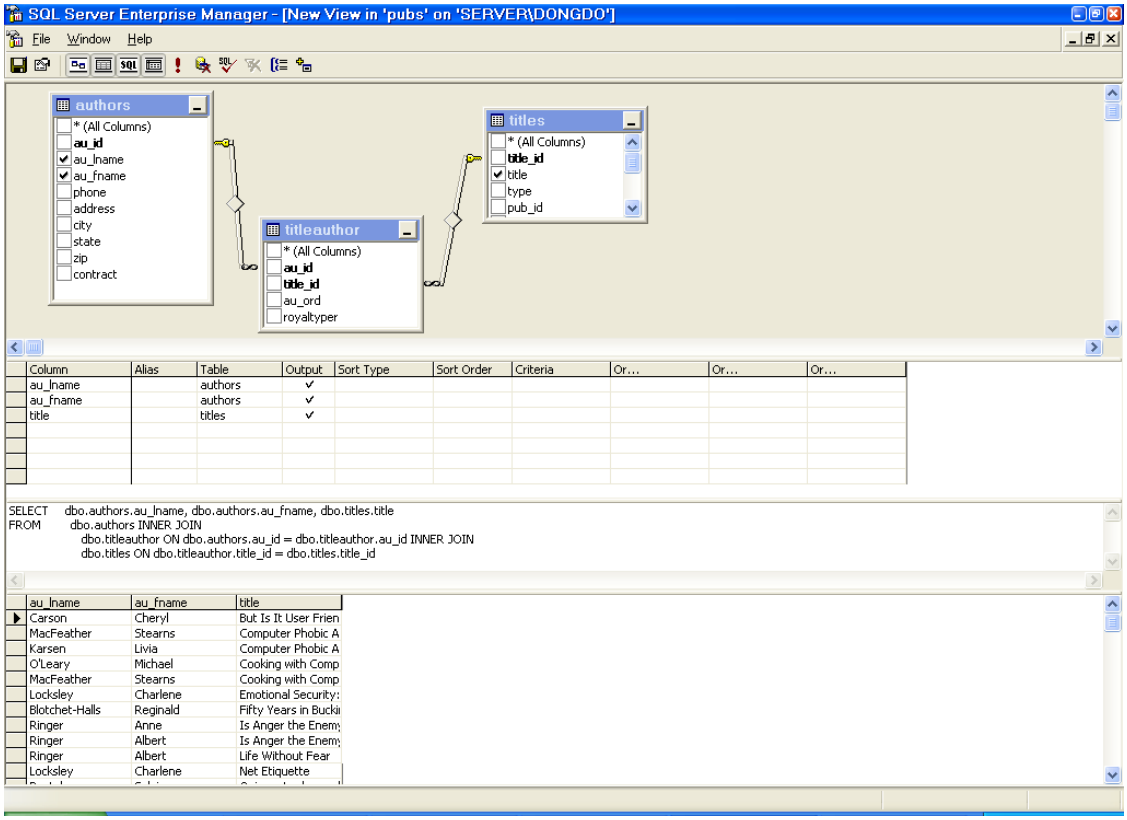


- Chọn New View.



- Thêm các bảng tham gia câu lệnh truy vấn dữ liệu cho View

- Soạn lệnh truy vấn hoặc đánh dấu các cột tham gia tạo View.



- Sửa đổi lệnh Select theo ý muốn.

- Ghi kịch bản -> đặt tên view.

Tạo theo câu lệnh.

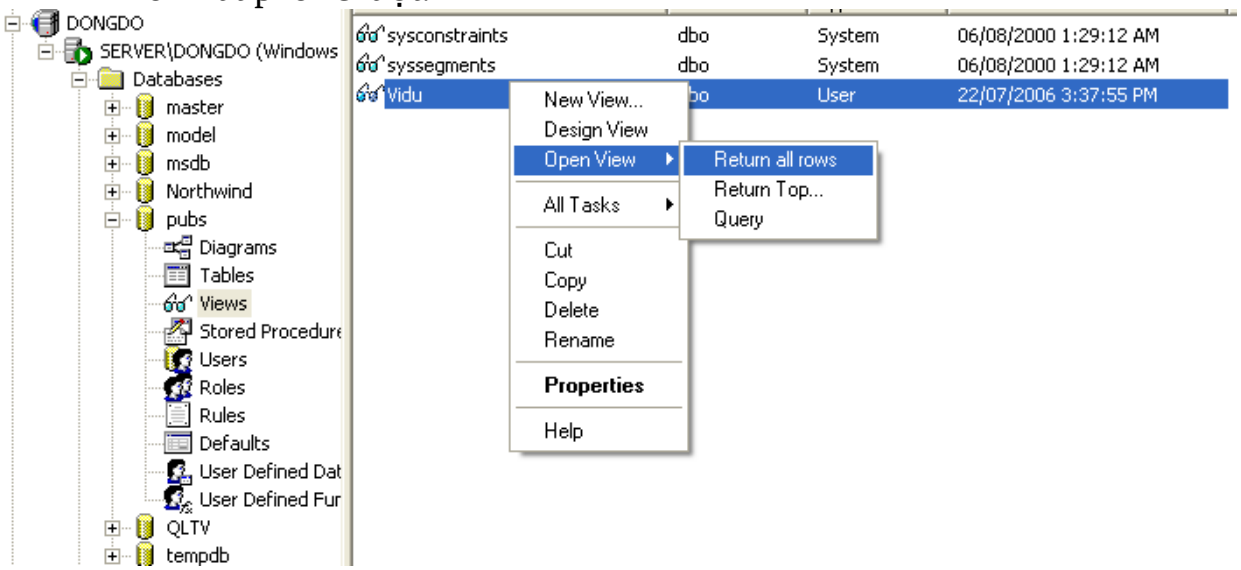
Sử dụng lệnh Create View:

```
CREATE VIEW VIDU as
SELECT     dbo.authors.au_lname, dbo.authors.au_fname, dbo.titles.title
FROM       dbo.authors INNER JOIN
                dbo.titleauthor ON dbo.authors.au_id = dbo.titleauthor.au_id INNER
JOIN
                dbo.titles ON dbo.titleauthor.title_id = dbo.titles.title_id
```

SỬ DỤNG VIEW.

- Chọn View

- Nhấn nút phải chuột.



Thực hiện các chức năng tương tự table.

THỦ TỤC LƯU TRỮ

KHÁI NIỆM THỦ TỤC LƯU TRỮ VÀ HÀM.

Thủ tục lưu trữ có thuật ngữ Stored Procedure, là một đối tượng của CSDL tương tự như khung nhìn, thủ tục lưu trữ có thể tạo ra từ công cụ và câu lệnh. THủ tục được thực hiện như câu lệnh (có thể thực hiện từ SQL Query analyzer, các vị trí gọi câu lệnh T-SQL).

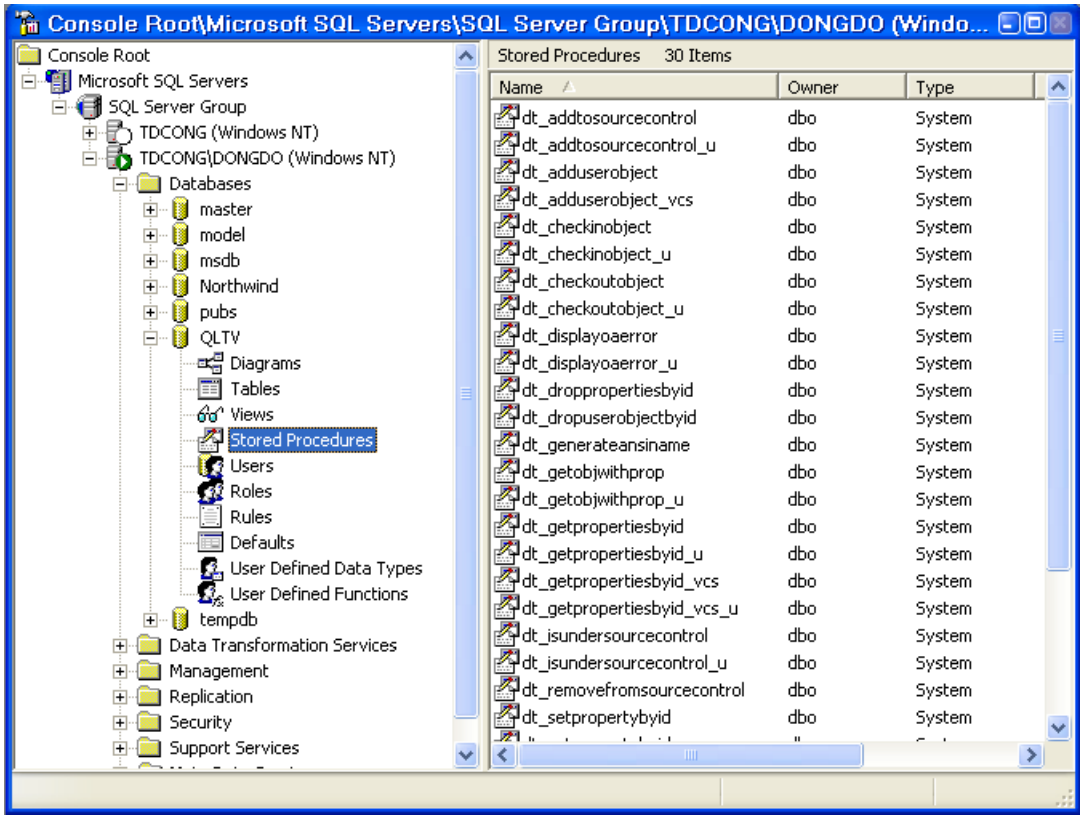
Thủ tục lưu trữ được kết cấu từ một kịch bản câu lệnh T-SQL, thủ tục có những đặc điểm cơ bản sau:

+ Truyền tham số.

+ Gọi thủ tục khác.

+ Trả về các giá trị tham số, chuyển giá trị tham số cho các thủ tục được gọi.

+ Trả về giá trị trạng thái thủ tục là thành công hay không thành công.



Thủ tục lưu trữ có nhiều ưu điểm so với thực hiện câu lệnh T-SQL từ các máy khách:

+ Lập trình theo module: Thủ tục được thiết lập trong từng CSDL một lần, có thể gọi thực hiện nhiều lần trong một ứng dụng, có thể gọi từ nhiều ứng dụng.

+ Thực hiện nhanh hơn: Khi cần thực hiện một lượng lớn câu lệnh T-SQL, thủ tục lưu trữ thực hiện nhanh hơn vì khi máy chủ nhận được nhiều cầu lệnh cùng một lúc đều phải kiểm tra tính hợp lệ quyền của tài khoản từ máy khách và các tham số khác. Khi thủ tục cần gọi nhiều lần trên các máy khách thì thủ tục thực hiện một lần đầu tiên, những lần sau máy khách sẽ chạy thủ tục đã được biên dịch.

+ Làm giảm lưu lượng trên mạng: Thay cho vì máy khách phải gửi nhiều dòng lệnh từ các ứng dụng đến máy chủ, khi sử dụng thủ tục thì nó chỉ cần gửi một lệnh, từ đó dẫn đến lưu lượng thông tin lệnh truyền qua mạng giảm.

+ An ninh bảo mật hơn: Khi không muốn cho một user trực tiếp khai thác một đối tượng hay bảng dữ liệu nào đó, mà cần cho user đó được khai thác thì thủ tục có thể giúp bạn gán quyền khai thác cho người đó. Việc gán quyền khai thác như nói trên sẽ giúp cho vấn đề an ninh bảo mật trong CSDL tốt hơn.

PHÂN LOẠI THỦ TỤC LƯU TRỮ.

Thủ tục lưu trữ được phân thành 5 loại như sau:

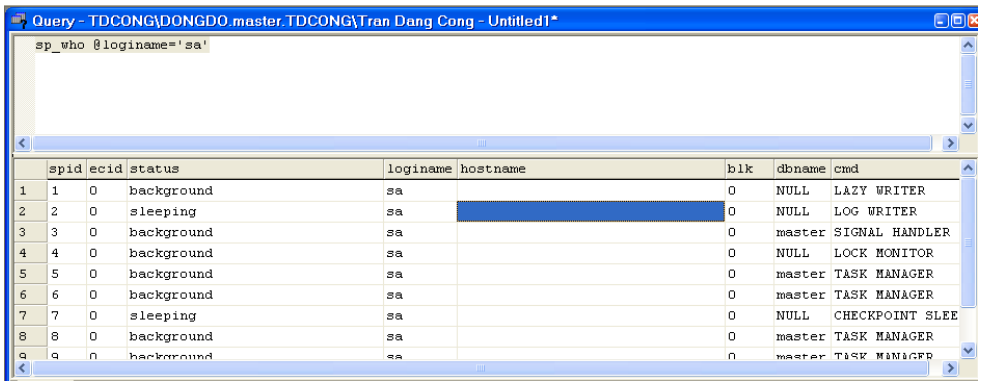
System Stored Procedure.

Là thủ tục được lưu trữ tỏng CSDL Master, thủ tục loại này được bắt đầu bằng chữ **sp_** thủ tục loại này thường được sử dụng trong quản trị CSDL và an ninh bảo mật.

Ví dụ: Muốn biết tất cả các tiến trình đang thực hiện bởi user nào:

sp_who @loginame='sa'

Kết quả:



	spid	ecid	status	loginame	hostname	blk	dbname	cmd
1	1	0	background	sa		0	NULL	LAZY WRITER
2	2	0	sleeping	sa		0	NULL	LOG WRITER
3	3	0	background	sa		0	master	SIGNAL HANDLER
4	4	0	background	sa		0	NULL	LOCK MONITOR
5	5	0	background	sa		0	master	TASK MANAGER
6	6	0	background	sa		0	master	TASK MANAGER
7	7	0	sleeping	sa		0	NULL	CHECKPOINT SLEE
8	8	0	background	sa		0	master	TASK MANAGER
9	9	0	background	sa		0	master	TASK MANAGER

Local Stored Procedure.

Đây là loại thủ tục thường dùng nhất, nằm trong CSDL do người dùng tạo ra, thực hiện một công việc nào đó. Thủ tục loại này thường được tạo bởi DBA (Database Administrator) hoặc người lập trình.

Temporary Stored Procedure.

Có chức năng tương tự như Local Stored Prcedure nhưng thủ tục loại này tự hủy khi kết nối tạo ra nó ngắt hoặc SQL Server ngưng hoạt động và nó được tạo ra trên CSDL TempDB.

Extended Stored Procedure.

Đây là loại thủ tục sử dụng chương trình ngoại vi đã được biên dịch thành DLL. Tên thủ tục được bắt đầu bằng **xp_**. Ví dụ thủ tục xp_sendmail dùng gửi mail, thủ tục xp_cmdshell dùng thực hiện lệnh của DOS (xp_cmdshell 'dir c:\').

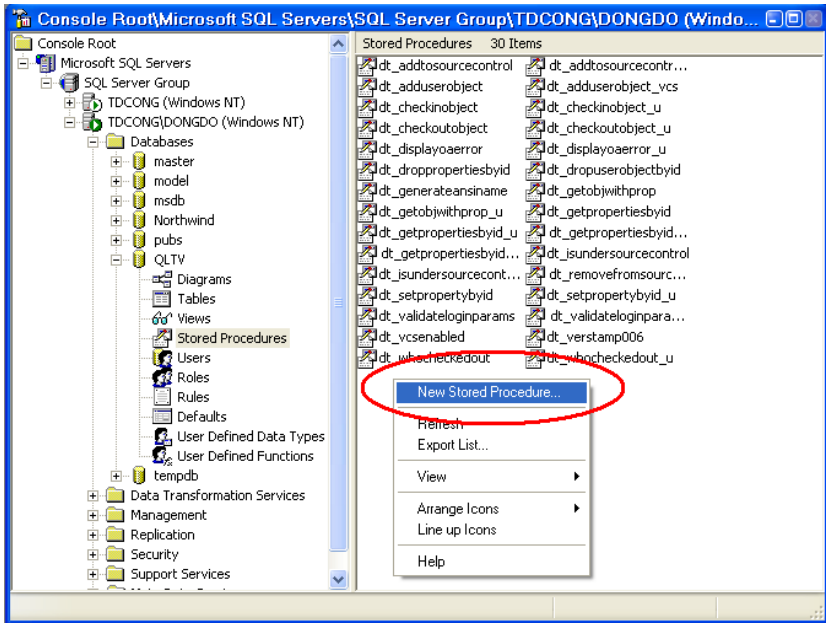
Remote Stored Procedure:

Là loại thủ tục sử dụng thủ tục của một server khác.

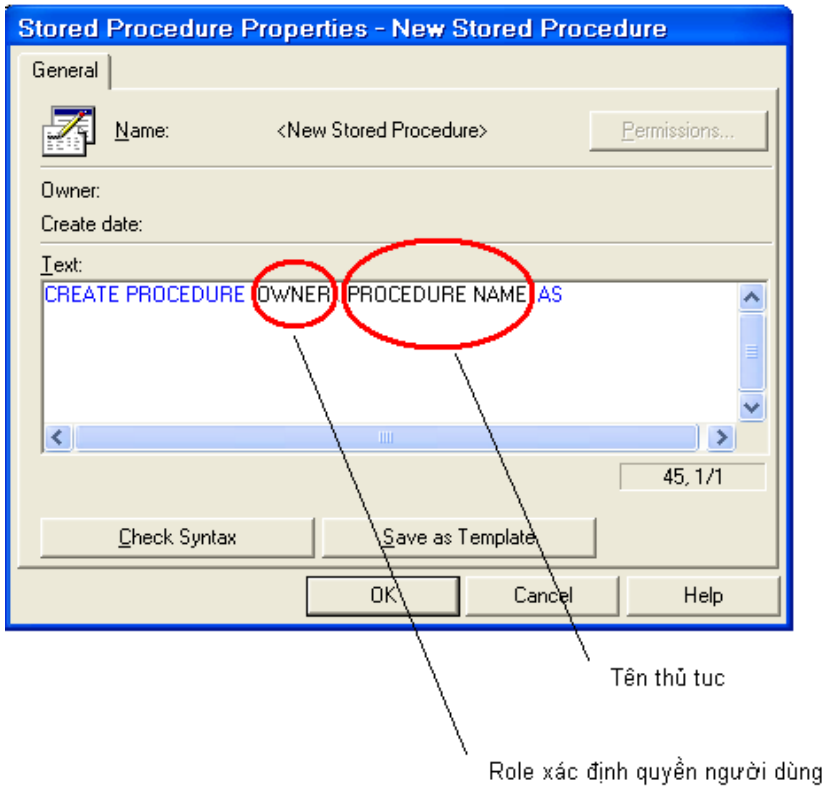
THIẾT LẬP THỦ TỤC LƯU TRỮ.

Sử dụng công cụ.

- Chọn CSDL cần tạo thủ tục trong Enterprise Manager –> Stored Procedures
- Nhấn nút phải chuột -> New Stored Procedure…



- Đặt tên thủ tục, xác định role người khai thác, soạn kịch bản câu lệnh.



Sử dụng câu lệnh.

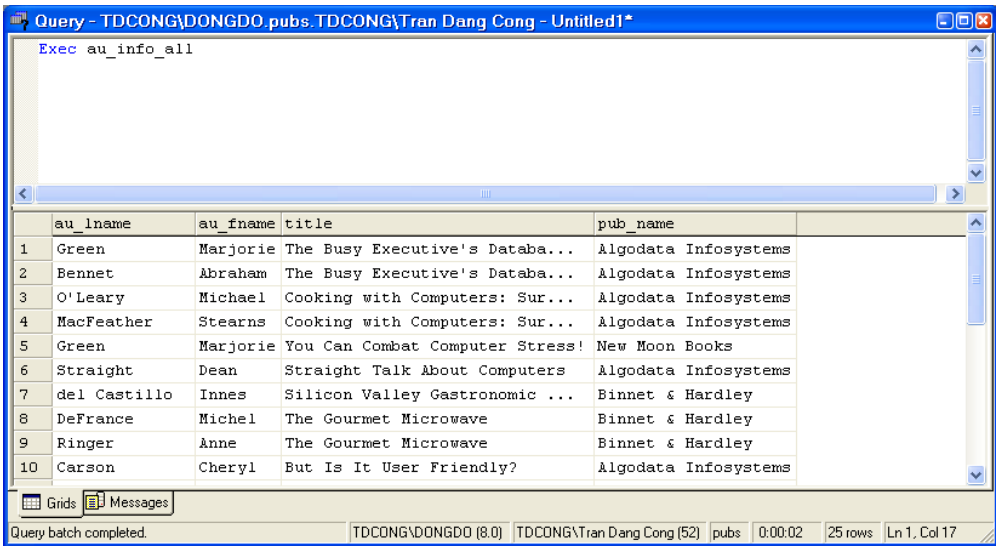
Sử dụng lệnh Create Procedure, để tiện xem xét ta xét theo các ví dụ , các ví dụ dưới đây thực hiện tạo thủ tục và thao tác với CSDL pubs để tiện trong dữ liệu mẫu, để tìm hiểu cú pháp câu lệnh T-SQL bạn xem phần câu lệnh T-SQL trong cùng tài liệu này.

Thủ tục không có tham số.

Thủ tục sau sẽ thực hiện liệt kê tất cả các tác giả, sách và nhà xuất bản mà tác giả viết sách.

```
Use Pubs
CREATE PROCEDURE au_info_all
AS
SELECT au_lname, au_fname, title, pub_name
  FROM authors a INNER JOIN titleauthor ta
    ON a.au_id = ta.au_id INNER JOIN titles t
    ON t.title_id = ta.title_id INNER JOIN publishers p
    ON t.pub_id = p.pub_id
GO
```

Kết quả thực hiện:



Thủ tục có tham số.

Thủ tục sau thực hiện lọc tìm tác giả có tên, họ truyền theo tham số.

```
USE pubs
GO
CREATE PROCEDURE au_info
   @lastname varchar(40),
   @firstname varchar(20)
AS
SELECT au_lname, au_fname, title, pub_name
   FROM authors a INNER JOIN titleauthor ta
      ON a.au_id = ta.au_id INNER JOIN titles t
            ON  t.title_id  =  ta.title_id  INNER  JOIN
publishers p
      ON t.pub_id = p.pub_id
   WHERE  au_fname = @firstname
      AND au_lname = @lastname
GO
```

Cách truyền tham số:

+ Gán giá trị theo thứ tự:

```
    EXECUTE au_info 'Dull', 'Ann'
+ Gán giá trị theo tên biến
   EXECUTE au_info @lastname = 'Dull', @firstname =
   'Ann'
+ Gán giá trị theo tên biến, không theo thứ tự
EXECUTE au_info @firstname = 'Ann', @lastname = 'Dull'
```

Thủ tục có tham số tùy lựa theo giá trị đưa vào.

Ví dụ này sẽ đề cập đến việc truyền tham số theo mẫu, giá trị tham số được ngầm định khi tạo thủ tục và thủ tục khi thực hiện sẽ kiểm tra giá trị tham số nhập vào.

```
USE pubs
GO
CREATE PROCEDURE au_info2
   @lastname varchar(30) = 'D%',
   @firstname varchar(18) = '%'
AS
SELECT au_lname, au_fname, title, pub_name
FROM authors a INNER JOIN titleauthor ta
   ON a.au_id = ta.au_id INNER JOIN titles t
   ON t.title_id = ta.title_id INNER JOIN publishers p
   ON t.pub_id = p.pub_id
WHERE au_fname LIKE @firstname
   AND au_lname LIKE @lastname
GO
```

Tham số % xác định giá trị tùy ý nhập vào tham số, tham số D% xác định giá trị đầu tiên của chuỗi phải bằng chữ D. Khi ngầm định các giá trị như trên tham có không được truyền giá trị sẽ tự nhận giá trị ngầm định.

Cách truyền tham số như sau:

```
+ Không truyền tham số:
    EXECUTE au_info2
+ Chỉ truyền tham số đầu, tham số sau sẽ nhận giá trị
ngầm định.
```

```
      EXECUTE au_info2 'Wh%'
+ Chỉ truyền một tham số, tham số xòn lại sẽ nhận giá
trị ngầm định.
    EXECUTE au_info2 @firstname = 'A%'
+ Tham số thứ nhất xác định giá trị một ký tự thuộc vị
trí có [CK] chỉ nhận ký tự ‘C’ hoặc ‘K’, [OE] chỉ nhận
giá trị ‘O’ hoặc ‘E’.
    EXECUTE au_info2 '[CK]ars[OE]n'
+ Xác định rõ giá trị tham số
    EXECUTE au_info2 'Hunter', 'Sheryl'
+ Xác định kiểu giá trị tham số.
    EXECUTE au_info2 'H%', 'S%'
```

Thủ tục sử dụng tham só láy giá trị ra (tham trị).

Ví dụ sau sẽ mô tả kỹ thuật sử dụng tham trị, như trong các ví dụ trước ta sử dụng tham số để truyền giá trị vào tên tham số bắt đầu bằng 1 chữ @, tham số được bắt đầu bằng 2 chữ @@. Sẽ được sử dụng trên nhiều dòng lệnh, sử dụng cùng từ khóa OUTPUT xác định là tham trị để lấy giá trị ra.

Ví dụ sau thực hiện truyền tham số vào và lấy giá trị ra:

```
USE pubs
GO
CREATE PROCEDURE titles_sum @@TITLE varchar(40) = '%',
@@SUM money OUTPUT
AS
SELECT 'Title Name' = title
FROM titles
WHERE title LIKE @@TITLE
SELECT @@SUM = SUM(price)
FROM titles
WHERE title LIKE @@TITLE
GO
```

Tham số bắt đầu bằng 2 ký tự @@ xác định được sử dụng cho nhiều câu lệnh, sử dụng cùng từ khóa OUTPUT xác định là biến tham trị.. Ví dụ trên sử dụng biến @@Title xác định điều kiện đưa ra tên sách đây là loại biến truyền vào sử dụng cho hai câu lệnh Select, biến @@Sum xác định là biến tham trị dùng lấy giá trị ra.

Cách sử dụng tham số như sau:

```
DECLARE @@TOTALCOST money
EXECUTE titles_sum 'The%', @@TOTALCOST OUTPUT
IF @@TOTALCOST < 200
BEGIN
   PRINT ' '
   PRINT 'All of these titles can be purchased for less
than $200.'
END
ELSE
   SELECT 'The total cost of these titles is $'
         + RTRIM(CAST(@@TOTALCOST AS varchar(20)))
```

Ví dụ trên sử dụng biến @@TOTALCOST vào vị trí biến @@SUM trong thủ tục. Kết quả thực hiện như sau:

```
Title Name
--------------------------------------------------
The Busy Executive's Database Guide
The Gourmet Microwave
The Psychology of Computer Cooking

(3 row(s) affected)

Warning, null value eliminated from aggregate.

All of these titles can be purchased for less than
$200.
```

Thủ tục sử dụng biến OUTPUT kiểu con trỏ (Cursor).

Ví dụ sau tạo thủ tục có biến kiểu Cursor, biến này sẽ quản lý một bảng dữ liệu được truy vấn bằng câu lệnh Select.

```
CREATE PROCEDURE titles_cursor @titles_cursor CURSOR
VARYING OUTPUT
AS
SET @titles_cursor = CURSOR
FORWARD_ONLY STATIC FOR
SELECT *
```

```
FROM titles

OPEN @titles_cursor
GO
```

Con trỏ được đưa vào biến kiểu Cursor có tên @Titles_cursor, hướng dịch chuyển Forward (tiến) và Static. Sử dụng biến như ví dụ sau:

```
USE pubs
GO
DECLARE @MyCursor CURSOR
EXEC titles_cursor @titles_cursor = @MyCursor OUTPUT
WHILE (@@FETCH_STATUS = 0)
BEGIN
   FETCH NEXT FROM @MyCursor
END
CLOSE @MyCursor
DEALLOCATE @MyCursor
GO
```

Biến con trỏ được đưa vào viến @MyCursor, khi mở con trỏ vị trí bản ghi đầu tiên của bảng được xác định. Trong ví dụ trên sử dụng vòng lặp duyệt từng bản ghi, việc xử lý dữ liệu thực hiện trong vòng lặp.

Thủ tục đặt thuộc tính ẩn kịch bản câu lệnh.

Ví dụ sau sẽ đặt thuộc tính WITH ENCRYPTION ẩn văn bản trong thủ tục với người sử dụng.

```
CREATE PROCEDURE encrypt_this
WITH ENCRYPTION
AS
SELECT *
FROM authors
GO
```

Khi sử dụng thủ tục hệ thống sp_helptext để xem nội dung thủ tục:

```
EXEC sp_helptext encrypt_this
```

Kết quả như sau:

```
The object's comments have been encrypted.
```

SỬA, XÓA THỦ TỤC

Sử dụng công cụ.

- Chọn thủ tục cần sửa, xóa -> thực hiện sửa nội dung hoặc chức năng xóa.

Sử dụng câu lệnh.

- Sửa sử dụng lệnh Alter Procedure
- Xóa sử dụng lệnh Drop Procedure

Bạn đọc có thể tự tìm hiểu về User Defined Function tong Book Online, là đối tượng gọi là hàm thuộc CSDL, có chức năng và cách thức hoạt động gần giống thủ tục.

TRIGGER

KHÁI NIỆM TRIGGER.

Trigger là một thủ tục đặc biệt mà việc thực thi của nó tự động khi có sự kiện xảy ra, các sự kiện gọi thủ tục đặc biệt này được định nghĩa trong câu lệnh, thông thương được thực hiện với các sự kiện liên quan đến Insert, Update, Delete dữ liệu.

Trigger được sử dụng trong việc bảo đảm toàn vẹn dữ liệu theo quy tắc xác định, được quản lý theo bảng dữ liệu hoặc khung nhìn.

NHỮNG TRƯỜNG HỢP SỬ DỤNG TRIGGER.

- Sử dụng Trigger khi các biện pháp toàn vẹn dữ liệu như Constraint, rule, … không bảo đảm. Khác với các công cụ bao đảm toàn vẹn dữ liệu đã nêu, các công cụ này sẽ thực hiện kiểm tra tính toán vẹn trước khi đưa dữ liệu vào CSDL (còn gọi là Declarative Data Integrity), còn Trigger thực hiện kiểm tra tính toàn vẹn khi công việc đã thực hiện rồi (còn gọi là Procedural Data Integrity).

- Khi CSDL chưa được chuẩn hóa (Normalization) thì có thể xảy ra dữ liệu thừa, chứa ở nhiều vị trí trong CSDL thì yêu cầu đặt ra là dữ liệu cần cập nhật thống nhất trong mọi nơi. Trong trường hợp này ta phải sử dụng Trigger.

- Khi thay đổi day chuyền dữ liệu giữa các bảng với nhau (khi dữ liệu bảng này thay đổi thì dữ liệu trong bảng khác cũng được thay đổi theo).

ĐẶC ĐIỂM CỦA TRIGGER.

- Một trigger có thể thực hiện nhiều công việc (theo kịch bản), có thể nhiều sự kiện kích hoạt thực thi trigger, có thể tách rời các sự kiện trong một trigger.

- Trigger không được tạo trên bảng temprate hay system.

- Trigger chỉ thực thi tự động thông qua các sự kiện mà không thực hiện bằng tay.

- Trigger sử dụng được với khung nhìn.

- Khi trigger thực thi theo các sự kiện Insert hoặc Delete thì dữ liệu khi thay đổi sẽ được chuyển sang các bảng Inserted Table, Deleted Tabla, là 2 bảng tạm thời chỉ chứa trong bộ nhớ, các bảng này chỉ được sử dụng với các lệnh trong trigger. Các bảng này thường được sử dụng để khôi phục lại phần dữ liệu đã thay đổi (roll back).

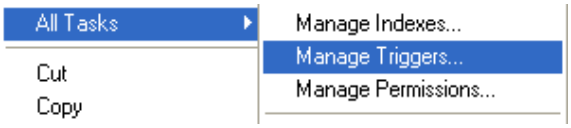
- Trigger chia thành 2 loại Instead of và After: Instead of là loại trigger mà hoạt động của sự kiện gọi nó sẽ bỏ qua và thay vào nó là các lệnh thực hiện trong trigger. After (tương đương với từ khóa For) đây là loại ngầm định, khác với loại Instead of thì loại trigger này sẽ thực hiện các lệnh trong nó sau khi đã thực hiện xong sự kiện gọi nó.

TẠO TRIGGER.

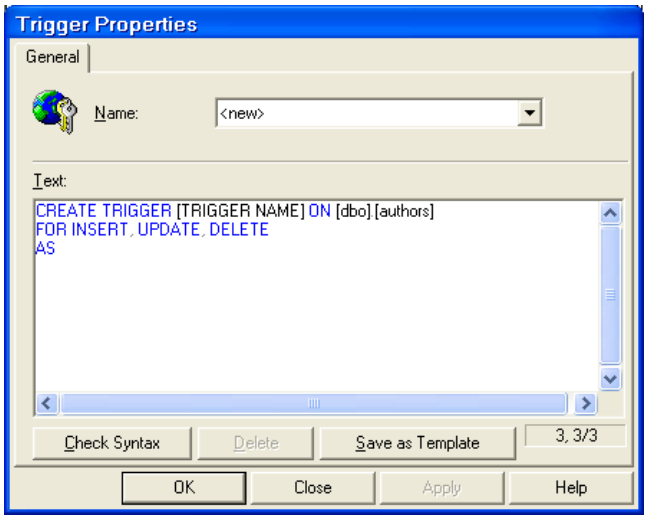
Tạo trigger được thực hiện thông công cụ và câu lệnh:

Tạo trigger bằng công cụ.

- Chọn bảng dữ liệu hoặc khung nhìn.
- Nhấn nút phải chuột.
- Chọn All tasks -> Manage Triggers...



- Soạn kịch bản tạo trigger.



(Cú pháp cụ thể hơn bạn xem trong phần tiếp theo)

Tạo trigger bằng câu lệnh.

Sử dụng lệnh Create Trigger, cú pháp chung như sau:

```
CREATE TRIGGER trigger_name
ON { table | view }
[ WITH ENCRYPTION ]
{
   { { FOR | AFTER | INSTEAD OF } { [ INSERT ] [ , ] [ UPDATE ] }
      [ WITH APPEND ]
      [ NOT FOR REPLICATION ]
      AS
      [ { IF UPDATE ( column )
         [ { AND | OR } UPDATE ( column ) ]
            [ ...n ]
      | IF ( COLUMNS_UPDATED ( ) { bitwise_operator } updated_bitmask )
            { comparison_operator } column_bitmask [ ...n ]
      } ]
      sql_statement [ ...n ]
   }
}
```

Các tham số cơ bản:

+ trigger_name: Tên trigger.

+ table/view: Tên bảng hoặc khung nhìn.

+ For/After/Instead Of: Loại trigger.

+ { [DELETE] [,] [INSERT] [,] [UPDATE] }: Sự kiện khi tự động gọi thực thi trigger.

+ *sql_statement* [...*n*]: Kịch bản các câu lệnh xử lý của trigger.

Các câu lệnh sau không được thực thi trong kịch bản các câu lệnh xử lý của trigger:

ALTER DATABASE	CREATE DATABASE	DISK INIT
DISK RESIZE	DROP DATABASE	LOAD DATABASE
LOAD LOG	RECONFIGURE	RESTORE DATABASE

RESTORE LOG

Để cụ thể hơn ta xét một số ví dụ sau:

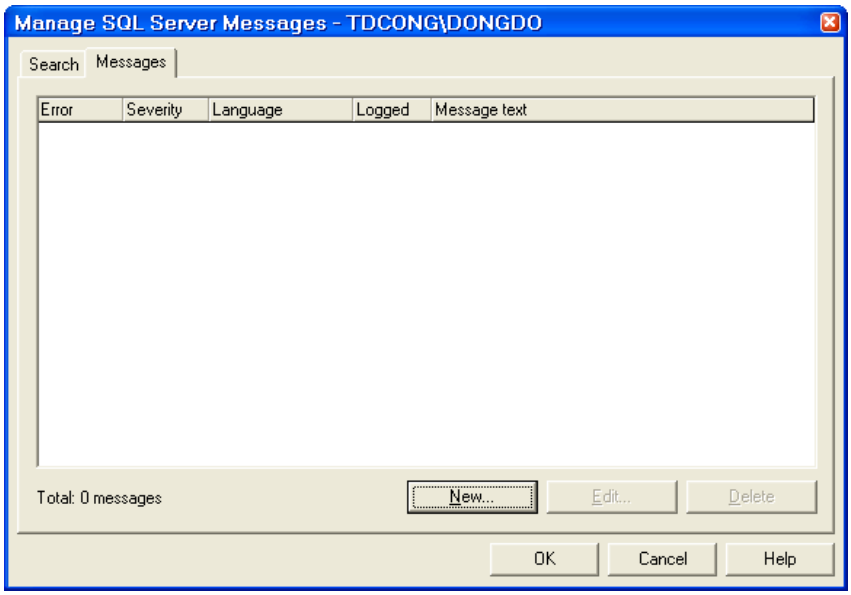
Ví dụ tạo một trigger thông báo.

```
CREATE TRIGGER reminder
ON titles
FOR INSERT, UPDATE
AS RAISERROR (50001, 16, 10)
GO
```

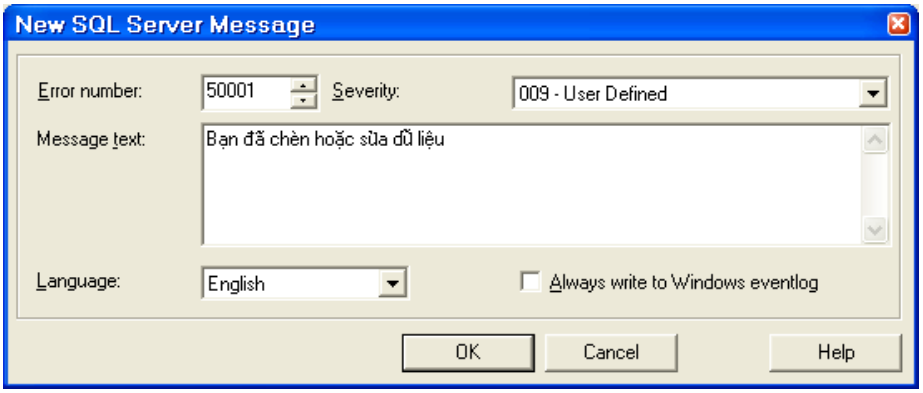
Ví dụ trên tạo một thông báo cho các client khi thực hiện thêm hoặc sửa dữ liệu trên bảng Titles, mã thông báo là 50001, là mã thông báo do người dùng định nghĩa.

Để tạo thông báo bạn thao tác như sau:

- Vào menu Tools -> Manage SQL Server Messages…



- Chọn bảng Messages -> New…



- Đặt mã, soạn nội dung, kiểu thông báo (Serverity), mã thông báo sẽ được sử dụng trong các ứng dụng hoặc câu lệnh yêu cầu.

Ví dụ tạo trigger tự động gửi Email khi được thực thi.

```
CREATE TRIGGER reminder
ON titles
FOR INSERT, UPDATE, DELETE
AS
   EXEC master..xp_sendmail 'MaryM',
      'Don''t forget to print a report for the
distributors.'
GO
```

Ví dụ tạo trigger kiểm soạn khoảng giá trị giữa 2 bảng.

Ví dụ sau sẽ tạo trigger thực hiện kiểm soát phạm vị mức lương của một nhân viên vừa chèn vào có thuộc giá trị định mức lương trong bảng mức lương hay không.

```
CREATE TRIGGER employee_insupd
ON employee
FOR INSERT, UPDATE
AS
DECLARE @min_lvl tinyint,
   @max_lvl tinyint,
   @emp_lvl tinyint,
   @job_id smallint
SELECT @min_lvl = min_lvl,
   @max_lvl = max_lvl,
   @emp_lvl = i.job_lvl,
   @job_id = i.job_id
FROM employee e INNER JOIN inserted i ON e.emp_id =
i.emp_id
   JOIN jobs j ON j.job_id = i.job_id
IF (@job_id = 1) and (@emp_lvl <> 10)
BEGIN
   RAISERROR ('Job id 1 expects the default level of
10.', 16, 1)
   ROLLBACK TRANSACTION
```

```
END
ELSE
IF NOT (@emp_lvl BETWEEN @min_lvl AND @max_lvl)
BEGIN
   RAISERROR ('The level for job_id:%d should be
between %d and %d.',
      16, 1, @job_id, @min_lvl, @max_lvl)
   ROLLBACK TRANSACTION
END
```

SỬA, XÓA TRIGGER.

Sử dụng công cụ.

- Chọn trigger trong mục Manage Triggers…
- Thực hiện sửa nội dung hoặc xóa.

Sửa, xóa theo câu lệnh.

- Sử dụng lệnh Alter trigger để sửa.
- Sử dụng lệnh Drop Trigger để xóa.

(Bạn có thể tìm hiểu chi tiết hơn trong Book Online)

XUẤT – NHẬP DỮ LIỆU

Trong chương này bạn đọc sẽ tìm hiểu kỹ thuật trao đổi dữ liệu với các mỗi trường ngoài Server của SQL, có thể với một CSDL khác, một Server SQL khác, một hệ quản trị CSDL khác hoặc nối ghép tập tin CSDL.

SERVER LIÊN KẾT – LINKED SERVER.

Tương tự như các hệ quản trị CSDL lớn khác (Access, Oracle), hệ thống cung cấp công cụ liên kết với hệ quản trị CSDL khác. Khi liên kết đã được thiết lập, với quyền hạn của user liên kết bạn có thể thực hiện khai thác dữ liệu liên kết trên SQL Server.

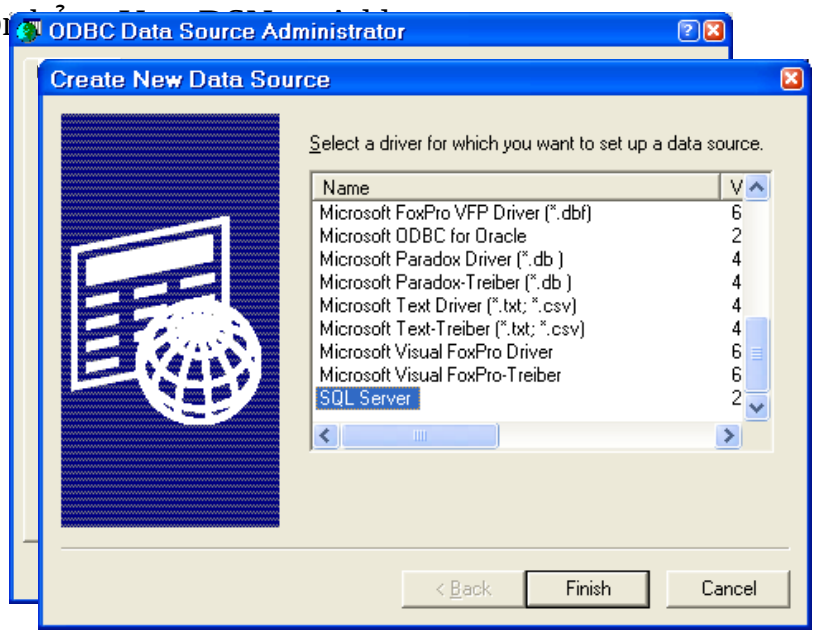
Tạo ODBC.

ODBC viết tắt của cụm từ Open DataBase Connectivity, là công cụ được Windows cung cấp với mục đích làm môi trường trao đổi dữ liệu giữa những CSDL, giữa nhiều hệ quản trị CSDL với ứng dụng.

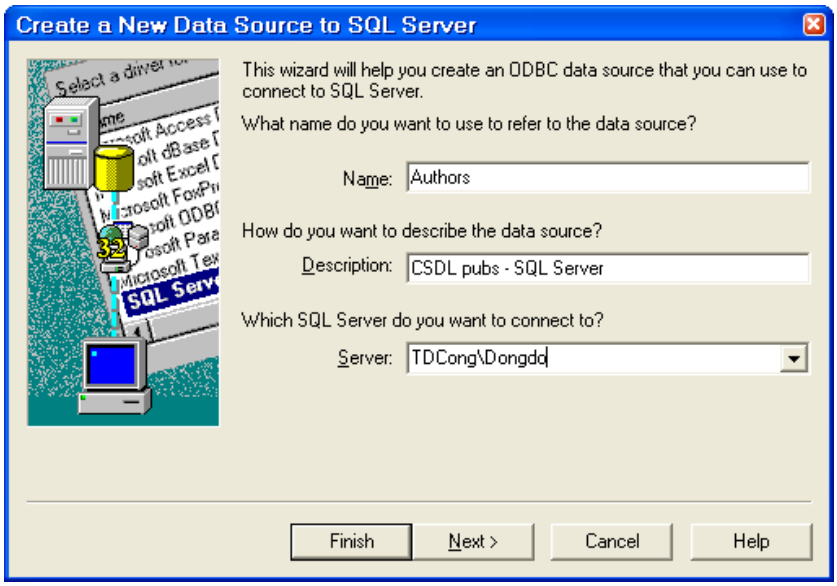
ODBC tạo những chuẩn chung nhất kết nối đến CSDL và ứng dụng. Khi thực hiện khai thác dữ liệu thông qua ODBC, ứng dụng liên kết theo tên ODBC, quyền hạn khai thác thực hiện khi tạo kết nối tự ODBC đến CSDL.

Cách tạo ODBC:

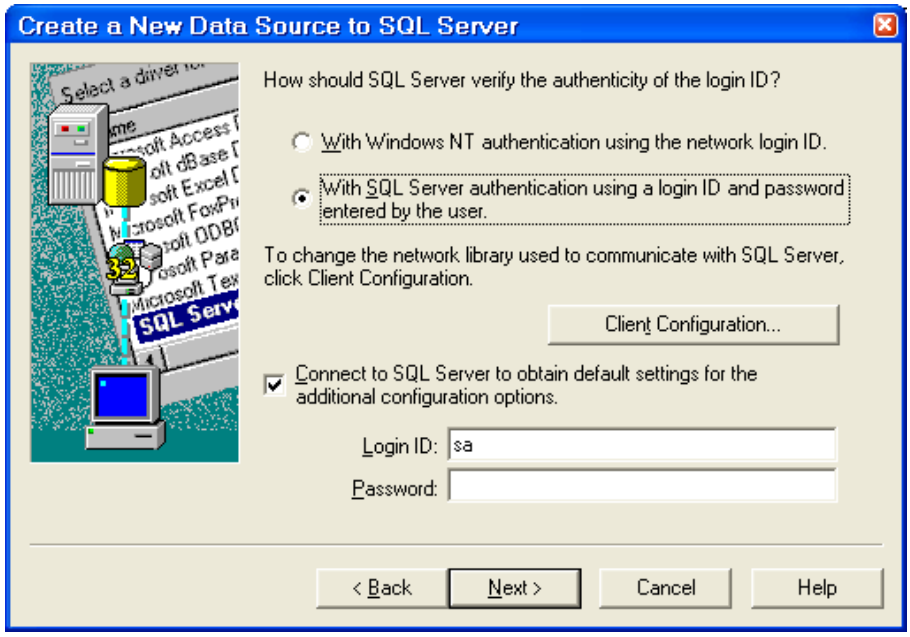
- Chọn ODBC trong Control panel.
- Chọn



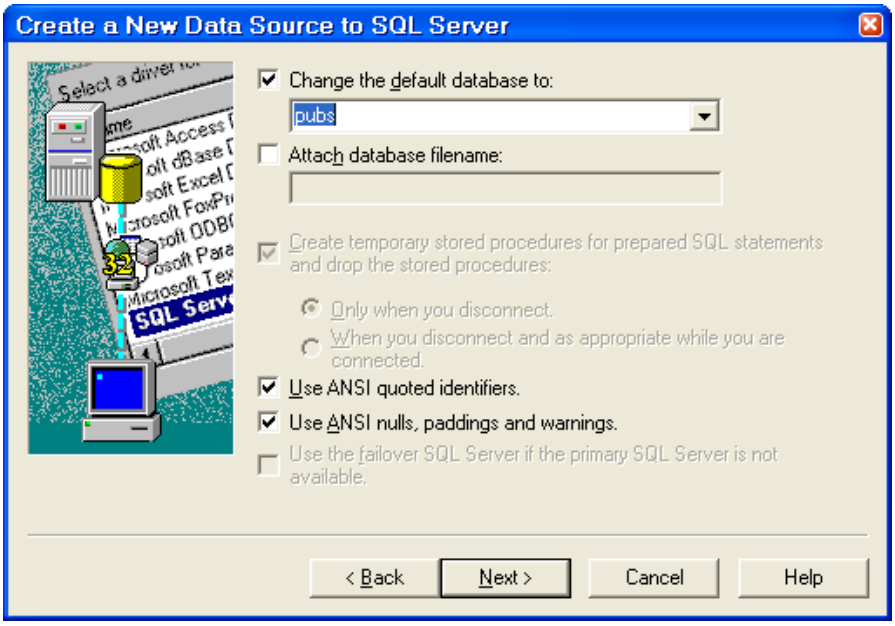
- Chọn Driver của hệ quản trị CSDL của CSDL cần liên tạo ODBC.
- Chọn Finish.



- Nhập tên ODBC (tên này sẽ sử dụng cho ứng dụng khác, nên nhập theo chuẩn chung để dễ sử dụng), các tham số khác (ở đây trên hình sử dụng Driver của SQL Server nên bạn phải chọn Server).
- Next.

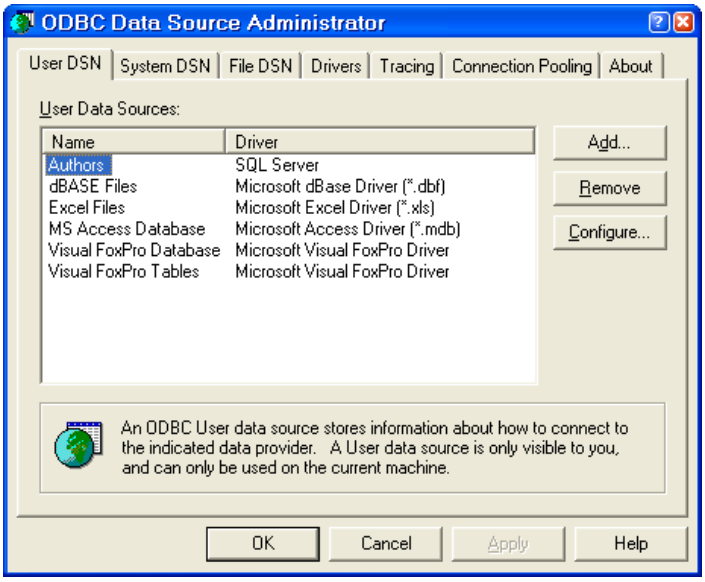


- Nhập Login ID, mật khẩu -> Next.



- Đánh dấu Change the default database to -> Chọn CSDL -> Next -> Finish

Sau khi tạo xong trong danh sách xuất hiện ODBC bạn vừa tạo, tư danh sách bạn có thể sửa đổi, xóa ODBC khi cần thiết.



Tạo liên kết từ Access.

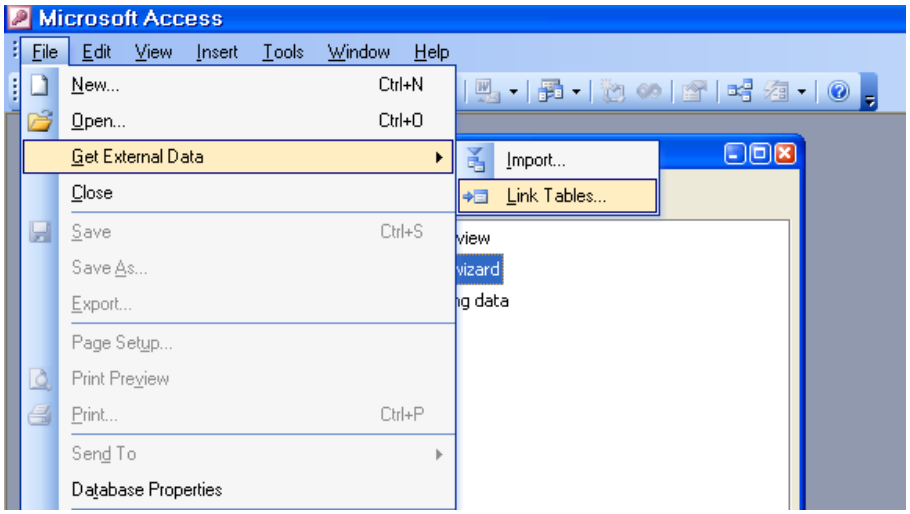
Từ hệ quản trị CSDL Access bạn có thể tạo liên kết đến các hệ quản trị CSDL khác (Acess, Dbase,…), hoặc thông qua ODBC. Trong ví dụ minh họa sử dụng liên kết từ Access với ODBC (đối với SQL Server hoặc Oracle, My SQL thì Access phải liên kết thông qua ODBC vì các hệ quản trị CSDL này không thực hiện khai thác dữ liệu qua tập tin chỉ khai thác thông qua tên CSDL, mà Access chỉ thực hiện theo phương thức mở tập tin).

Các bước thực hiện như sau:

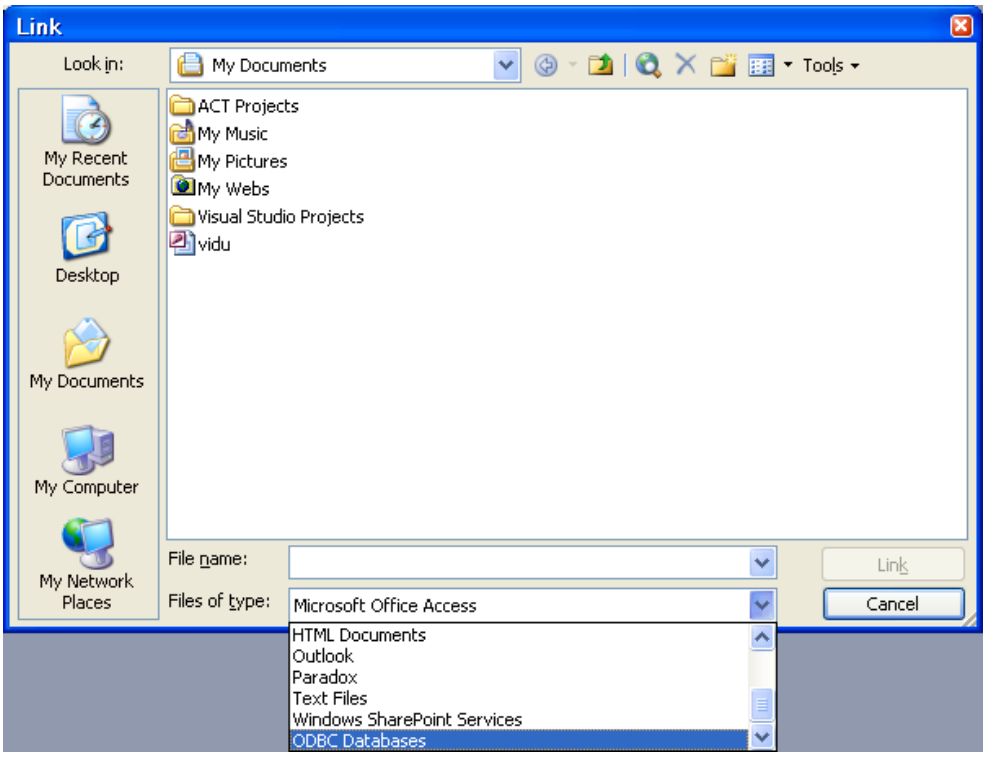
- Mở hệ quản trị CSDL Access.

- Mở hoặc tạo CSDL mới từ Access

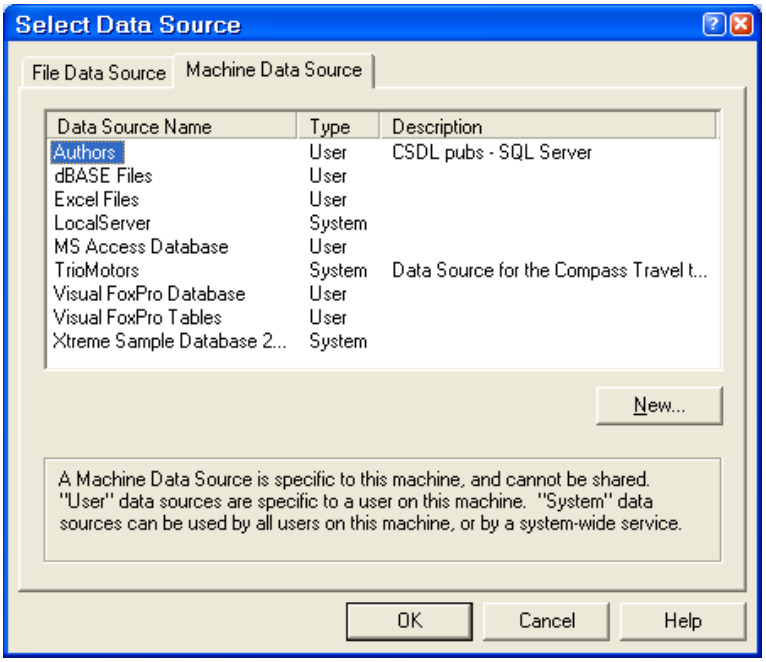
- Chọn File -> Get External Data -> Link Tables.



- Chọn ODBC Databases.



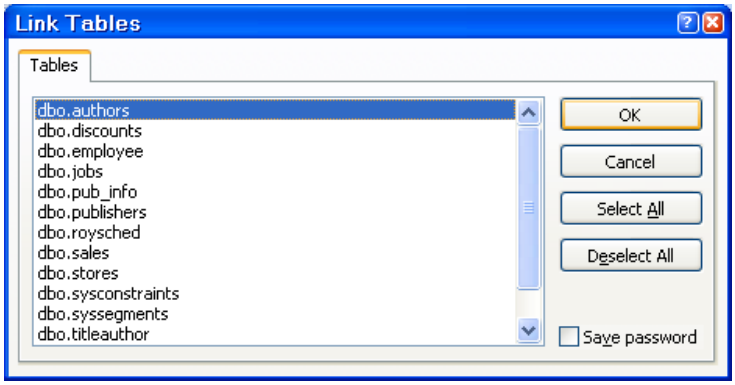
- Chọn ODBC cần liên kết (Authors).



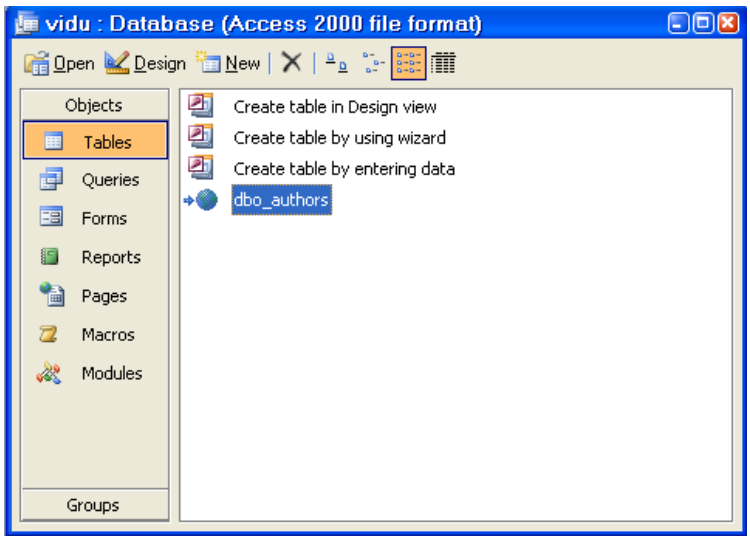
- Ok.

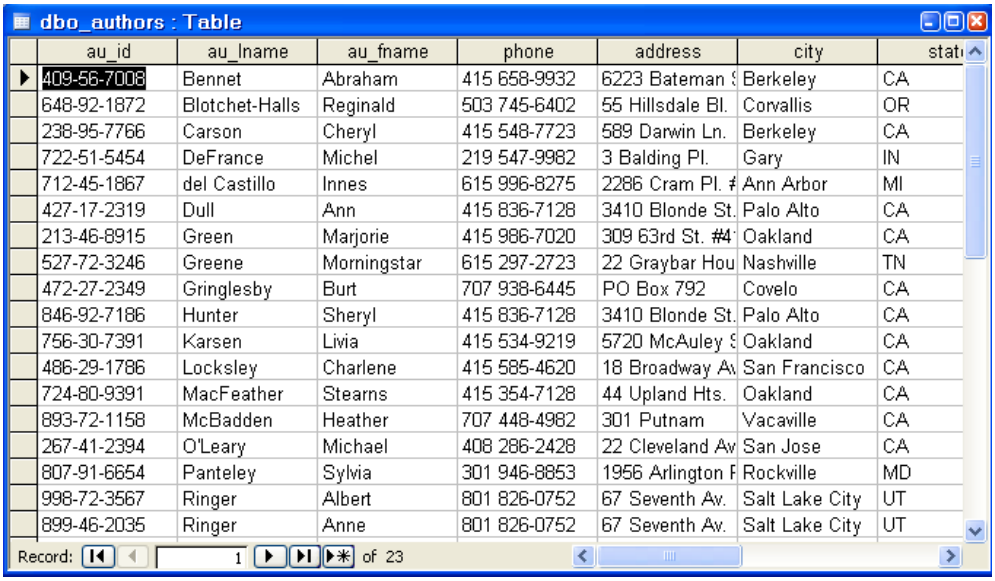
- Nhập Login ID và mật khẩu.

- Chọn bảng hoặc khung nhìn cần liên kết trong danh sách.



- Nhấn Ok, danh sách các bảng trong Access được khai thác tương tự như các bảng khác.



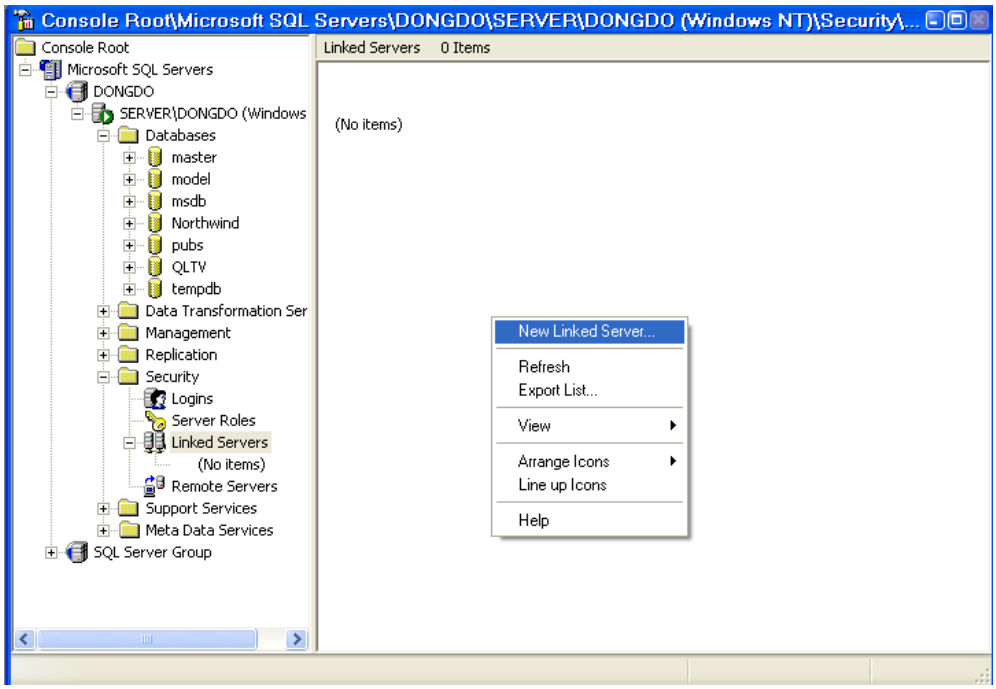


Tạo Server liên kết – Linked Server.

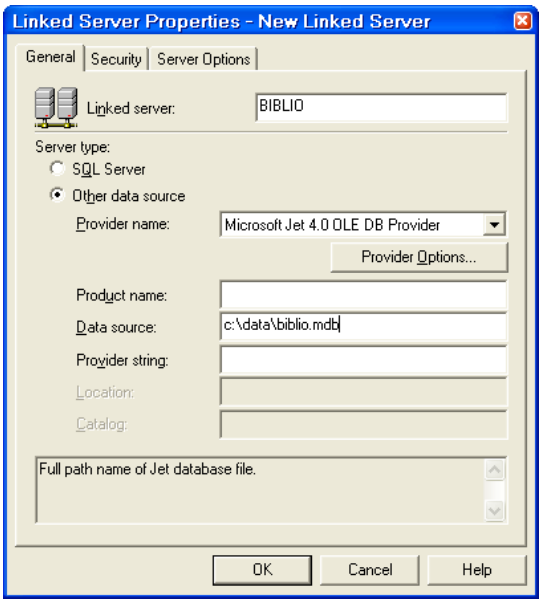
Từ SQL Server có thể tạo liên kết trực tiếp đến các hệ quản trị CSDL khác (Access, SQL Server, Oracle, My SQL,...) mà không cần thiết phải thông qua ODBC như Access đã xét trước.

Tạo bằng công cụ.

- Vào mục Security -> Linked Server.



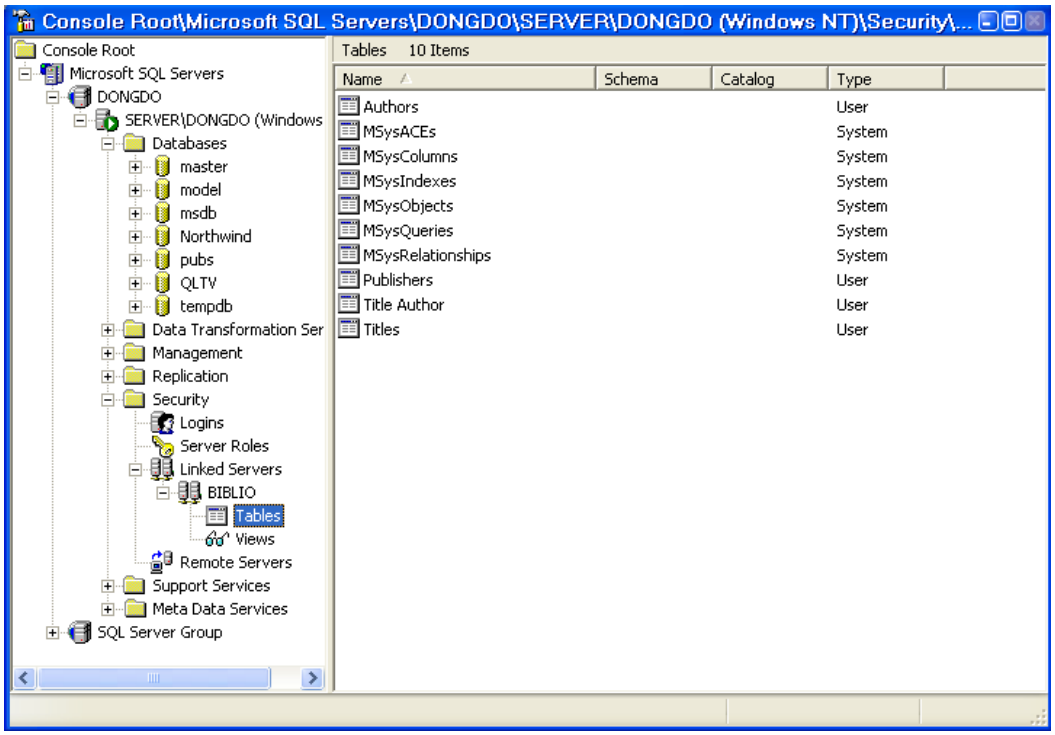
- Nhấn nút phải chuột -> New Linked Server.



- Nhập các tham số:

+ Tên Server.

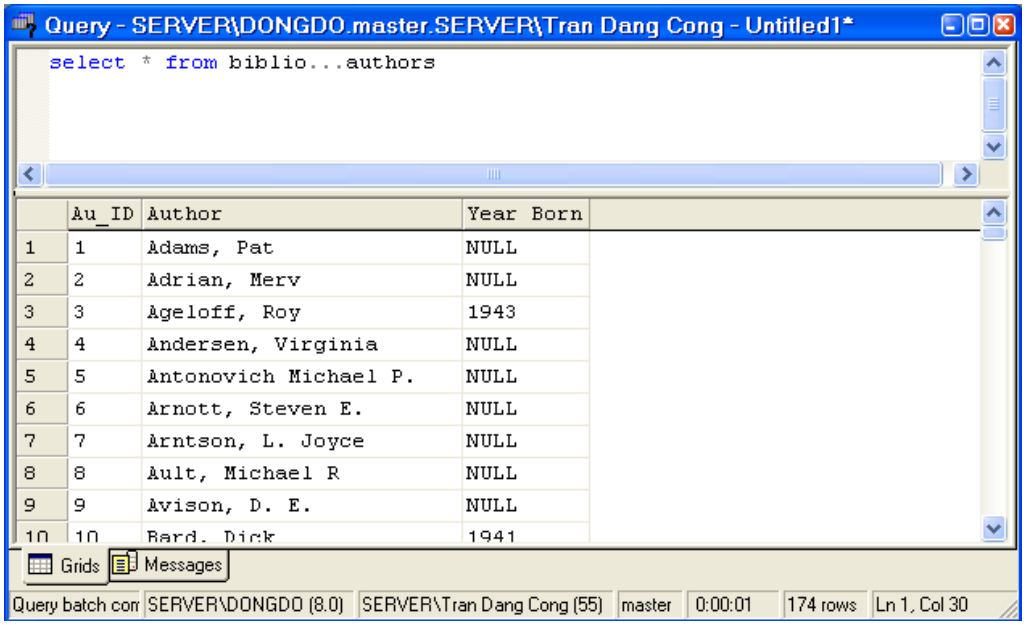
+ Provider (Driver của hệ quản trị CSDL cần thiết lập liên kết, trong ví dụ minh họa thực hiện với Access).



- Thực hiện khai thác thông qua câu lệnh, trong câu lSQL phải chỉ đủ đường dẫn, ví dụ

Select * from Biblio...Authors

sẽ thực hiện liệt kê toàn bộ danh sách các bản ghi của bảng authors.



Tạo bằng câu lệnh.

Sử dụng lệnh `sp_addlinkedserver` tạo server liên kết.

Ví dụ tạo Linked Server Biblio:

```
sp_addlinkedserver 'Biblio',
     'Access 97', 'Microsoft.Jet.OLEDB.4.0',
                              'c:\data\biblio.mdb'
```

(Đường dẫn phải phù hợp với Server)

Xóa Linked Server.

- Sử dụng công cụ: Chọn Linked Server cần xóa -> thực hiện xóa.

- Sử dụng lệnh :

sp_dropserver [**@server** =] '*server*'
[**,** [**@droplogins** =] { '**droplogins**' | NULL}]

SỬ DỤNG BCP VÀ BULK INSERT NHẬP DỮ LIỆU.

Bcp là câu lệnh dạng command prompt, dùng xuất (export) và nhập (import) dữ liệu giữa SQL Server và tập tin (dạng text hoặc excel). Các tập tin tham gia xuất nhập phải có cấu trúc dữ liệu kiểu bảng (hàng, cột), bảng dữ liệu của SQL Server khi thực hiện nhập dữ liệu phải có cấu trúc tương đương có sẵn.

Bulk insert là câu lệnh tương tự bcp nhưng chỉ thực hiện import dữ liệu mà không export.

Cú pháp lệnh bcp.

Lệnh bcp được thực hiện tại cửa sổ lệnh (command prompt).

bcp {[[*database_name***.**][*owner*]**.**]{*table_name* | *view_name*} | "*query*"}
{**in** | **out** | **queryout** | **format**} *data_file*
[**-m** *max_errors*] [**-f** *format_file*] [**-e** *err_file*]
[**-F** *first_row*] [**-L** *last_row*] [**-b** *batch_size*]
[**-n**] [**-c**] [**-w**] [**-N**] [**-V** (**60** | **65** | **70**)] [**-6**]

[**-q**] [**-C** *code_page*] [**-t** *field_term*] [**-r** *row_term*]
[**-i** *input_file*] [**-o** *output_file*] [**-a** *packet_size*]
[**-S** *server_name*[*****instance_name*]] [**-U** *login_id*] [**-P** *password*]
[**-T**] [**-v**] [**-R**] [**-k**] [**-E**] [**-h** "*hint* [**,**...*n*]"]

(bạn tìm hiểu thêm trong book online)

Ví dụ sử dụng lệnh bcp.

+ Sử dụng lệnh có từ khóa out để copy toàn bộ dữ liệu từ một bảng hoặc khung nhìn ra tập tin.

```
bcp pubs..titleview out titleview.txt -c -Sservername -
Uusername -Ppassword
```

+ Sử dụng lệnh Select để copy một tập ra tập tin, có từ khóa queryout.

```
bcp "SELECT au_fname, au_lname FROM pubs..authors ORDER
BY au_lname" queryout c:\Authors.txt -c -Sservername -
Uusername -Ppassword
```

Kết quả thực hiện: Nội dung tập tin Authors.txt

Abraham	Bennet
Reginald	Blotchet-Halls
Cheryl	Carson
Michel	DeFrance
Innes	del Castillo
Ann	Dull
Marjorie	Green
Morningstar	Greene
Burt	Gringlesby
Sheryl	Hunter
Livia	Karsen

Charlene Locksley
Stearns MacFeather
Heather McBadden
Michael O'Leary
Sylvia Panteley
Albert Ringer
Anne Ringer
Meander Smith
Dean Straight
Dirk Stringer
Johnson White
Akiko Yokomoto

Một số tham số cơ bản:

- Out: Copy toàn bộ một Table hoặc view ra tập tin.

- Queryout: Copy tập dữ liệu được truy vấn theo câu lệnh.

- c: Chỉ ra rằng câu lệnh dùng kiểu ký tự để phân định các cột, nếu không chỉ thì câu lệnh tự nhận tab (\t) để phân định và dùng new line để xuống dòng mới.

Cú pháp lệnh Bulk Insert.

Lệnh Bulk Insert gần giống lệnh bcp nhưng Bulk Insert chỉ sử dụng để nhập dữ liệu vào SQL Server (Insert), lệnh này được thực hiện bằng SQl Query Analyzer.

Cú pháp chung:

```
BULK INSERT [ [ 'database_name'.] [ 'owner' ].] { 'table_name' FROM 'data_file' }
   [ WITH
      (
         [ BATCHSIZE [ = batch_size ] ]
         [ [ , ] CHECK_CONSTRAINTS ]
         [ [ , ] CODEPAGE [ = 'ACP' | 'OEM' | 'RAW' | 'code_page' ] ]
```

```
      [ [ , ] DATAFILETYPE [ =
         { 'char' | 'native'| 'widechar' | 'widenative' } ] ]
      [ [ , ] FIELDTERMINATOR [ = 'field_terminator' ] ]
      [ [ , ] FIRSTROW [ = first_row ] ]
      [ [ , ] FIRE_TRIGGERS ]
      [ [ , ] FORMATFILE = 'format_file_path' ]
      [ [ , ] KEEPIDENTITY ]
      [ [ , ] KEEPNULLS ]
      [ [ , ] KILOBYTES_PER_BATCH [ = kilobytes_per_batch ] ]
      [ [ , ] LASTROW [ = last_row ] ]
      [ [ , ] MAXERRORS [ = max_errors ] ]
      [ [ , ] ORDER ( { column [ ASC | DESC ] } [ ,...n ] ) ]
      [ [ , ] ROWS_PER_BATCH [ = rows_per_batch ] ]
      [ [ , ] ROWTERMINATOR [ = 'row_terminator' ] ]
      [ [ , ] TABLOCK ]
   )
 ]
```

Ví dụ thực hiện copy toàn bộ dữ liệu tập tin newpubs.dat vào bảng publishers2 (bảng này đã có cấu trúc tương ứng), tập tin kiểu ký tự, ngăn cách giữa các cột là dấu phẩy ‘,’, xuống dòng mới bằng ký tự ‘\n’ (xuống dòng dưới và chuyển về đầu dòng).

```
BULK INSERT pubs..publishers2 FROM 'c:\newpubs.dat'
WITH (
   DATAFILETYPE = 'char',
   FIELDTERMINATOR = ',',
   ROWTERMINATOR = '\n'
)
```

DETTACH VÀ ATTACH CƠ SỞ DỮ LIỆU.

Mục này sẽ nói về ký thuật hủy và nối ghép tập tin CSDL với Server. Giả sử bạn đã có các tập tin của CSDL (gồm tập tin dữ liệu và nhật ký có thể được copy từ vị trí khác).

Copy tập tin của CSDL.

Trước tiên ta xem kỹ thuật copy các tập tin CSDL sang một vị trí khác (mà vẫn giữ vị trí), sau khi copy sang vị trí khác bạn có thể sử dụng sang Instance mới. Các bước thực hiện như sau:

- Stop dịch vụ SQL của Instance có CSDL.

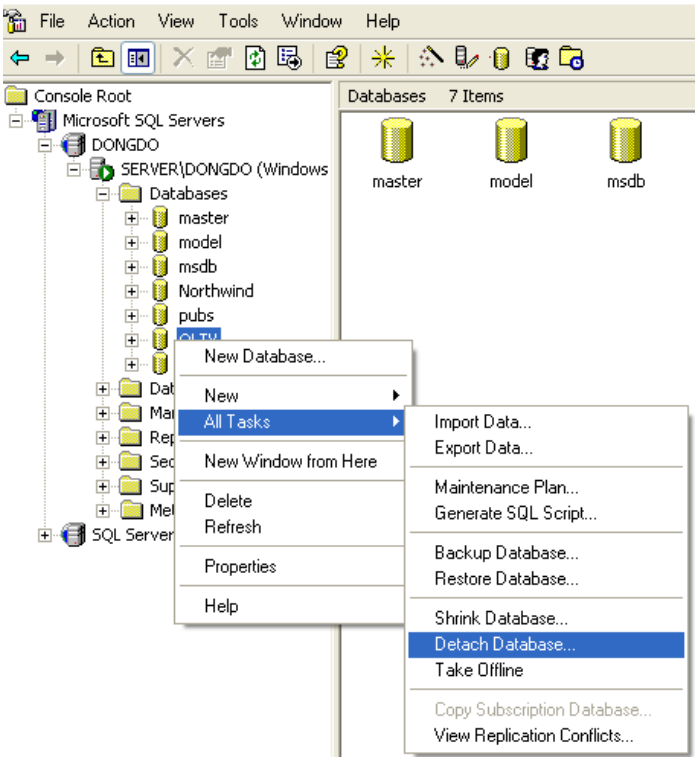
- Copy các tập tin của CSDL sang vị trí cần thiết.

- Start dịch vụ SQL của Instance để tiếp tục làm việc.

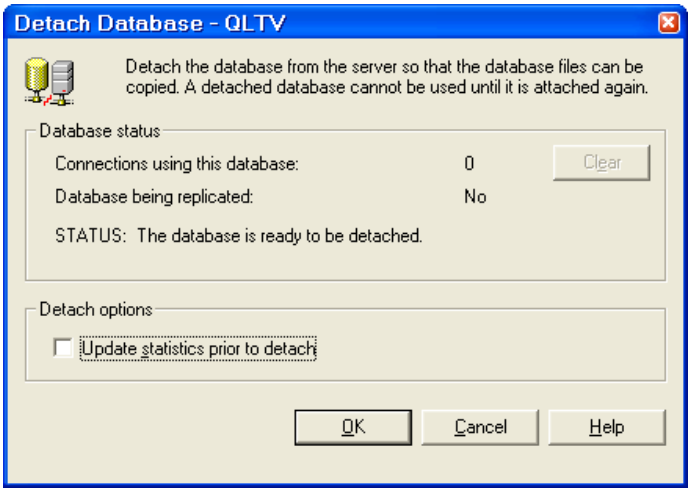
Dettach cơ sở dữ liệu.

Là bước thực hiện tách CSDL khỏi Instance, Ínntance không quản lý CSDL nhưng khác với xóa CSDL là các tập tin chứa CSDL vẫn còn.

- Chọn CSDL cần dettach. -> All tasks -> Dettach Database



- Nhấn Ok.

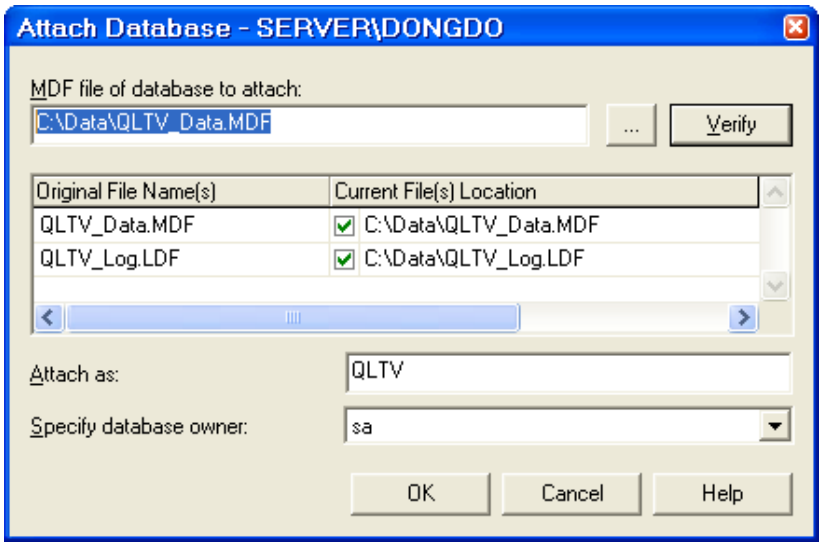


Attach tập tin CSDL vào Instance.

Mục này giới thiệu kỹ thuật ghép nối tập tin CSDL vào Instance, là bước tiếp theo của các bước Copy và Dettach. Các bước thực hiện như sau:

- Chọn Instance cần Attach CSDL -> Databases -> all tasks -> attach database...

- Chọn nút browse (...)



- Chọn tập tin mdf của CSDL cần attach.

- Đặt tên CSDL.

- Xác định User owner.

- Ok.

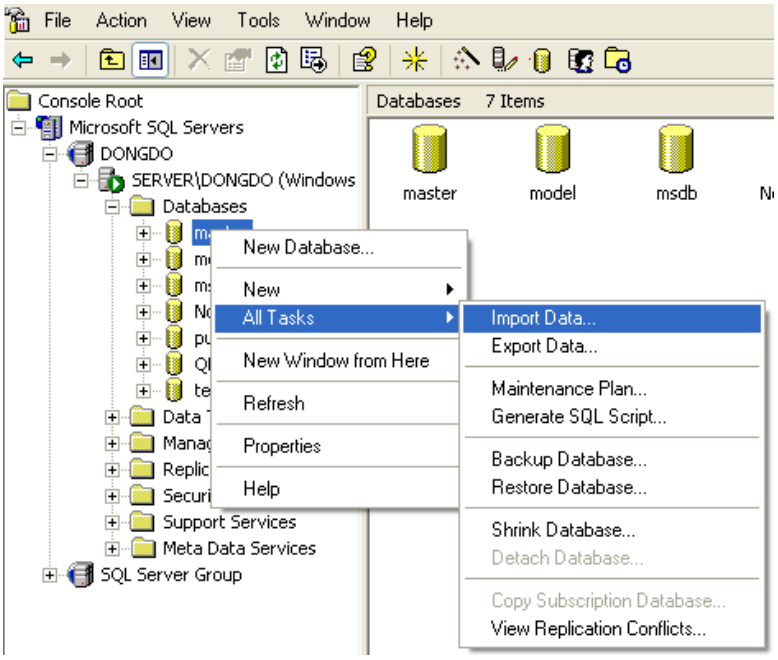
IMPORT VÀ EXPORT CƠ SỞ DỮ LIỆU.

Phần này sẽ trình bày kỹ thuật nhập và xuất dữ liệu từ CSDL với các hệ quản trị CSDL khác hoặc Instance, CSDL khác của SQL Server.

Import – Nhập dữ liệu.

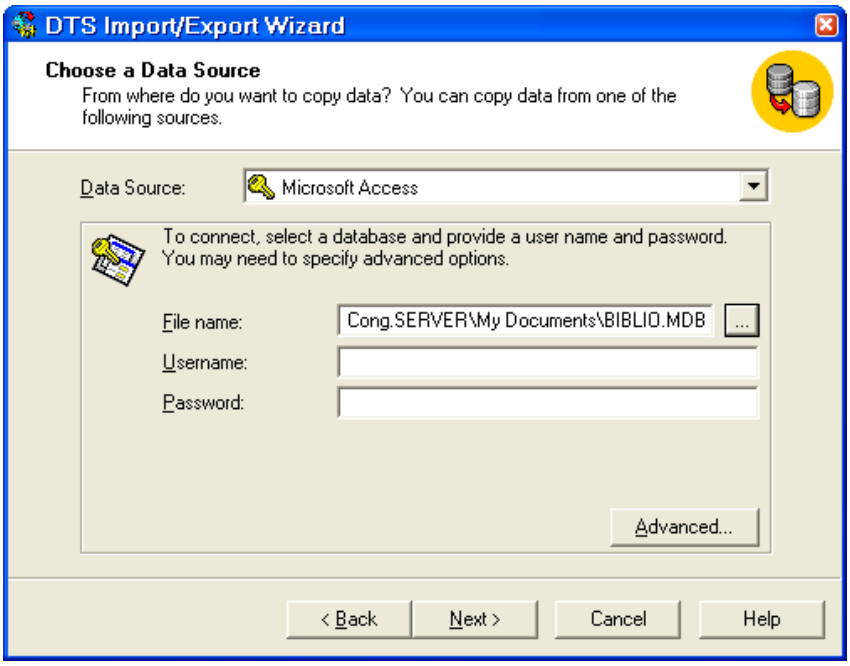
Dùng nhập dữ liệu từ ngoài vaof CSDL từ hệ quản trị CSDL khác hoặc CSDL khác của SQL Server.

- Chọn Databases -> All tasks -> Import Data...

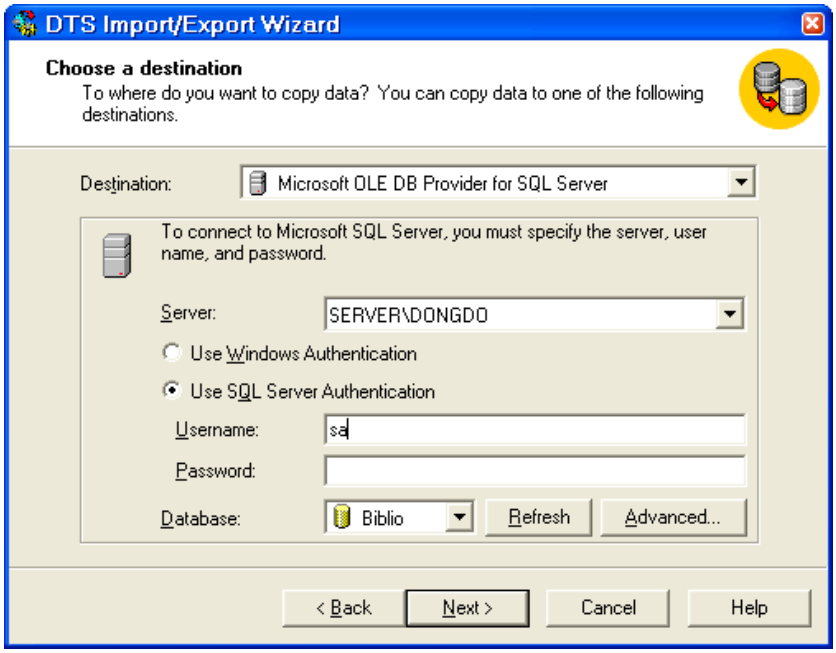


- Next -> Chọn Data Source (Có thể là SQL Server, Oracle, Access,...), trong ví dụ minh họa chọn Access.

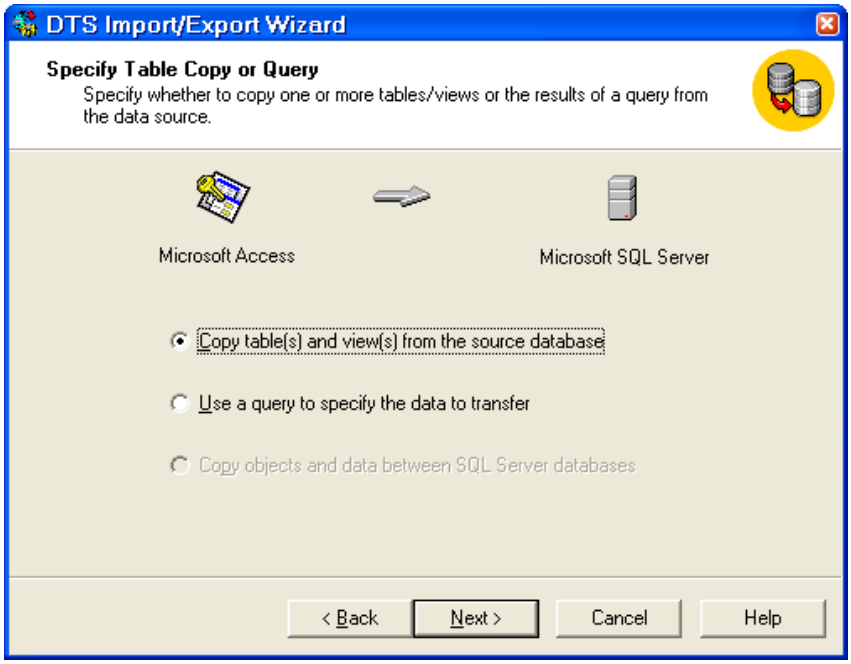
- Chọn tập tin (file name) -> Next



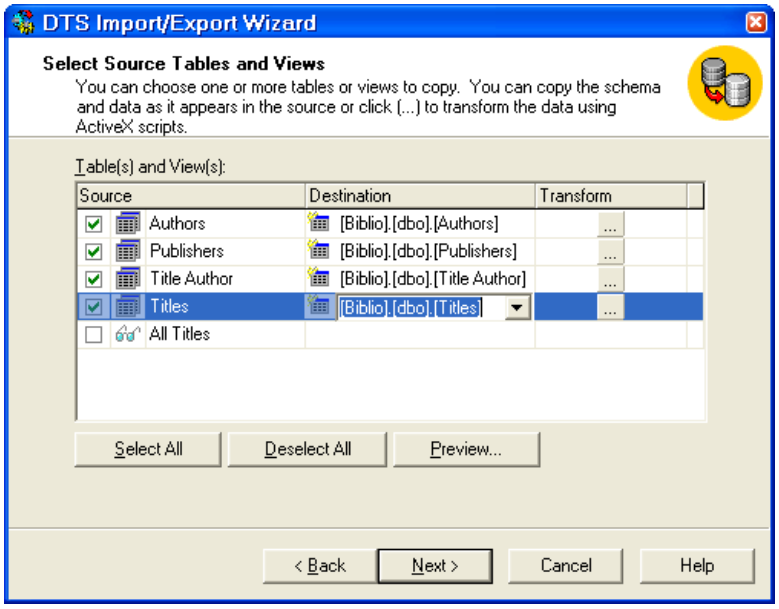
- Chọn Instance cần chuyển dữ liệu vào, user name., tên CSDL (có trước hoặc tạo tại thời điểm này bằng cách chọn New) -> Next



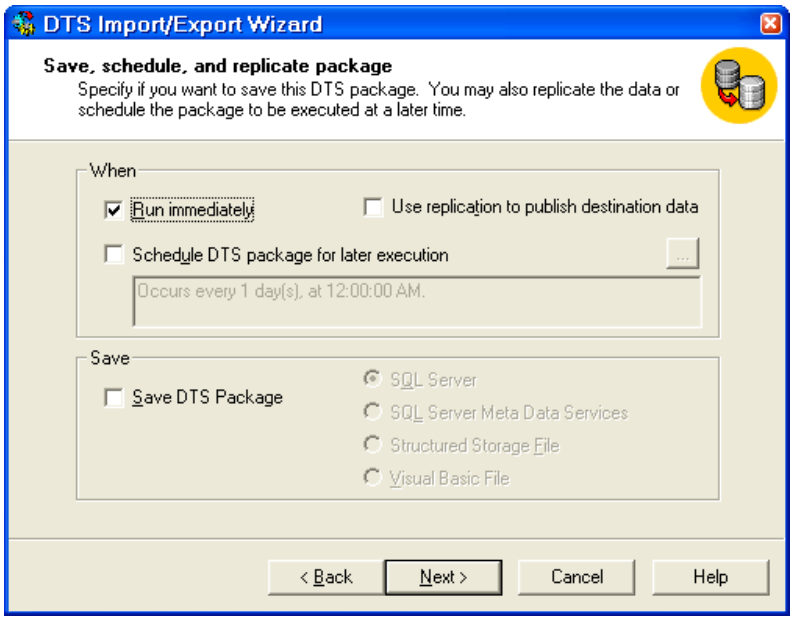
- Chọn cách chuyển toàn bộ bảng dữ liệu hay thông qua câu lệnh truyên vấn (trong ví dụ minh họa chọn bảng dữ liệu) -> Next



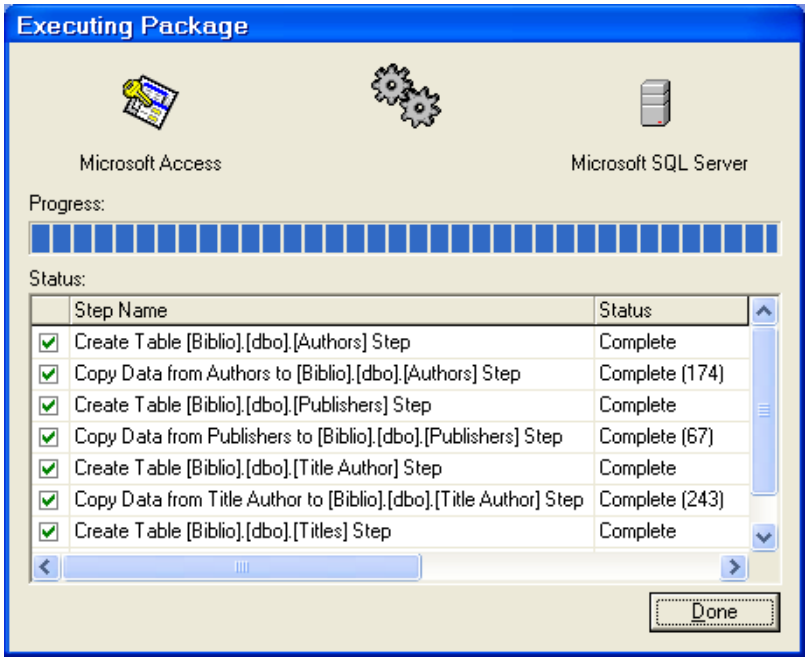
- Chọn các bảng, khung nhìn cần Import (có thể lựa chọn một số chức năng khác cụ thể hơn, bạn đọc tự tìm hiểu), tên các bảng, khung nhìn của SQL Server nhận dữ liệu -> Next.



- Chọn chức năng thực hiện ngày hay theo lịch –> Next -> Finish

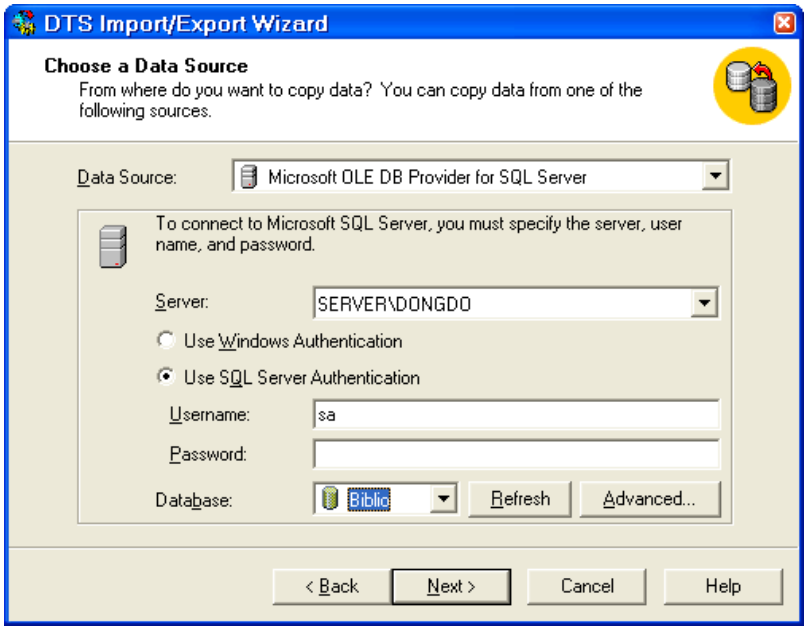


- Xem thông báo sau khi chuyển -> Done



EXPORT – XUẤT DỮ LIỆU.

Phần này giới thiệu kỹ thuật xuất dữ liệu từ một CSDL của SQL Server ra một hệ quản trị CSDL khác hoặc một CSDL khác của SQL Server. Tương tự như Import nhưng Export thực hiện Data Source là SQL Server, còn Destination là hệ quản trị CSDL khác hoặc CSDL khác của SQL Server (phần này bạn đọc tự xem xét).



SAO LƯU, KHÔI PHỤC DỮ LIỆU

Chương này sẽ giới thiệu kỹ thuật sao lưu (backup) và khôi phục (restore) dữ liệu, là kỹ thuật thường được sử dụng bảo đảm an toàn dữ liệu phòng trường hợp CSDL bị hỏng, nhật ký dữ liệu. Chức năng này được thực hiện bằng 2 phương pháp: Bằng công cụ và câu lệnh T-SQL.

NHỮNG LÝ DO PHẢI SAO LƯU VÀ KHÔI PHỤC DỮ LIỆU.

Trong quá trình thực hiện quản trị CSDL SQL Server thì một số nguyên nhân sau đây bắt buộc bạn phải xem xét đến kỹ thuật sao lưu và khôi phục dữ liệu:

+ Ổ đĩa bị hỏng (chứa các tập tin CSDL).

+ Server bị hỏng.

+ Nguyên nhân bên ngoài (thiên nhiên, hỏa hoạn, mất cắp,...)

+ User vô tình xóa dữ liệu.

+ Bị vô tình hay cố ý làm thông tin sai lệch.

+ Bị hack.

CÁC LOẠI BACKUP.

Backup dữ liệu trong SQL Server gồm các loại sau:

+ Full Database Backups: Copy toàn bộ CSDL (các tập tin bao gồm các bảng, khung nhìn, các đối tượng khác).

+ Differential Database Backups: Copy những dữ liệu thay đổi trong Data file kể từ lần full backup gần nhất.

+ File or file group backups: Copy một file đơn hay file group.

+ Differential File or File Group Backups: Thực hiện như Differential Database nhưng copy phần dữ liệu thay đổi của file đơn hoặc file group.

+ Transaction log backups: Ghi nhận tất cả các transaction chứa trong transaction log file kể từ lần transaction log backup gần nhất. Với loại sao lưu này ta có thể khôi phục dữ liệu tại một thời điểm.

CÁC MÔ HÌNH PHỤC HỒI DỮ LIỆU.

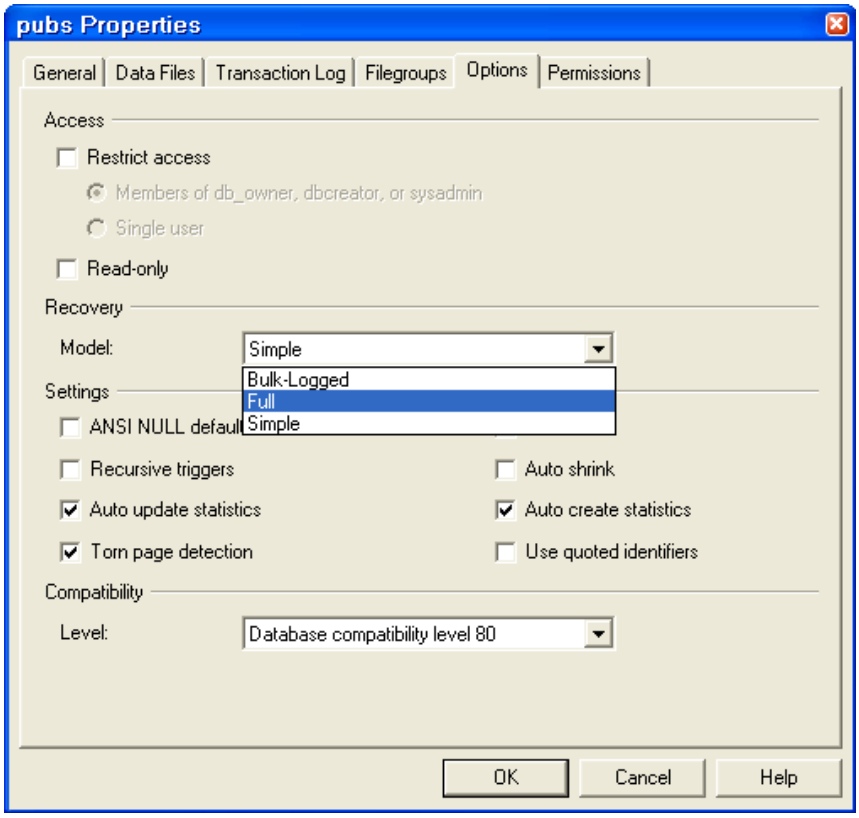
+ Full Recovery model: Là mô hình phục hồi toàn bộ hoạt động giao dịch của dữ liệu (Insert, Update, Delete, hoạt động bởi lệnh bcp, bulk insert). Với mô hình này ta có thể phục hồi dữ liệu tại một thời điểm trong quá khứ đã được lưu trong transaction log file.

+ Bulk-Logged Recovery Model: Mô hình này được thực thi cho các thao tác bcp, bulk insert, create index, writetext, updatetext, các hoạt động này chỉ nhật ký sự kiện vào log để biết mà không sao lưu toàn bộ dữ liệu, chi tiết như trong full recover. Các sự kiện Insert, Update, Delete vẫn được nhật ký và khôi phục bình thường.

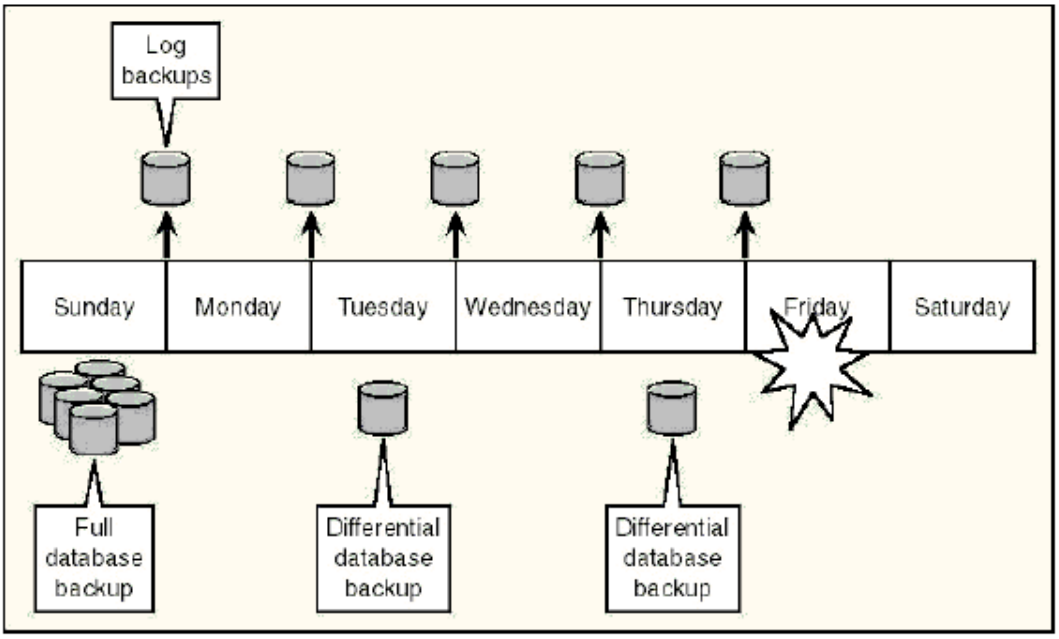
+ Simple Recovery Model: Với mô hình này bạn chỉ phục hồi lại thời điểm backup gần nhất mà không theo thời điểm khác trong quá khứ.

Cách đặt mô hình khôi phục:

- Chọn CSDL.
- Nhấn nút phải chuột -> Properties -> Options -> Recovery



Xét ví dụ sau: Giả sử ta có một CSDL được backup theo chiến lược như hình vẽ:



Nhìn hình trên ta thấy CSDL được lập lịch Full Database Backup vào ngày chủ nhật, Differential Database Backup vào ngày thứ ba và thứ năm, còn Log Database Backup vào 5 ngày trong tuần, ngày thứ sáu có sự cố với CSDL data file bị hỏng, vấn đề đặt ra là phải phục hồi dữ liệu và CSDL hoạt động bình thường. Ta phải làm các bước sau:

+ Thực hiện Backup log file (giả sử log file không bị hỏng).

+ Khôi phục Full Database của ngày chủ nhật.

+ Phục hồi Differential Database của ngày thứ năm.

+ Khôi phục Transaction log backup ngày thứ năm.

SAO LƯU CƠ SỞ DỮ LIỆU - BACKUP DATABASE.

Trước khi xem xét kỹ thuật sao lưu CSDL, ta thống nhất một số thuật ngữ bằng tiếng Anh như sau:

+ Backup: Là quá trình copy toàn bộ hoặc một phần database, transaction log, file, file group thành lập một backup set được chứa trong backup media (disk hoặc tape) bằng cách sử dụng một backup device (tape drive name hoặc physical filename).

+ Backup Device: Một file vật lý hoặc một drive tape.

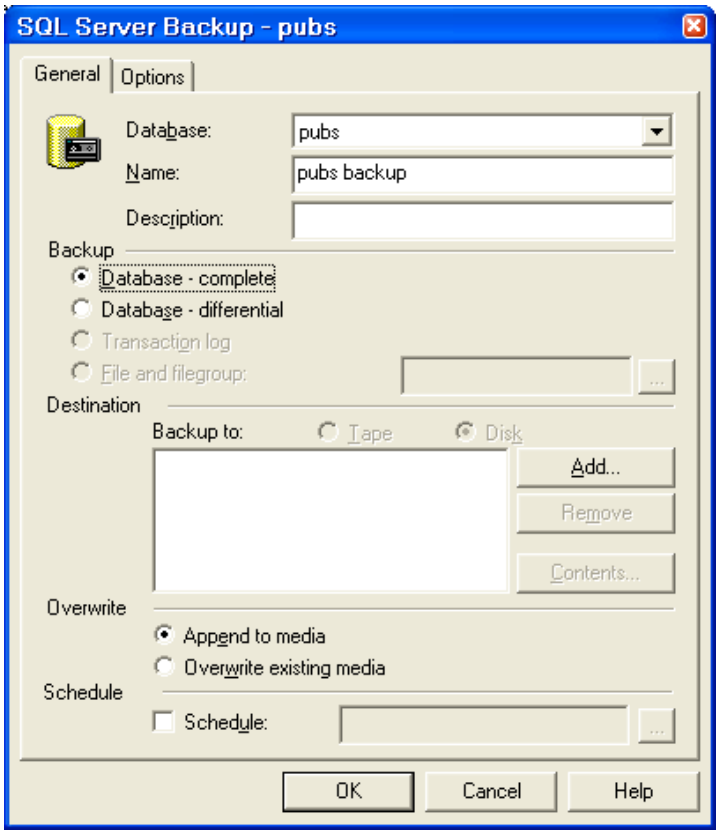
+ Backup file: Một file chứa Backup set.

+ Backup media: LÀ Disk hoặc tape.

+ Backup set: Một bộ backup một lần backup đơn chứa trên backup media.

Các bước thực hiện backup như sau:

- Chọn CSDL cần backup.
- Nhấn phải chuột -> All Tasks -> Backup Database…



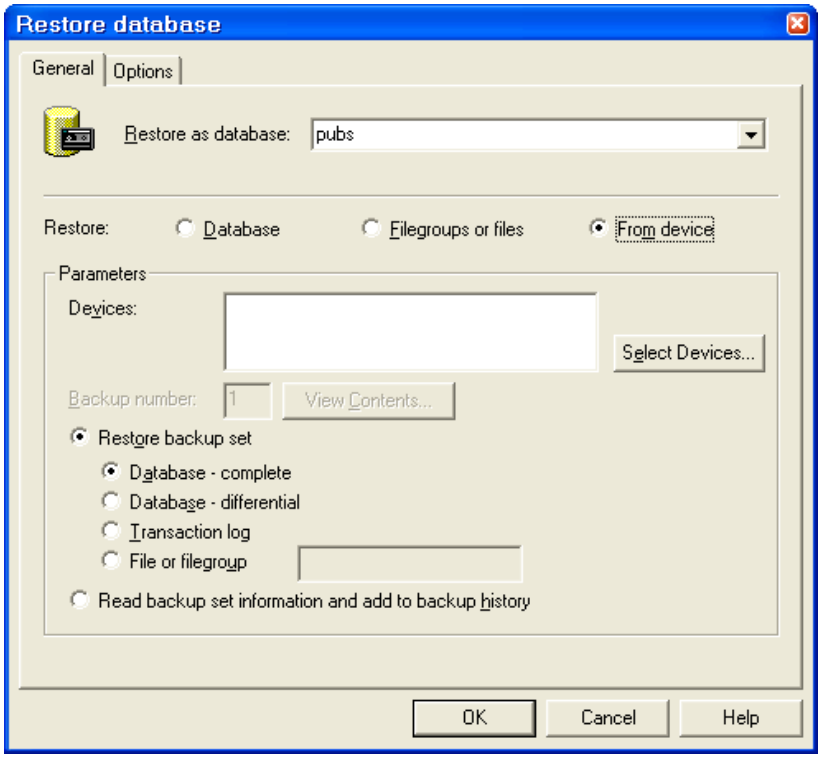
- Nhập các tham số, lựa chọn kiểu.

KHÔI PHỤC DỮ LIỆU – RESTORE DATABASE.

Là chức năng thực hiện khôi phục dữ liệu đã sao lưu, tùy theo chiến lược backup mà bạn có thể phục hồi đến thời điểm nào, thu được bộ dữ liệu trong quá khứ như thế nào. Khôi phục dữ liệu được thực hiện theo thứ tự backup, thông tin này được lưu trữ trong msdb

Các bước thực hiện như sau:

- Chọn mục Databases -> Nhấn nút phải chuột -> All Tasks -> Restore Database…



- Nhập tham số, chọn mô hình khôi phục.

PHÂN QUYỀN, BẢO MẬT

Chương này sẽ giới thiệu bạn đọc kỹ thuật phân quyền, quản lý người dùng, đặt các mức bảo mật cho CSDL.

CHẾ ĐỘ BẢO MẬT – SECURITY MODE.

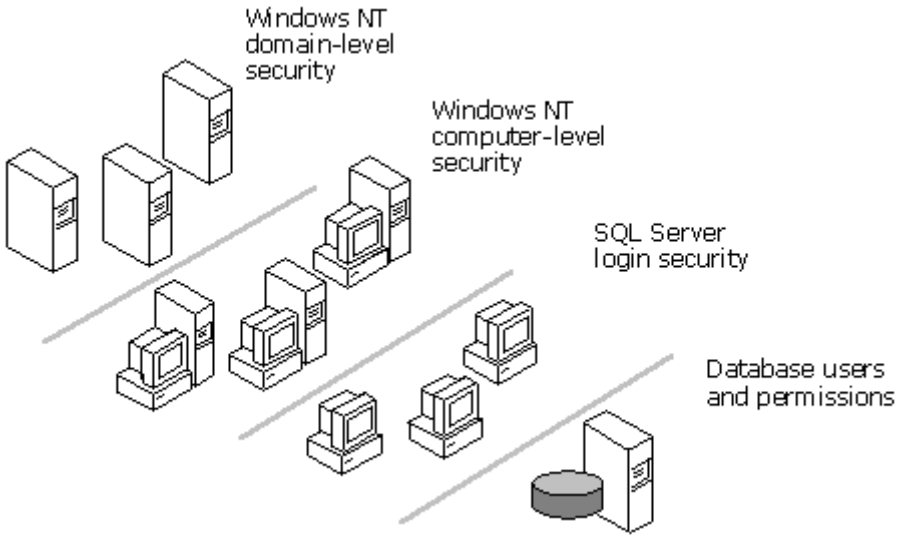
Như đã gặp trong phần cài đặt SQL Server, SQL Server có 2 chế độ bảo mật:

+ Windows Authentication Mode (Windows Authentication)

+ Mixed Mode (Windows Authentication and SQL Server Authentication)

Windows Authentication.

Là chế độ bảo mật mà những User truy nhập SQL Server phải là những User của Windows. Khi Server đặt ở chế độ bảo mật này, những User phải là những User được Windows quản lý mới được truy nhập.

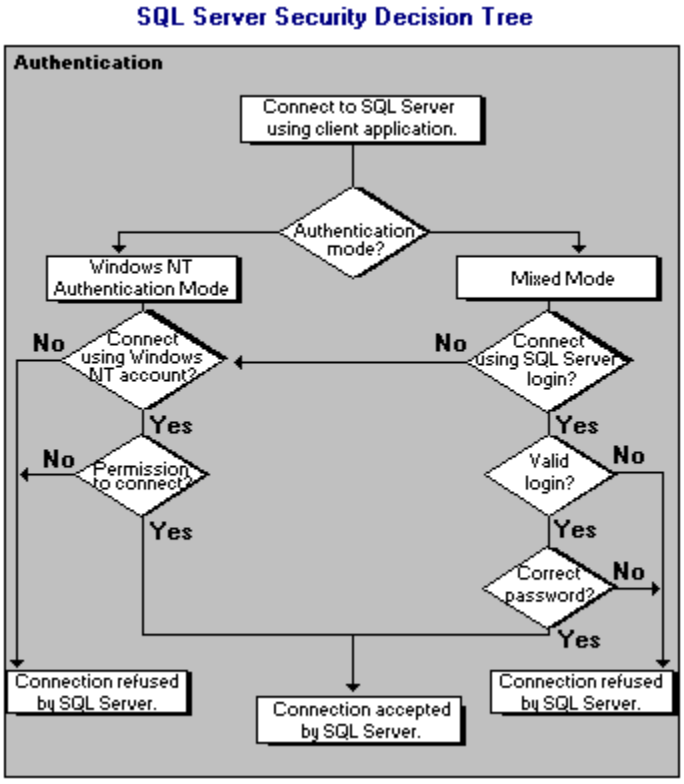


Nhìn trên hình ta thấy khi thực hiện chế độ này người sử dụng muốn khai thác SQL Server phải thông qua 4 bước xác thực (1- Domain, 2- Computer, 3- SQL Server, 4- Database).

SQL Server Authentication.

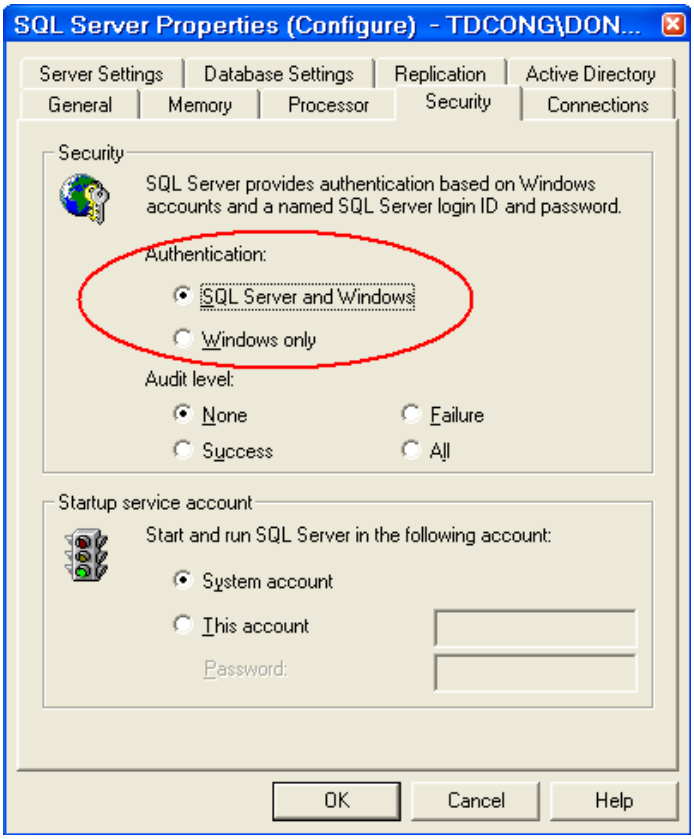
Khi thiết lập ở chế độ bảo mật này, những User được quyền khai thác phải là những User do quản trị SQL Server tạo ra, mà những user của Windows không được khai thác.

Tuy nhiên, SQL Server cho phép thiết lập hai chế độ Windows Authentication Mode (Windows Authentication) và Mixed Mode (Windows Authentication and SQL Server Authentication), chế độ Mixed Mode là sự kết hợp của Windows Authentication và SQL Server Authentication, ở chế độ này cả user của Windows và SQL Server để có thể thiết lập để truy nhập SQL Server.



Đặt chế độ.

- Nhấn phải chuột chọn tên Server (Instance).
- Chọn Properties.
- Chọn bảng Security.



- Chọn chế độ bảo mật -> Ok

SERVER ROLE, DATABASE ROLE.

Role là đối tượng xác định nhóm thuộc tính để gán quyền cho các user tham gia khai thác SQL Server.

Server Role.

Nhóm các quyền thực hiện quản trị hệ thống, gồm các nhóm sau:

+ Bulk Insert Administrators: Được phép thực hiện Bulk Insert.

+ Database Creators: Được phép tạo và sửa đổi cấu trúc CSDL.

+ Disk Administrators: Có thể quản trị các file trên đĩa.

+ Proccess Administrator: Quản trị các dịch vụ đang chạy của SQL Server.

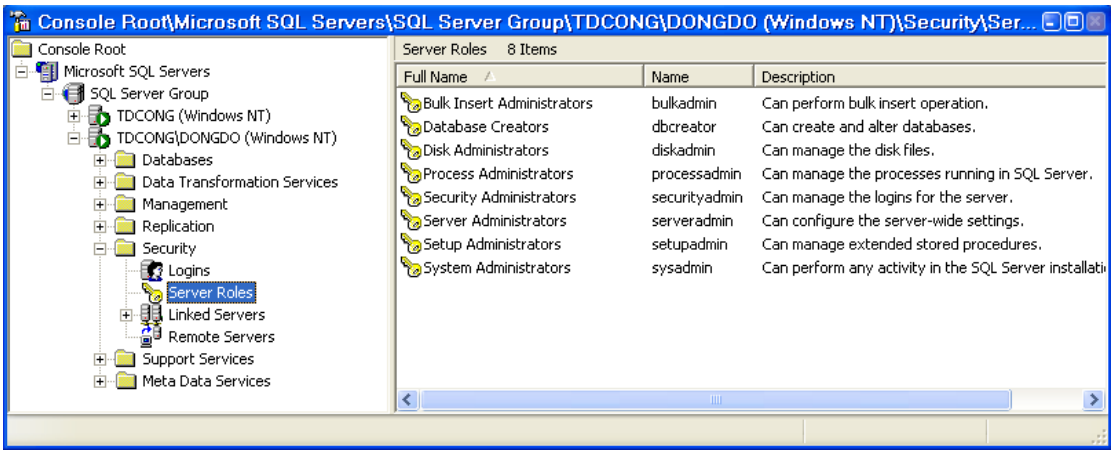
+ Security Administrators: Quản trị hệ thống bảo mật.

+ Setup Administrators: Quản trị các thủ tục mở rộng (xp_).

+ System Administrators: Quản trị hệ thống SQL Server.

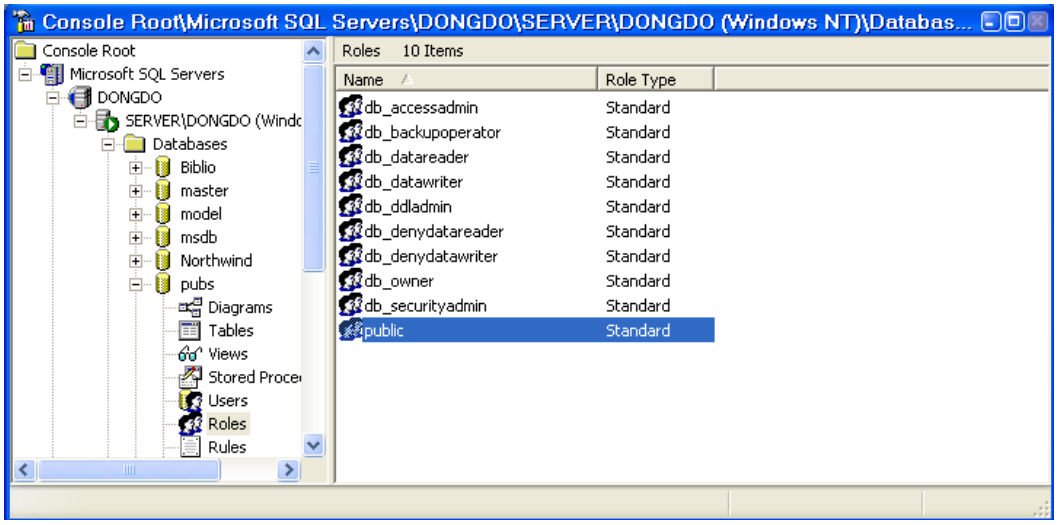
Xem cụ thể như sau:

- Mở rộng Server (nhấn dấu '+' phần tên Server).
- Mở rộng Security.
- Chọn Server Roles:



Database Role.

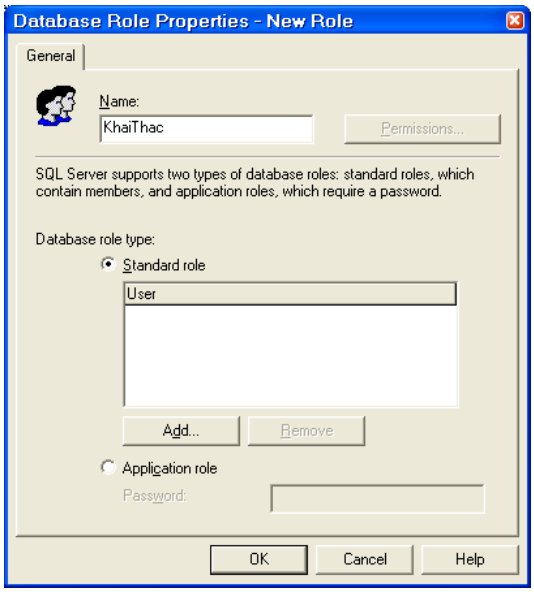
Role là đối tượng mà thông qua nó người quản trị có thể gán quyền khai thác cho người sử dụng. Role do CSDL quản lý, khi tạo CSDL hệ thống tự đặt một số Role ngầm định.



Người những Role ngầm định ta có thể tạo Role mới.

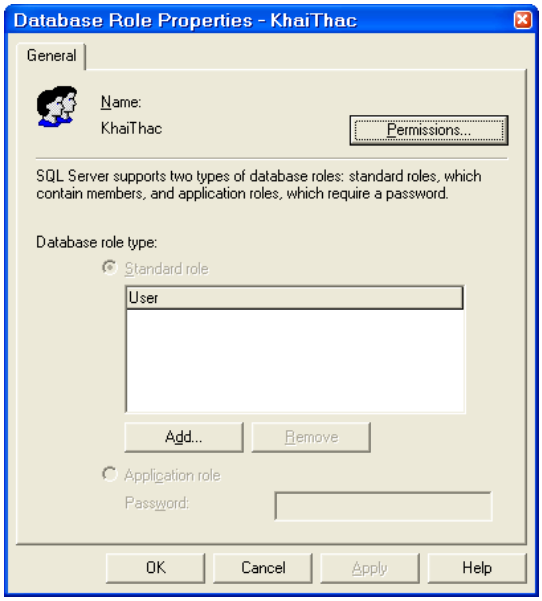
Tạo Role theo công cụ.

- Chọn Roles trong CSDL -> Nhấn phải chuột -> New Database Role..
- Đặt tên, chọn user (chọn user có thể làm sau).
- Nhấn Ok.

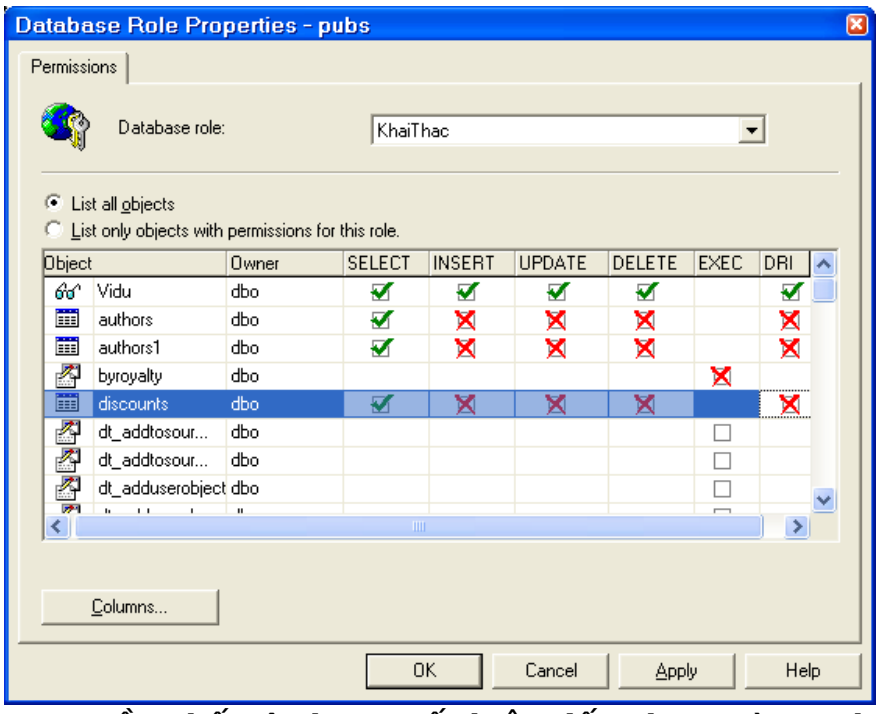


Sau khi tạo xong, thực hiện gán quyền khai thác cho Role.

- Chọn Role cần gán quyền.



- Chọn Perrmissions...
- Đặt các quyền cho từng đối tượng trong CSDL.



Nếu chọn quyền nhấn ô chọn xuất hiện dấu chọn màu xanh, nếu cấm nhấn ô chọn xuất hiện dấu màu đỏ. Có thể đặt quyền khai thác đối với role cho từng cột của bảng dữ liệu.

Mọi thao tác xóa, sửa được thực hiện như các đối tượng khác.

Tạo theo câu lệnh.

Sử dụng câu lệnh

sp_addrole [**@rolename** =] '*role*'
[**,** [**@ownername** =] '*owner*']

Ví dụ: Thêm Role có tên Managers:

```
EXEC sp_addrole 'Managers'
```

QUẢN TRỊ NGƯỜI DÙNG.

Người dùng trong SQL Server được chia thành 2 mức: Người truy nhập vào SQL Server gọi là Login, người khai thác CSDL gọi là User.

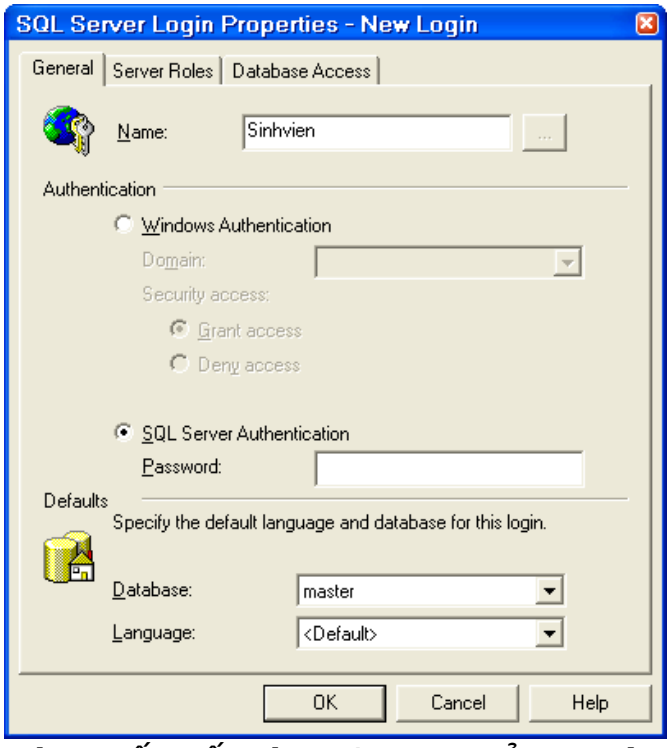
Login.

Là đối tượng được quyền truy nhập vào SQL Server, tùy theo chế độ bảo mật của SQL Server mà những login là account của Windows NT hay của SQL Server, login do Server quản lý trực tiếp.

Tạo bằng công cụ.

- Chọn chức năng Security của Server -> Logins

- Nhấn phải chuột -> New Login...



- Nhập các tham số: Nếu chọn Account của Windows NT thì bạn có thể chọn trong danh sách. Nếu tạo login của SQL Server thì bạn nhập tên mới, mật khẩu, chọn login thuộc server role nào, có thể gán quyền truy nhập khai thác CSDL nào.

Tạo bằng câu lệnh. Sử dụng câu lệnh

sp_addlogin [**@loginame** =] '*login*'
[**,** [**@passwd** =] '*password*']
[**,** [**@defdb** =] '*database*']
[**,** [**@deflanguage** =] '*language*']

[, [**@sid** =] *sid*]
[, [**@encryptopt** =] '*encryption_option*']

Ví dụ: Tạo login có tên ‘Albert’, mật khẩu ‘corporate’

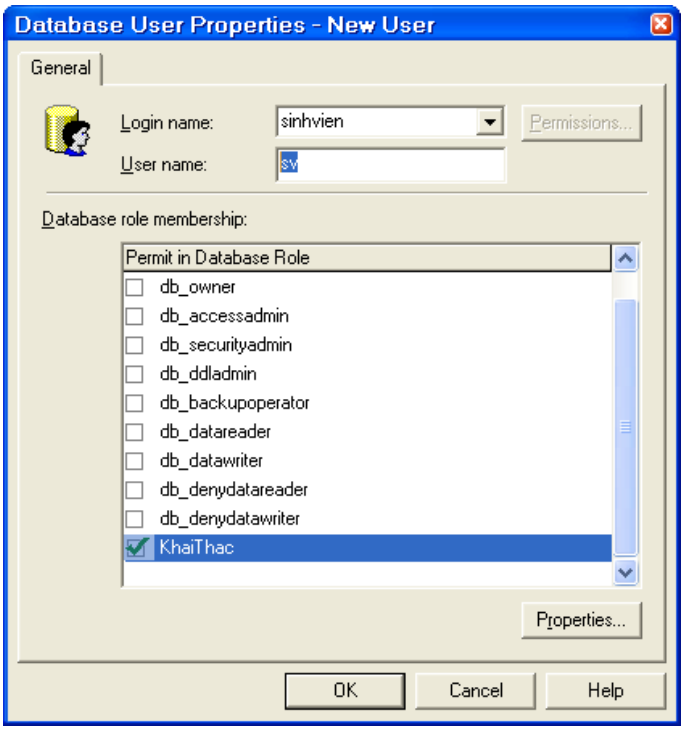
```
EXEC sp_addlogin 'Albert', 'food', 'corporate'
```

Mọi thao tác sửa, xóa được thực hiện như các đối tượng khác.

User.

User là đối tượng khai thác CSDL, nếu login chỉ xác định truy nhập vào SQL Server thì User là login ID tham gia khai thác CSDL, user do CSDL quản lý trực tiếp.

- Chọn CSDL -> users
- Nhấn phải chuột -> new user...



- Chọn Login, nhập user name, chọn role mà user thuộc ->Ok

Các thao tác xóa, sửa thực hiện như các đối tượng khác, để gán quyền cho user bạn có thể chọn lại user vừa tạo cho CSDL sau đó vào nhấn vào Permissions.

NHÂN BẢN DỮ LIỆU

Chương này bạn sẽ giới thiệu với bạn kỹ thuật làm giảm lưu lượng dữ liệu giao dịch với SQL Server khi đã cấu hình nhiều Server trên mạng.

GIỚI THIỆU VỀ NHÂN BẢN DỮ LIỆU.

Nhân bản dữ liệu tên tiếng anh gọi là Replication, là công cụ được sử dụng copy một hoặc nhiều CSDL đến một hoặc nhiều server (SQL Server) khác, các Server được đặt trong mạng máy tính nội bộ (LAN), người khai thác có thể thực hiện truy nhập đến CSDL có trong Server được chuyển dữ liệu đến. Dữ liệu giữa các máy được thực hiện đồng bộ với nhau theo lịch hoặc theo sự kiện, khi có yêu cầu. Nhân bản dữ liệu có những ưu điểm sau:

+ Dữ liệu được lưu trữ ở nhiều nơi, hiệu quả trong việc có nhiều ứng dụng cùng truy nhập, khai thác.

+ Thích hợp các ứng dụng phân tích dữ liệu OLTP của DataWare House.

+ Có thể khai thác dữ liệu khi không kết nối đến Server.

+ Giảm thiểu xung khắc do số lượng lớn các giao dịch trên mạng.

+ Là một giải pháp an toàn khi Server bị lỗi hoặc bảo dưỡng.

Mô hình nhân bản.

Dịch vụ nhân bản dữ liệu gồm các thành phần cơ bản sau: Publisher, Distributor, Subscribers, Publications, Articles, Subscriptions.

Publisher: Là server cung cấp dữ liệu nhân bản cho các server khác. Một publisher có thể thiết lập nhiều bộ dữ liệu nhân bản (gọi là publication).

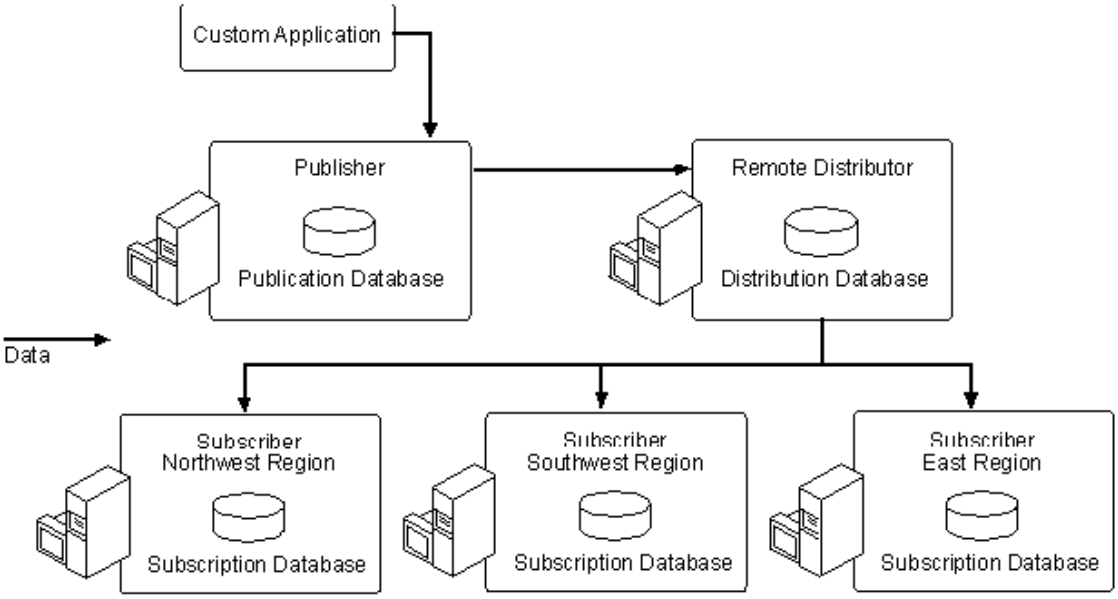
Distributor: Là server quản lý các thông tin nhân bản, lưu trữ dữ liệu trong các giao dịch thực hiện nhận và chuyển dữ liệu từ Publisher đến các Subscriber. Remote distributor là server tách rời khỏi publisher và được cấu hình là distributor. Local distributor là một server được cấu hình là Publisher và Distributor.

Subscriber: Là server nhận dữ liệu nhân bản. Subscriber gắn liền với publication (là máy chủ nhận dữ liệu nhân bản của một bộ dữ liệu cấu hình nhân bản).

Article: Là một bảng, tập dữ liệu hoặc đối tượng của CSDL cấu hình để nhân bản.

Publication: Là một tập gồm một hoặc nhiều article.

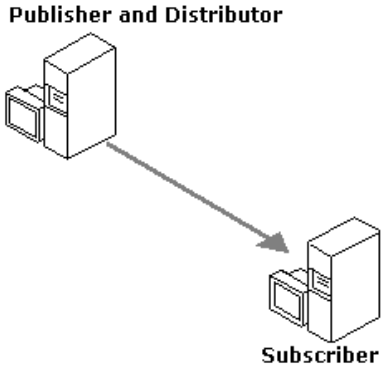
Subscription: Là một giao dịch yêu cầu bản sao bộ dữ liệu hoặc các đối tượng của CSDL thực hiện nhân bản. Trong mỗi giao dịch publisher thực hiện dẩy (push subscription) dữ liệu, subscriber thực hiện kéo (pull subscription).



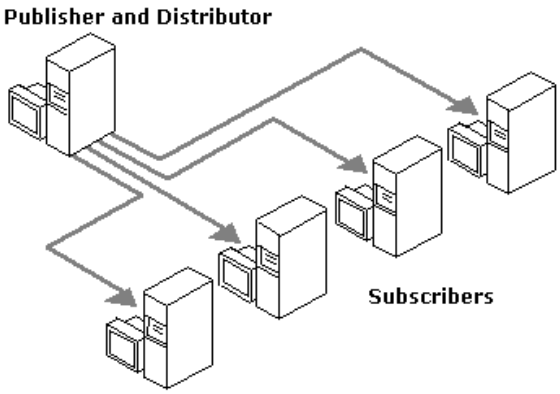
Nhân bản dữ liệu được thực hiện theo những mô hình cơ bản sau:

+ *Central Publisher*: Là mô hình Publisher và Distributor thiết lập trên một máy. Gồm các mô hình sau:

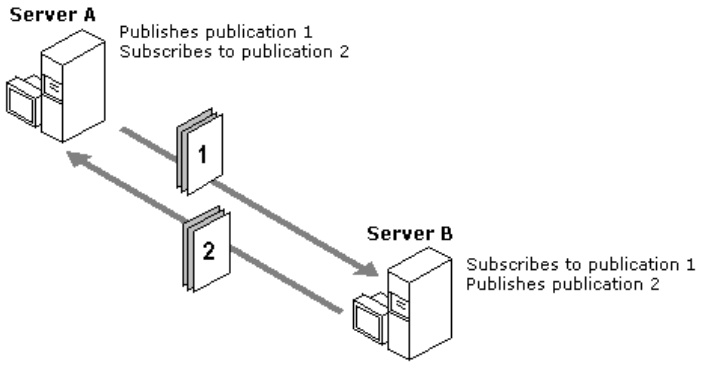
- Một Publishers và một Subscriber:



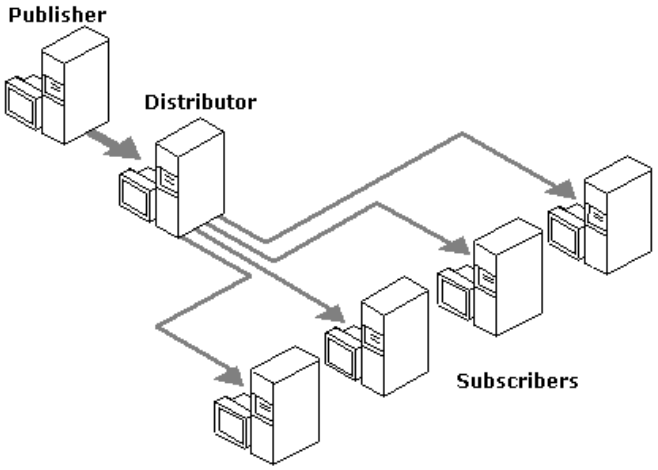
- Một Publisher và nhiều Subscriber.



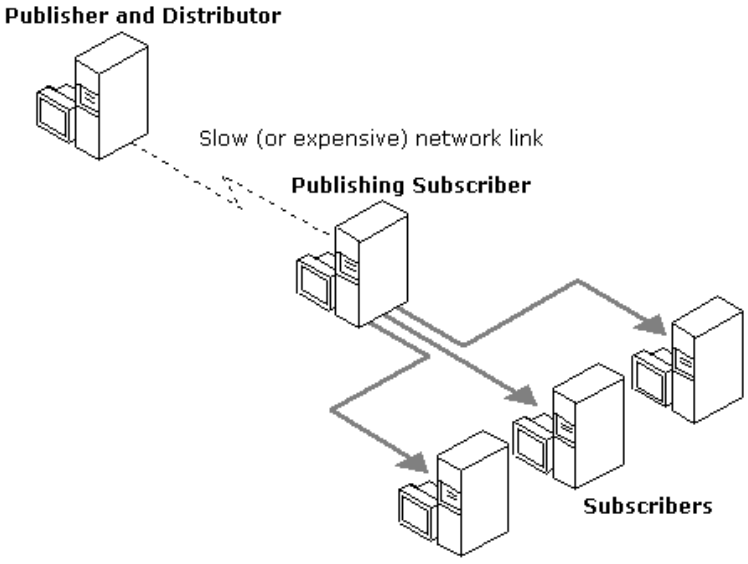
- Publisher và Subscriber được thiết lập trên một máy:



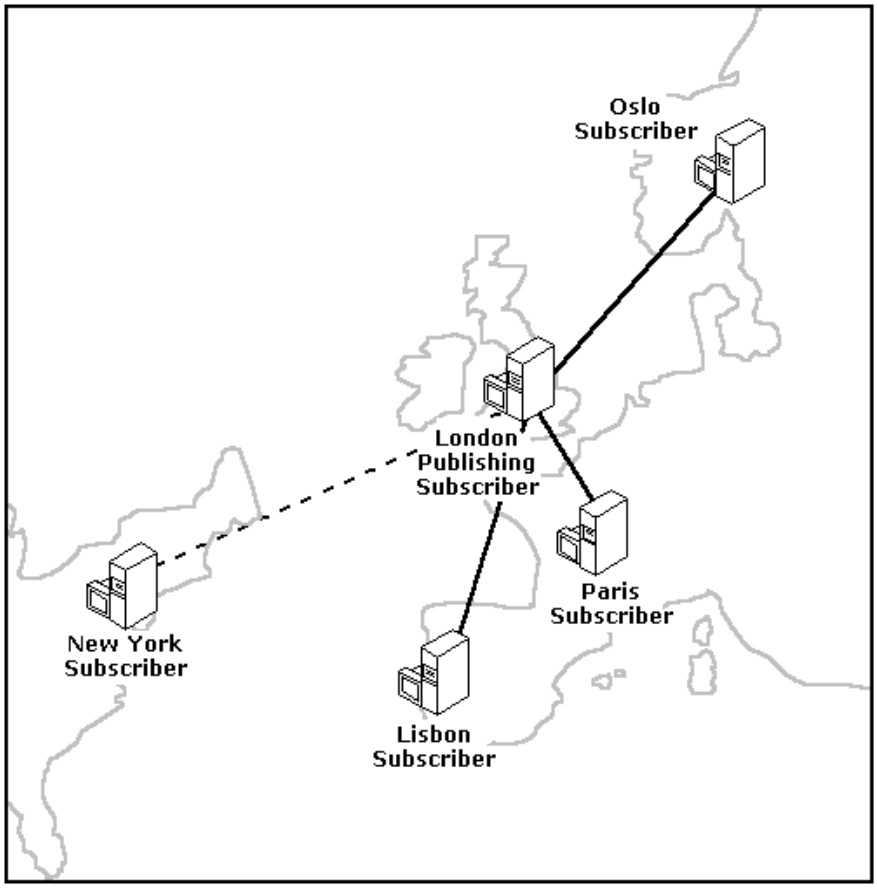
+ *Publisher và Distributor không thiết lập trên một máy*:



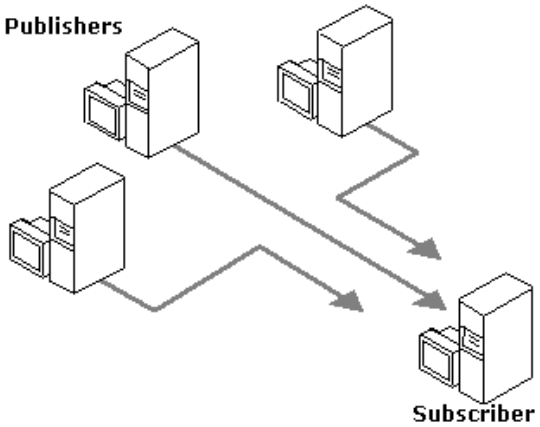
+ *Republisher*: Là mô hình Publisher xuất bản dữ liệu đến Subscriber, sau đó Subscriber được thiết lập là Publisher xuất bản dữ liệu đến Subscriber khác.



Đường truyền giữa hai máy được thiết lập là Publisher có thể tốc độ thấp, phù hợp với vị trí xa nhau. Ví dụ mô hình giữa các vùng cách xa nhau:



+ *Central Subscriber*: Là mô hình Subscriber thiết lậpn nhận dữ liệu xuất bản từ nhiều Publisher.



Những kiểu nhân bản dữ liệu.

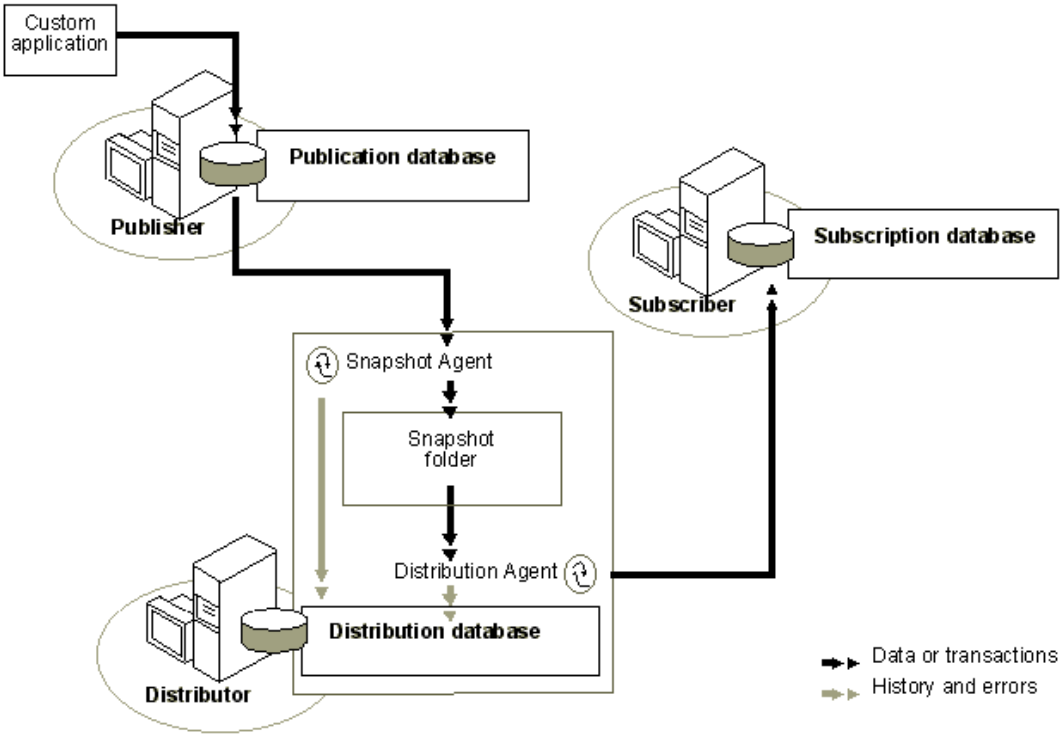
Có 3 kiểu nhân bản dữ liệu Snaphot, Transaction, Merge.

Snapshot replication: Là kiểu nhân bản thực hiện sao chép, phân tán dữ liệu hoặc các đối tượng của CSDL tạo một thời điểm.

Snapshot thường được sử dụng cho những tình huống sau:

+ Dữ liệu thường là tĩnh, ít thay đổi.

+ Nhân bản số lượng dữ liệu nhỏ.



Transaction replication: Là kiểu nhân bản mà bắt đầu bằng nhân bản snapshot, sau đó sẽ thực hiện nhân giao dịch dữ liệu theo các sự kiện insert, update, delete và những thay đổi liên quan đến thực hiện stored procedure, index view.

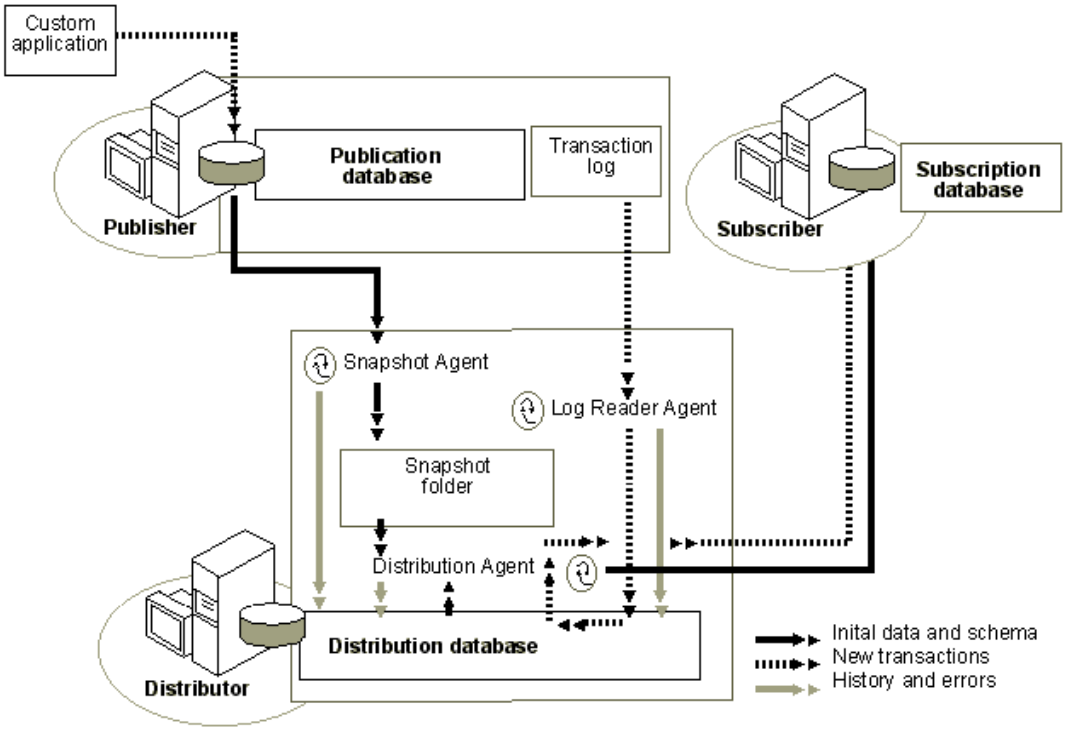
Nhân bản kiểu này cho phép thực hiện lọc dữ liệu tại xuất bản, cho phép user sửa đổi dữ liệu nhân bản tại subscriber và chuyển dữ liệu đã sửa đổi đến Publisher hoặc Subscriber khác, dữ liệu sửa đổi này có thể coi là dữ liệu được xuất bản.

Nhân bản kiểu này được thực hiện khi:

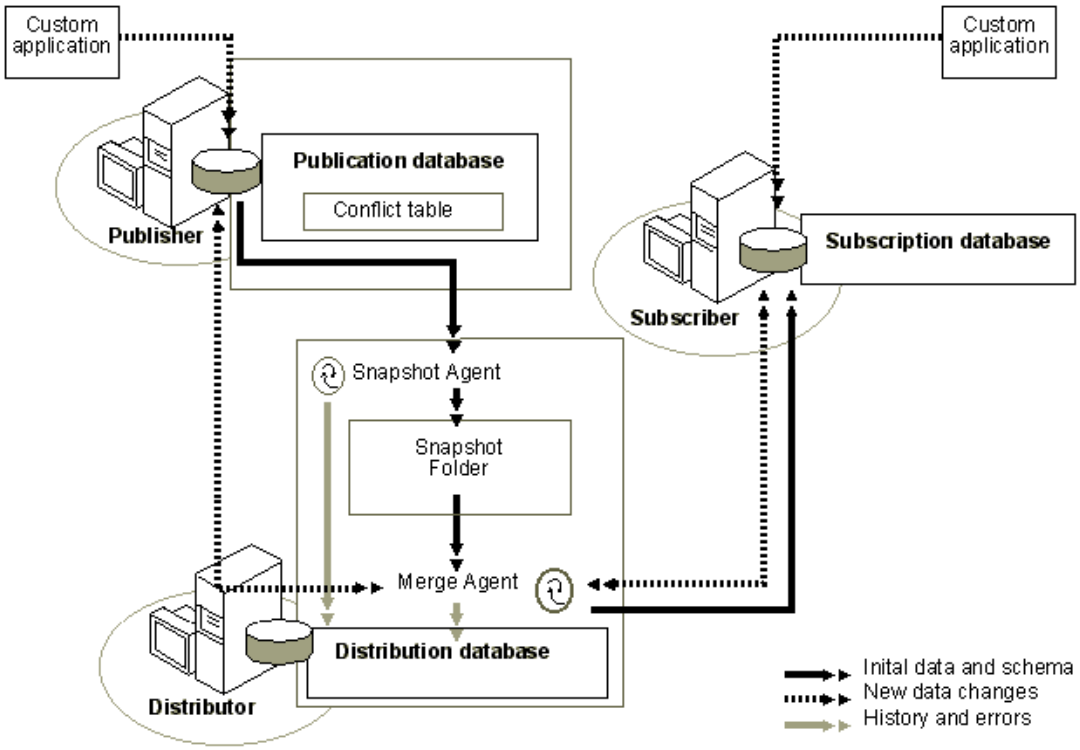
+ Muốn sửa đổi dữ liệu được xuất bản chuyển đến Subscriber, thời gian thực hiện theo giây, hoặc tức thời.

+ Cần giao dịch trên toàn bộ hệ thống nhân bản dữ liệu (dữ liệu có thể chuyển đến tất cả các Subscriber hoặc không chuyển đến Subscriber nào).

+ Subscriber thường xuyên kết nối với Publisher.



Merge replication: Là kiểu nhân bản dữ liệu cho phép thực hiện nhân sửa đổi dữ liệu trên nhiều Subscriber, có thể kết nối (online) hoặc không kết nối (offline)



đến Publisher. Dữ liệu sẽ được đồng bộ theo lịch hoặc theo yêu cầu, dữ liệu cập nhật có thời điểm sau sẽ được chấp nhận.

Kiểu nhân bản này thực hiện khi:

+ Nhiều Subscriber có nhu cầu cập nhật dữ liệu và chuyển dữ liệu cập nhật đến Publisher hoặc Subscriber khác.

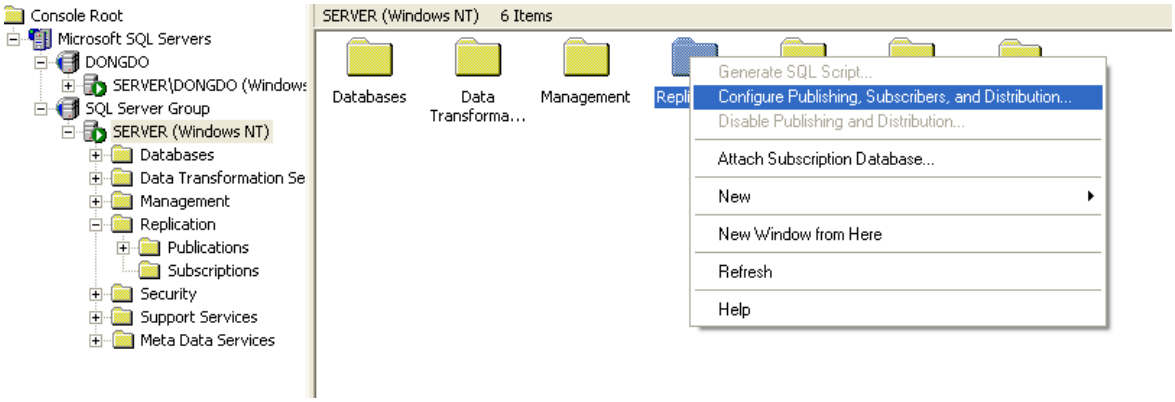
+ Subscriber yêu cầu nhận hoặc chuyển dữ liệu khi offline, đồng bộ dữ liệu với các Subscriber và Publisher sau.

CẤU HÌNH PUBLISHER VÀ DISTRIBUTOR.

Trước khi thực hiện cấu hình các máy thành Publisher hay Distributor ta phải thực hiện chạy dịch vụ SQL Server Agant trong chức năng Service manager. các bước cấu hình như sau:

- Chọn Server cần cấu hình -> Replication

- Nhấn phải chuột -> Configure Publishing Subscription and Distribution...

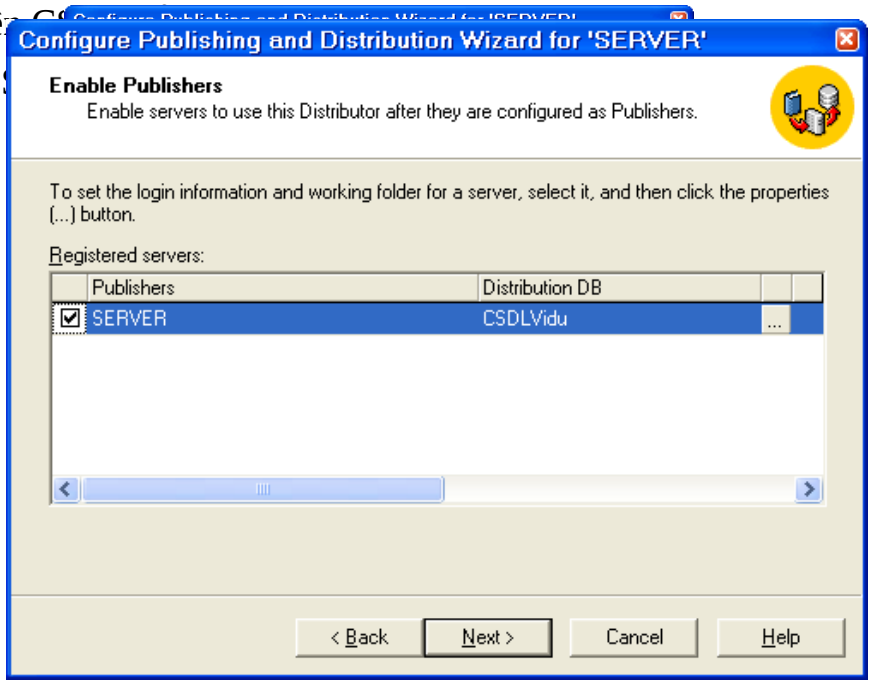


- Thực hiện thao các bước:

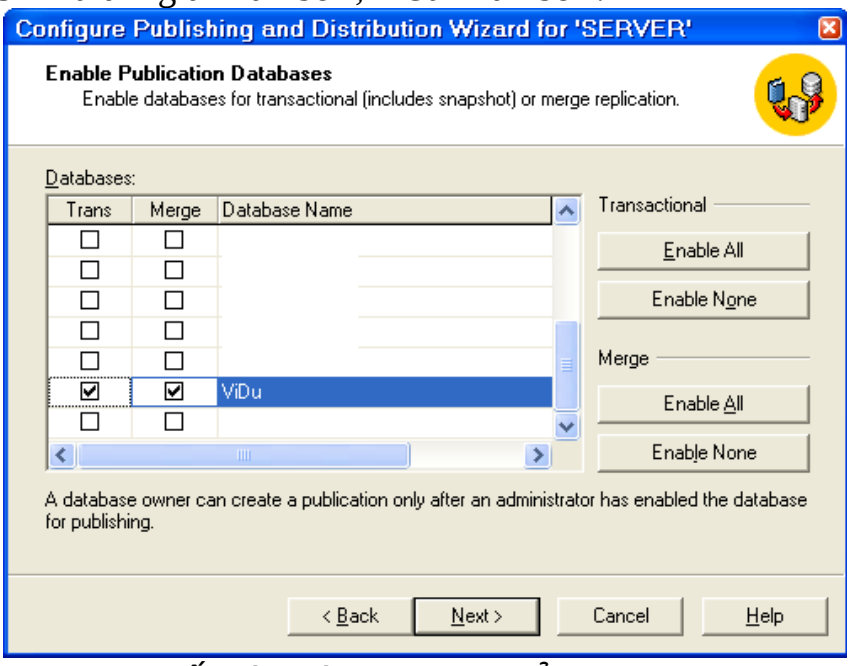
+ Chọn thư mục Snapshot: Thư mục này sẽ sử dụng cho

- Đặt tê

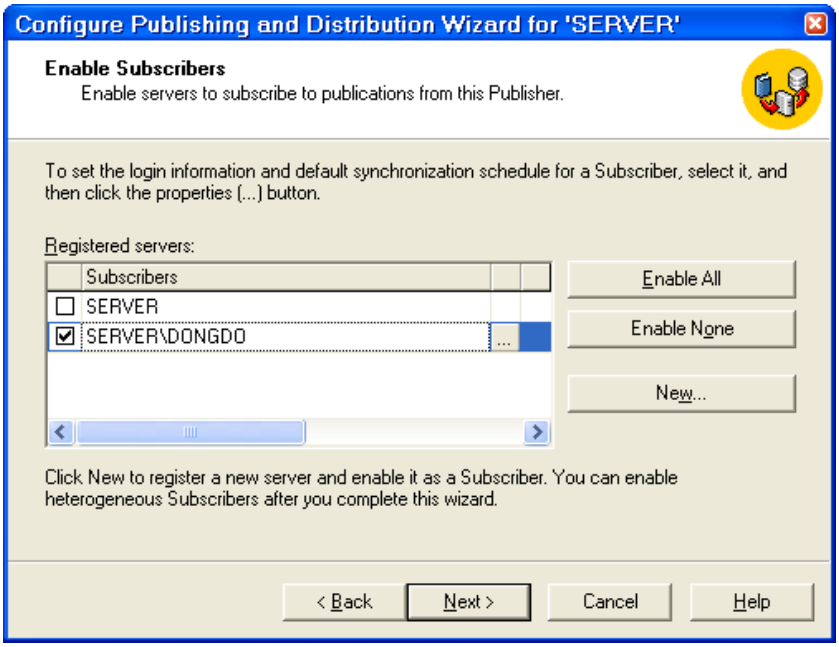
- Chọn



- Chọn CSDL tham gia nhân bản, kiểu nhân bản.



- Chọn Server được cấu hình là Subscriber của Publisher đang thiết lập.



- Kết thúc.

TẠO PUBLICATION.

Bước này sẽ thực hiện tạo Publication, cách thực hiện như sau:

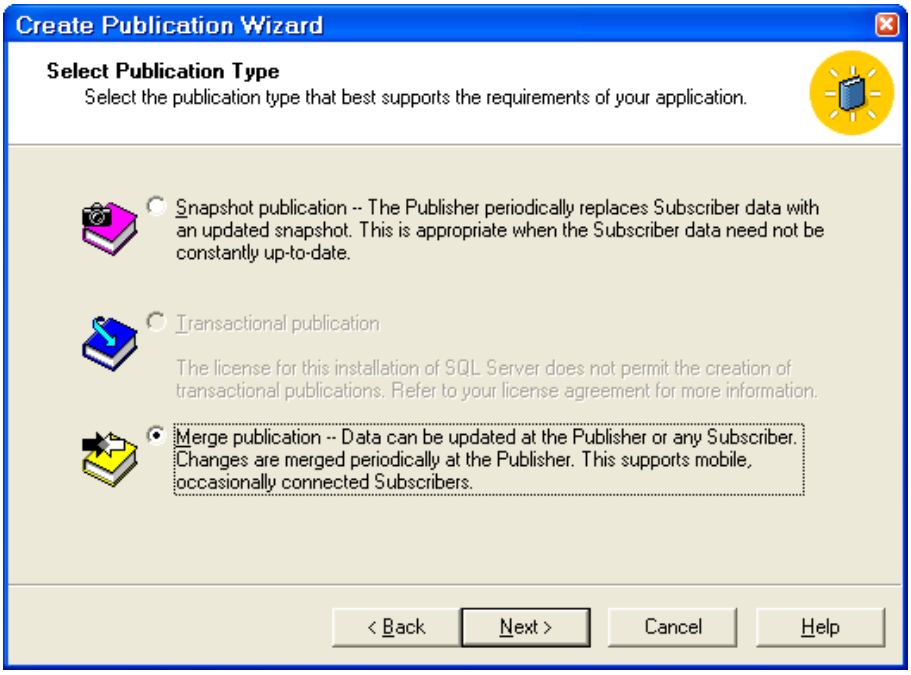
+ Chọn Publication trong Replication của Publisher.

+ Nhấn phải chuột -> New Publication...

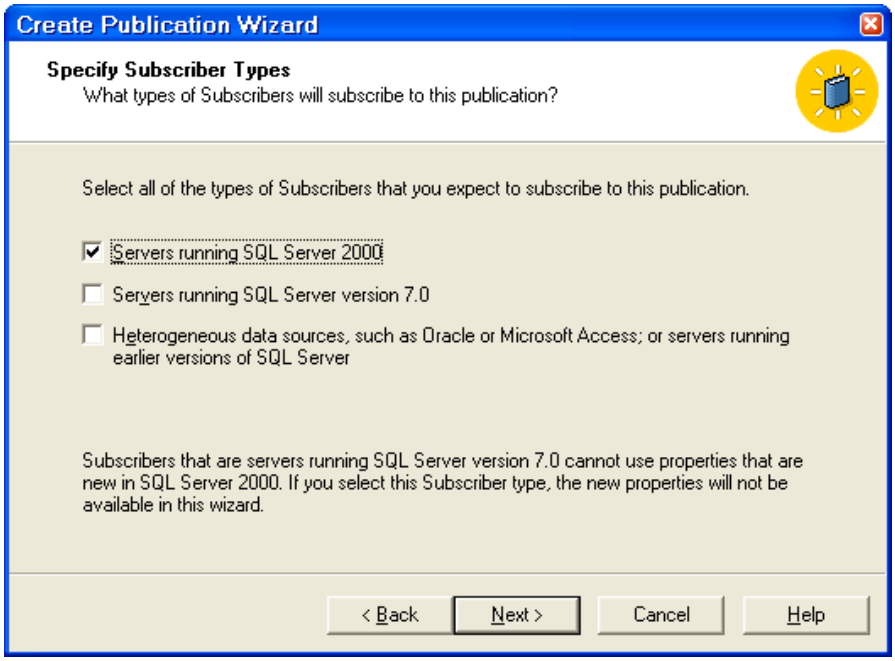
+ Thực hiện theo các bước:

- Chọn CSDL cần xuất bản dữ liệu hoặc đối tượng.

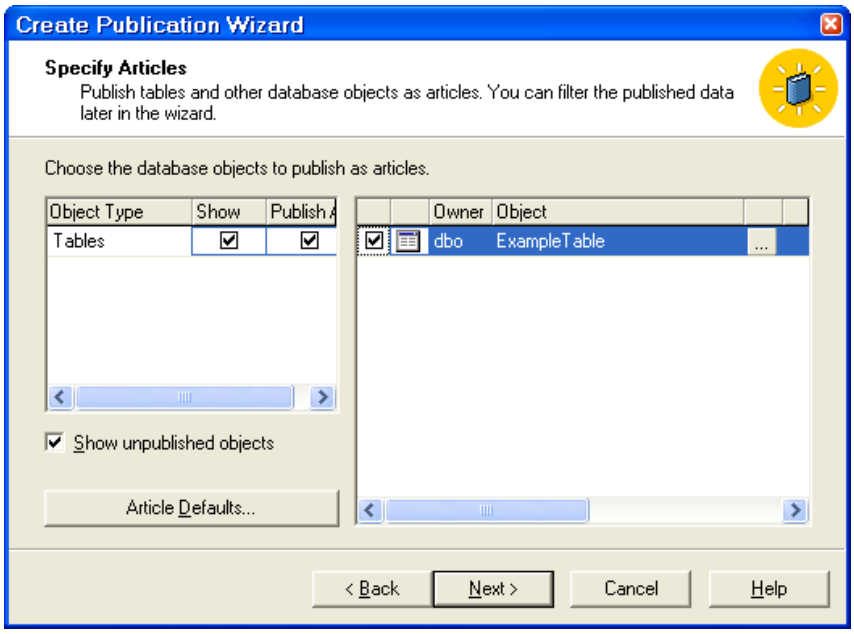
- Chọn kiểu nhân bản (trong ví dụ này thực hiện kiểu Merge)



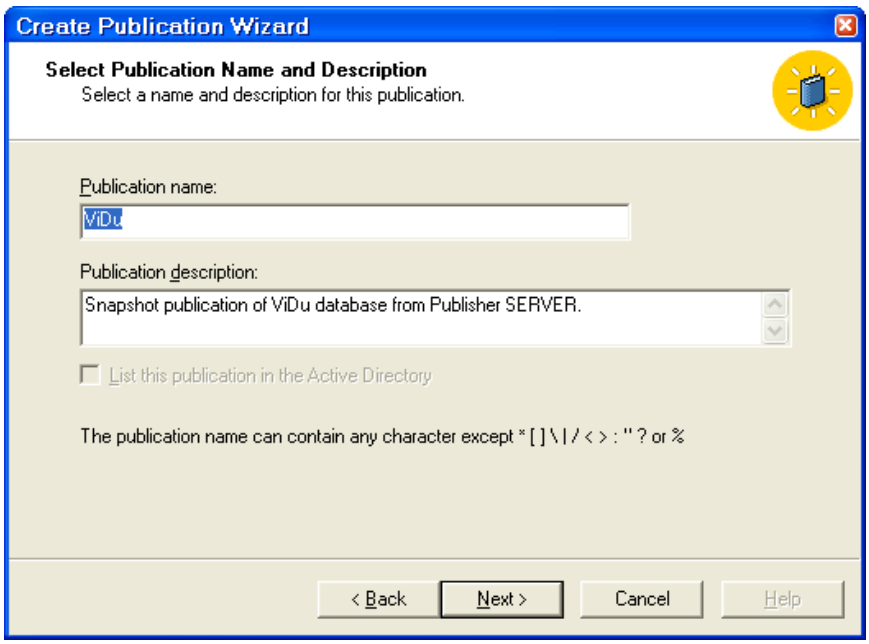
- Chọn phiên bản SQL Server của Subscriber.



- Chọn Article tham gia Publication.



- Đặt tên cho Publication.



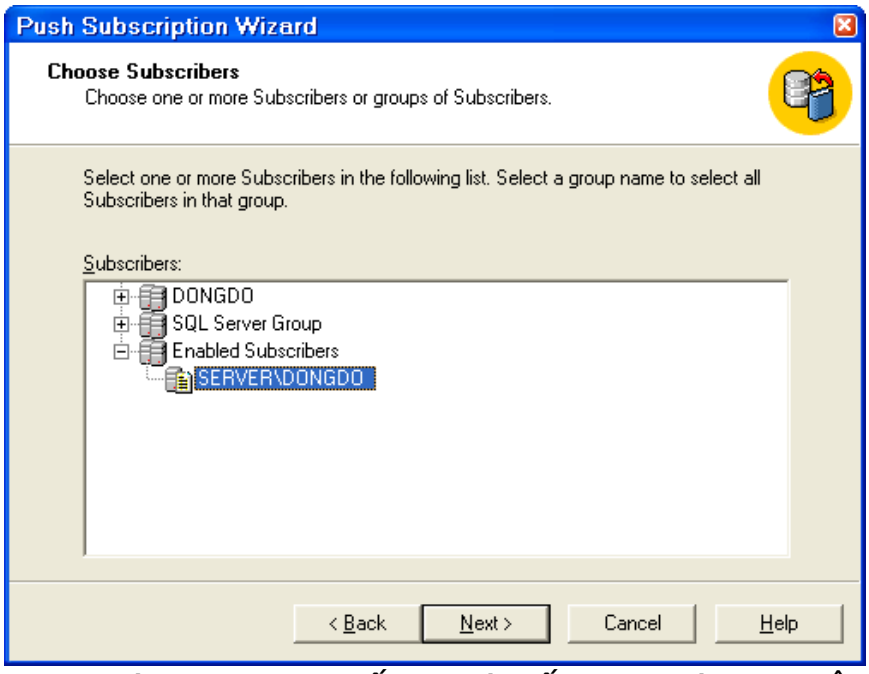
- Kết thúc.

TẠO PUSH SUBSCRIPTION.

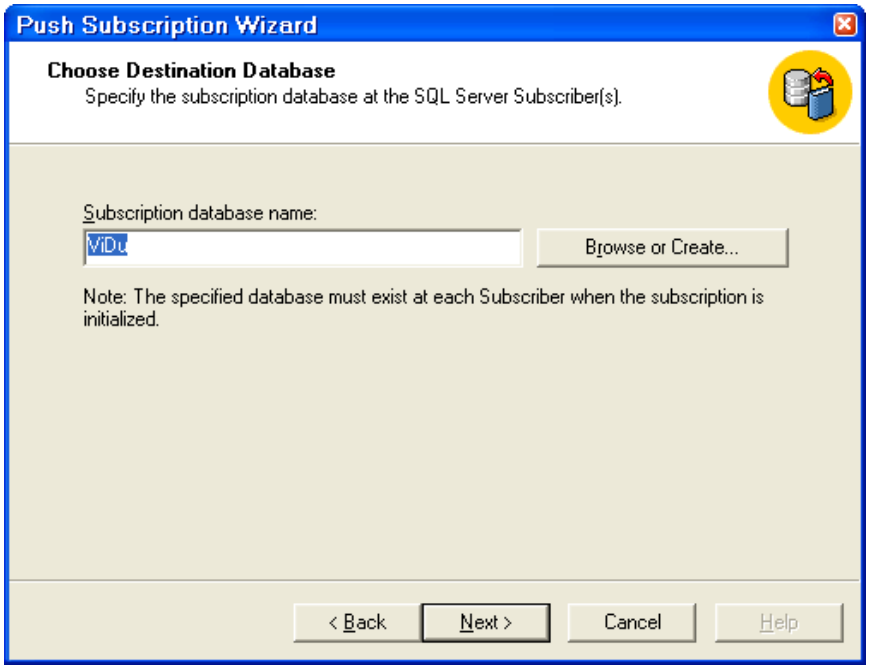
Bước này thực hiện tạo thủ tục đẩy (push) từ Publisher (Distributor trong ví dụ này) đến Subscriber, được thực hiện trên Publisher. Các bước thực hiện như sau:

- Chọn Publication của Publisher -> Nhấn phải chuột -> Push new Subscription...

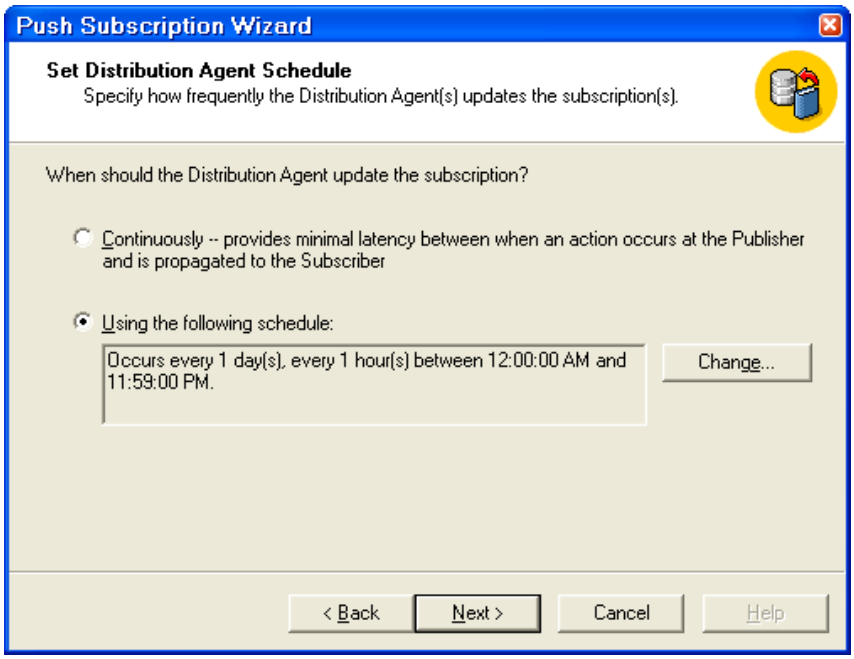
- Chọn Subscriber.



- Chọn CSDL trên Subscriber nếu đã có, nếu chưa có thực hiện chọn chức năng tạo mới.



- Chọn lịch thực hiện đồng bộ dữ liệu.



- Kết thúc. Sau khi thiết lập xong trên Subscriber sẽ có CSDL theo tên đã tạo.

TẠO PULL SUBSCRIPTION.

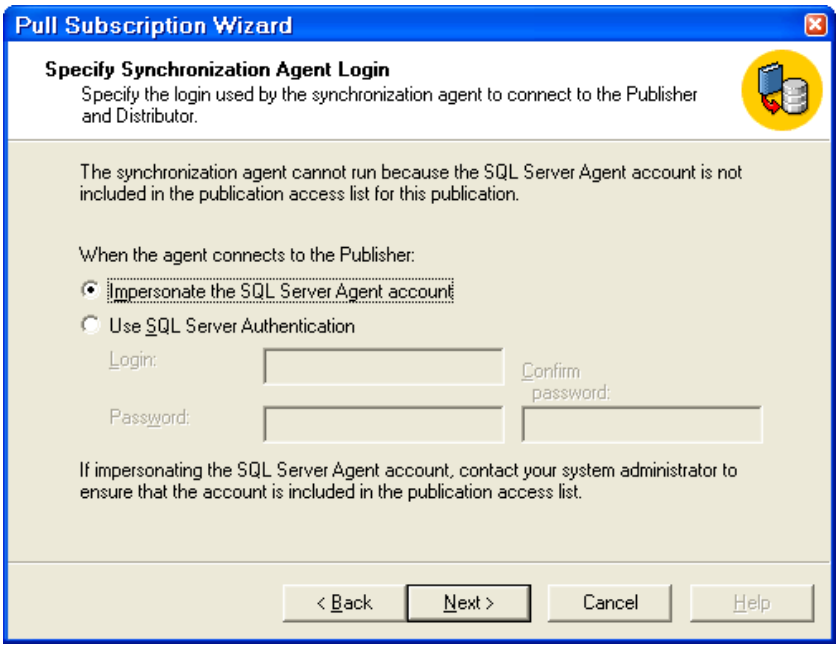
Bước này thực hiện tạo công cụ kéo dữ liệu nhân bản từ Publisher về Subscriber, được thực hiện trên Subscriber.

- Chọn Subscription của Subscriber -> Nhấn phải chuột -> New Pull Supcription...

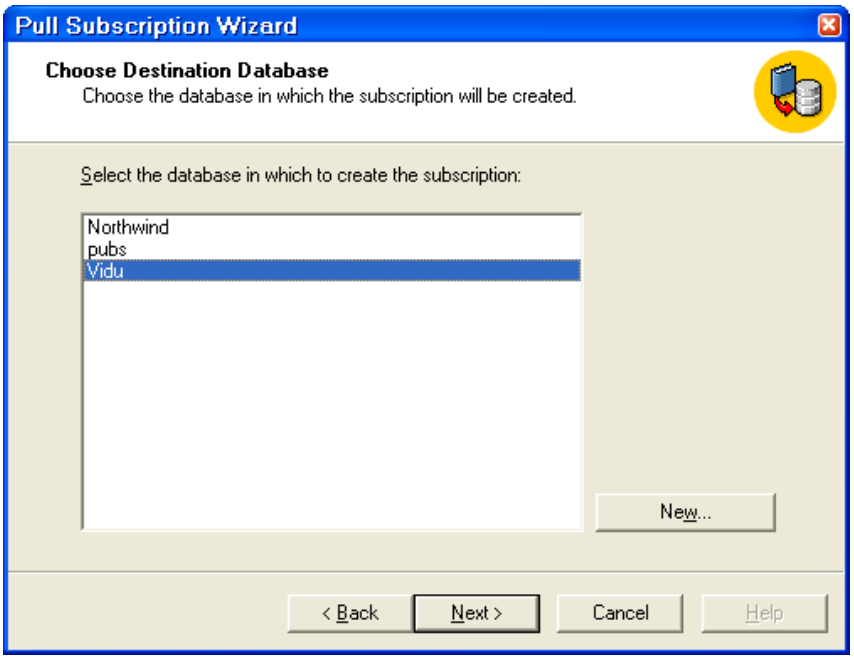
- Thực hiện theo các bước:

+ Chọn Publication.

- Chọn Agent tham gia kết nối Publisher.



- Chọn CSDL đích.



- Thực hiện tiếp các bước và kết thúc.

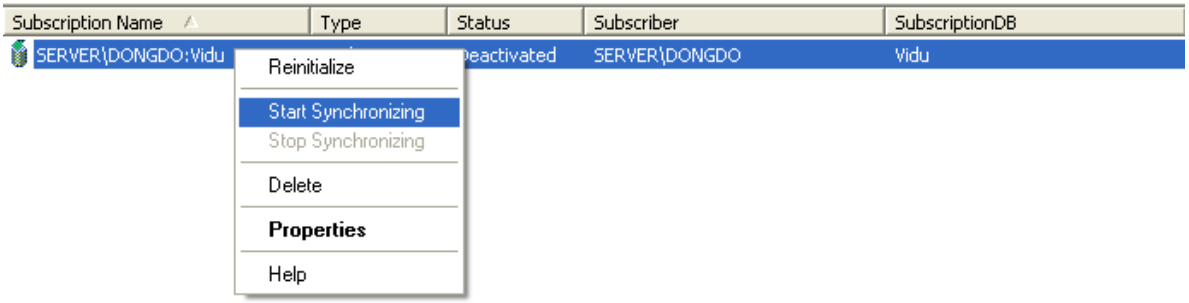
Nếu đã tạo Push Subscription với một CSDL sẽ không được tạo Pull Subscription với CSDL đó.

THỰC HIỆN ĐỒNG BỘ DỮ LIỆU.

Sau khi thiết lập theo các mô hình nhân bản xong, bạn có thể thực hiện đồng bộ dữ liệu bằng cách:

- Thực hiện theo lịch.

- Theo yêu cầu: Chọn Subscription (Push hoặc Pull) -> Nhấn phải chuột -> Start Synchronizing



Sau khi thực hiện xong dữ liệu sẽ được đồng bộ giữa Publisher và Subscriber. Ngoài thực hiện theo công cụ bạn có thể tìm hiểu thực hiện nhân bản theo câu lệnh T-SQL hoặc Stored Procedure.

Phần 2. CÂU LỆNH T-SQL

Trong phần này sẽ giới thiệu cấu trúc, kỹ thuật soạn kịch bản lệnh T-SQL, đối với các hệ quản trị CSDL Foxfro, Access thì câu lệnh thực hiện truy vấn, khai thác CSDL là ngôn ngữ truy vấn SQL (Structure Query Language), các lệnh được thực hiện theo từng câu lệnh mà không thực hiện theo kịch bản hoặc theo tập hợp nhiều câu lệnh với nhau. Đối với hệ quản trị CSDL Oracle thì ngôn ngữ truy vấn dữ liệu là SQL/PL (SQL Plus), còn SQL Server ngôn ngữ có tên Transact-SQL viết tắt là T-SQL.

ĐỊNH NGHĨA DỮ LIỆU (DATA DEFINITION LAGUAGE - DDL).

Phần này sẽ xem xét các lệnh liên quan đến tạo mới, sửa đổi, xóa các đối tượng liên quan đến Table, View và các đối tượng khác.

Tạo kiểu dữ liệu mới.

Tạo kiểu dữ liệu dạng user-defined.

Cú pháp:

sp_addtype [**@typename** =] *type***,**
[**@phystype** =] *system_data_type*
[**,** [**@nulltype** =] '*null_type*']
[**,** [**@owner** =] '*owner_name*']

Ví dụ:

sp_addtype ssn, ‘varchar(11)’, ‘NOT NULL’

Xóa kiểu dữ liệu đã tạo.

Cú pháp:

sp_droptype [**@typename** =] '*type*'

Ví dụ:

Sp_droptype ssn

Tạo ràng buộc (Constraint).

Tạo ràng buộc được thực hiện trong 2 câu lệnh Create Table hoặc Alter Table: Check, Default, Foreign Key, Primary Key, Unique.

Xét một số ví dụ sau:

+ Tạo một Check. trong bảng authors.

ALTER TABLE authors ADD CONSTRAINT chau_id CHECK(au_id LIKE ‘[0-9][0-9][0-9]-[0-9][0-9]- [0-9][0-9] [0-9][0-9]’)

+ Tạo Check trong bảng Publishers.

ALTER TABLE publishers ADD chpub_id CHECK(pub_id IN (‘1389’, ‘0736’, ‘0877’, ‘1622’, ‘1756’) OR pub_id LIKE ‘99[0-9][0-9]’)

+ Tạo ràng buộc Default.

ALTER TABLE authors ADD DEFAULT ‘UNKNOWN’ for au_lname

+ Tạo ràng buộc Foreign Key.

ALTER TABLE titles ADD CONSTRAINT FK_pub_id FOREIGN KEY(pub_id) REFERENCES publishers(pub_id)

+ Tạo ràng buộc Primary Key.

ALTER TABLE authors ADD CONSTRAINT UPKCL_auidind PRIMARY KEY CLUSTERED (au_id)

+ Tạo ràng buộc Unique.

ALTER TABLE stores ADD CONSTRAINT UNC_name_city UNIQUE NONCLUSTERED(store_name, city)

Xóa ràng buộc.

Sử dụng Drop trong các câu lệnh Create Table hoặc Alter Table.

+ Ví dụ xóa Constraint sử dụng câu lệnh Alter Table.

ALTER TABLE authors DROP CONSTRAINT UPKCL_auidind

Hiển thị ràng buộc.

sp_helpconstraint titltes

Tạo bảng.

Để tạo bảng dữ liệu có thể sử dụng 2 câu lệnh Create Table hoặc Select Into.

+ Tạo bảng tạm thời local (là bảng chỉ hiện với phiên hiện thời, tên bảng được bắt đầu bằng một dấu #).

```
CREATE TABLE #MyTempTable (cola INT PRIMARY KEY)
INSERT INTO #MyTempTable VALUES (1)
```

+ Tạo bảng tạm thời global (hiện với tất cả các phiên, tên bảng được bắt đầu bằng 2 dấu #).

```
CREATE TABLE ##MyTempTable (cola INT PRIMARY KEY)
INSERT INTO ##MyTempTable VALUES (1)
```

+ Tạo bảng dữ liệu.

```
/* ************************** jobs table ********* */
CREATE TABLE jobs
(
   job_id  smallint
      IDENTITY(1,1)
      PRIMARY KEY CLUSTERED,
   job_desc        varchar(50)     NOT NULL
      DEFAULT 'New Position - title not formalized
yet',
   min_lvl tinyint NOT NULL
      CHECK (min_lvl >= 10),
   max_lvl tinyint NOT NULL
      CHECK (max_lvl <= 250)
)
```

```
/* ************************ employee table  */
CREATE TABLE employee
(
   emp_id  empid
      CONSTRAINT PK_emp_id PRIMARY KEY NONCLUSTERED
      CONSTRAINT CK_emp_id CHECK (emp_id LIKE
         '[A-Z][A-Z][A-Z][1-9][0-9][0-9][0-9][0-9][FM]'
or
         emp_id LIKE '[A-Z]-[A-Z][1-9][0-9][0-9][0-9]
[0-9][FM]'), fname   varchar(20)     NOT NULL,
   minit   char(1) NULL,
   lname   varchar(30)     NOT NULL,
   job_id  smallint        NOT NULL
      DEFAULT 1
      REFERENCES jobs(job_id),
   job_lvl tinyint
      DEFAULT 10,
   pub_id  char(4) NOT NULL
      DEFAULT ('9952')
      REFERENCES publishers(pub_id),
   hire_date       datetime        NOT NULL
      DEFAULT (getdate())
)

/* ***************** publishers table *** */
CREATE TABLE publishers
(
   pub_id  char(4) NOT NULL
         CONSTRAINT UPKCL_pubind PRIMARY KEY CLUSTERED
         CHECK (pub_id IN ('1389', '0736', '0877',
'1622', '1756')
            OR pub_id LIKE '99[0-9][0-9]'),
   pub_name      varchar(40)     NULL,
   city         varchar(20)     NULL,
   state      char(2) NULL,
   country      varchar(30)     NULL
            DEFAULT('USA')
)
```

Xóa bảng.

Sử dụng lệnh Drop Table.

+ Xóa bảng trong CSDL hiện thời:

Drop Table MyTable

+ Xóa bảng trong CSDL khác.

```
DROP TABLE pubs.dbo.authors2
```

Đổi tên bảng.

Sử dụng thủ tục sp_rename

+ Đổi tên bảng:

Sp_rename titltes, books

Sửa cấu trúc bảng.

Sử dụng lệnh Alter Table.

+ Thêm một cột vào bảng.

```
CREATE TABLE doc_exa ( column_a INT)
GO
ALTER TABLE doc_exa ADD column_b VARCHAR(20) NULL
GO
EXEC sp_help doc_exa
GO
DROP TABLE doc_exa
GO
```

+ Xóa một cột khỏi bảng.

```
CREATE TABLE doc_exb ( column_a INT, column_b
VARCHAR(20) NULL)
GO
ALTER TABLE doc_exb DROP COLUMN column_b
GO
EXEC sp_help doc_exb
GO
DROP TABLE doc_exb
GO
```

Tạo Index.

Sử dụng lệnh Create Index.

+ Tạo Index.

```
SET NOCOUNT OFF
USE pubs
IF EXISTS (SELECT name FROM sysindexes
      WHERE name = 'au_id_ind')
   DROP INDEX authors.au_id_ind
```

```
GO
USE pubs
CREATE UNIQUE CLUSTERED INDEX au_id_ind
   ON authors (au_id)
GO
```

Xem thông tin Index.

Sử dụng thủ tục sp_helpindex

+ Xem Index của bảng authors.

sp_helpindex authors

Xóa Index.

Sử dụng lệnh Drop Index.

+ Xóa Index của bảng authors.

DROP INDEX authors.au_id_ind

Tạo khung nhìn.

Sử dụng lệnh Create View.

+ Tạo View.

```
USE pubs
IF EXISTS (SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.VIEWS
      WHERE TABLE_NAME = 'titles_view')
   DROP VIEW titles_view
GO
CREATE VIEW titles_view
AS
SELECT title, type, price, pubdate
FROM titles
GO
```

Xóa khung nhìn.

Sử dụng lệnh Drop View.

+ Xóa khung nhìn.

```
USE pubs
IF EXISTS (SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.VIEWS
         WHERE TABLE_NAME = 'titles_view')
   DROP VIEW titles_view
```

```
GO
```

Đổi tên khung nhìn.

Sử dụng lệnh thủ tục sp_rename.

+ Đổi tên view.

sp_rename titles_view, view_titles

THAO TÁC VỚI DỮ LIỆU (DATA MANIPULATION LANGUAGE - DML).

Phần này sẽ xem xét các câu lệnh thao tác với dữ liệu như Insert, Select, Delete.

Lệnh Insert - Chèn dữ liệu vào bảng.

Sử dụng câu lệnh Insert.

+ Chèn dữ liệu vào tất cả các cột, theo thứ tự của trong bảng.

```
IF EXISTS(SELECT TABLE_NAME FROM
     INFORMATION_SCHEMA.TABLES
           WHERE TABLE_NAME = 'T1')
        DROP TABLE T1
     GO
CREATE TABLE T1 ( column_1 int, column_2 varchar(30))
INSERT T1 VALUES (1, 'Row #1')
```

+ Chèn dữ liệu vào các cột không theo thứ tự.

```
IF EXISTS(SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES
      WHERE TABLE_NAME = 'T1')
   DROP TABLE T1
GO
CREATE TABLE T1 ( column_1 int, column_2 varchar(30))
INSERT T1 (column_2, column_1) VALUES ('Row #1',1)
```

+ Chèn dữ liệu số giá trị ít hơn số cột.

```
IF EXISTS(SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES
```

```
      WHERE TABLE_NAME = 'T1')
   DROP TABLE T1
GO
CREATE TABLE T1
( column_1 int identity,
  column_2 varchar(30)
    CONSTRAINT default_name DEFAULT ('column default'),
  column_3 int NULL,
  column_4 varchar(40)
)
INSERT INTO T1 (column_4)
   VALUES ('Explicit value')
INSERT INTO T1 (column_2,column_4)
   VALUES ('Explicit value', 'Explicit value')
INSERT INTO T1 (column_2,column_3,column_4)
   VALUES ('Explicit value',-44,'Explicit value')
SELECT *
FROM T1
```

+ Chèn dữ liệu với bảng có cột dữ liệu IDENTITY.

Ví dụ sau sẽ thực hiện chèn dữ liệu vào bảng có cột kiểu IDENITY, cột có kiểu IDENTITY sẽ tự động gán giá trị khi hàng mới được tạo, nên người nhập sẽ không nhập và sửa đổi. Tuy nhiên vó thể sử dụng câu lệnh SET IDENTITY_INSERT để nhập giá trị.

```
IF EXISTS(SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES
      WHERE TABLE_NAME = 'T1')
   DROP TABLE T1
GO
CREATE TABLE T1 ( column_1 int IDENTITY, column_2
varchar(30))
INSERT T1 VALUES ('Row #1')
INSERT T1 (column_2) VALUES ('Row #2')
SET IDENTITY_INSERT T1 ON
INSERT INTO T1 (column_1,column_2)
   VALUES (-99,'Explicit identity value')
SELECT *
FROM T1
```

Lệnh Insert - Chèn dữ liệu vào bảng kết hợp lệnh SELECT.

Câu lệnh này được thực hiện gần tương tự như câu lệnh chèn dữ liệu sử dụng từ khóa VALUES, nhưng giá trị chèn vào được truy vấn từ câu lệnh SELECT. Đối với các cột dữ liệu có kiểu Nchar, Nvarchar hỗ trợ Unicode thì khi chèn dữ liệu trực tiếp phải gán thêm tiền tố N, ví dụ Lname=N'John Smith'.

+ Chèn dữ liệu được truy vấn từ các cột trong lệnh SELECT.

```
USE pubs
INSERT INTO MyBooks
   SELECT *
   FROM titles
   WHERE type = 'mod_cook'
```

+ Chèn dữ liệu được truy vấn từ một số cột.

```
USE pubs
INSERT INTO MyBooks
   SELECT title_id, title, type
   FROM titles
   WHERE type = 'mod_cook'
```

Lệnh Update – Sửa dữ liệu.

Lệnh Update sử dụng sửa dữ liệu trong bảng hoặc View, xem xét cu pháp thông qua các ví dụ sau.

+ Sửa dữ liệu sử dụng lệnh Update sử dụng mệnh đề SET.

```
UPDATE Northwind.dbo.Products
SET UnitPrice = UnitPrice * 1.1
WHERE CategoryID = 2
```

Hoặc gán giá trị trực tiếp:

```
UPDATE authors
    SET authors.au_fname = 'Annie'
    WHERE au_fname = 'Anne'
```

Hoặc gán giá trị NULL cho một cột.

```
UPDATE publishers
```

```
SET pub_name = NULL
```

+ Sửa dữ liệu sử dụng mệnh đề Where xác định hàng được sửa dữ liệu.

```
UPDATE authors
   SET state = 'PC', city = 'Bay City'
      WHERE state = 'CA' AND city = 'Oakland'
```

+ Sửa dữ liệu sử dụng mệnh đề From, sử dụng thông tin từ một bảng khác.

```
UPDATE titles
    SET ytd_sales = t.ytd_sales + s.qty
    FROM titles t, sales s
    WHERE t.title_id = s.title_id
    AND s.ord_date = (SELECT MAX(sales.ord_date) FROM
sales)
```

Hoặc ví dụ giá trị xác định là tổng từ bảng khác.

```
UPDATE titles
   SET ytd_sales =
      (SELECT SUM(qty)
         FROM sales
            WHERE sales.title_id = titles.title_id
            AND sales.ord_date IN (SELECT MAX(ord_date)
FROM sales))
   FROM titles, sales
```

+ Sửa dữ liệu sử dụng mệnh đề Top, xác định số lượng hàng đầu tiên được sửa dữ liệu.

```
UPDATE authors
SET state = 'ZZ'
FROM (SELECT TOP 10 * FROM authors ORDER BY au_lname)
AS t1
WHERE authors.au_id = t1.au_id
```

Lệnh WriteText – Sửa dữ liệu Text, Image.

Lệnh WriteText được sử dụng cập nhật cột có kiểu Text hoặc Image. Dữ liệu kiểu Text và Image thường có kích thước lớn, có thể đến Gyga byte, nên làm việc với kiểu dữ liệu này phải sử dụng con trỏ. Để sử dụng được lệnh này trước hết người quản trị (Administrator) phải đặt thuộc tính select into/bulk copy là true, thực hiện đặt như sau:

```
USE master

EXEC sp_dboption 'pubs', 'select into/bulkcopy', 'TRUE'
```

Với cột dữ liệu kiểu Text, Image ta có thể gán giá trị NULL hoặc sử dụng các lệnh WriteText, UpdateText để gán giá trị, khi sử dụng các lệnh trên, hàng dữ liệu có cột cần chèn đã tồn tại (không đồng thời với câu lệnh Insert). Riêng đối với cột dữ liệu kiểu Text bạn có thể sử dụng lệnh thêm dữ liệu như các cột kiểu chuỗi khác nhưng kích thước của dữ liệu tối đa chỉ được 4096 ký tự.

+ Thực hiện chèn đoạn văn bản vào cột dữ liệu kiểu Text.

```
DECLARE @ptrval binary(16)
SELECT @ptrval = TEXTPTR(pr_info)
FROM pub_info pr, publishers p
WHERE p.pub_id = pr.pub_id
   AND p.pub_name = 'New Moon Books'
WRITETEXT pub_info.pr_info @ptrval 'New Moon Books
(NMB) has just released another top ten publication.
With the latest publication this makes NMB the hottest
new publisher of the year!'

GO
```

Xem ví dụ trên ta thấy, để chèn dữ liệu vào cột Text hoặc Image ta phải sử dụng con trỏ kiểu binary hoặc varbinarry, con trỏ sẽ được xác định vào cột text, image và hàng tương ứng đã có trong bảng dữ liệu, sau đó sử dụng lệnh WriteText để gán giá trị. Trong thực tế khi thực hiện lệnh này ta thường thực hiện thông qua thủ tục lưu trữ của CSDL, giá trị được gán qua biến. Lệnh WriteText thường được sử dụng khi cột dữ liệu đó là NULL hoặc đè toàn bộ dữ liệu đã có (không chèn thêm).

Lệnh UpdateText – Sửa dữ liệu Text, Image.

Lệnh UpdateText có chức năng thực hiện sửa dữ liệu kiểu Text, Image, tuy nhiên UpdateText khác WriteTex, UpdateText có thể sửa., xóa dữ liệu theo từng đoạn hoặc thêm dữ liệu vào phần dữ liệu đã có của cột dữ liệu.

+ Cú pháp chung:

```
UPDATETEXT { table_name.dest_column_name dest_text_ptr }
   { NULL | insert_offset }
   { NULL | delete_length }
   [ WITH LOG ]
   [ inserted_data
      | { table_name.src_column_name src_text_ptr } ]
```

Trong đó:

- Insert_offset: Xác định vị trí theo byte dữ liệu sẽ được đặt vào hoặc bắt đầu xóa.

- Delete_length: Xác định độ dài dữ liệu được xóa tính từ vị trí insert_offset.

Việc chèn, xóa, sửa dữ liệu được điều khiển thông qua các tham số insert_offset, delete_offset, ví dụ muốn sửa dữ liệu, đầu tiên phải xác định vị trí bắt đầu cần sửa dữ liệu (insert_offset) và độ dài dữ liệu cần sửa, bắt đầu từ vị trí cần xóa dữ liệu mới sẽ được chèn vào.

+ Ví dụ sửa nội dung cột kiểu Text.

```
USE pubs
GO
EXEC sp_dboption 'pubs', 'select into/bulkcopy', 'true'
GO
DECLARE @ptrval binary(16)
SELECT @ptrval = TEXTPTR(pr_info)
   FROM pub_info pr, publishers p
      WHERE p.pub_id = pr.pub_id
      AND p.pub_name = 'New Moon Books'
UPDATETEXT pub_info.pr_info @ptrval 88 1 'b'
GO
EXEC sp_dboption 'pubs', 'select into/bulkcopy',
'false'
GO
```

Cursor - Điều khiển con trỏ.

Cursor là kiểu biến xác định con trỏ cho một tập dữ liệu, là kết quả của câu lệnh Select. Cursor được kết hợp cùng lệnh Fetch để xác định vị trí hàng trong tập dữ liệu. Cursor có 2 kiểu Cursor thông thường và Scroll Cursor.

Các thao tác thực hiện với Cursor:

+ Declare: Khai báo.

+ Open: Mở con trỏ để làm việc với tập dữ liệu.

+ Fetch: Dịch chuyển vị trí hàng trong tập dữ liệu.

+ Close: Đóng con trỏ.

+ DeAllocate: Giải phóng con trỏ.

+ Ví dụ sử dụng Curcor, liệt kê danh sách các hàng của bảng Authors.

```
USE pubs
GO
DECLARE authors_cursor CURSOR FOR
SELECT au_lname FROM authors
WHERE au_lname LIKE "B%"
ORDER BY au_lname

OPEN authors_cursor

-- Perform the first fetch.
FETCH NEXT FROM authors_cursor

-- Check @@FETCH_STATUS to see if there are any more
rows to fetch.
WHILE @@FETCH_STATUS = 0
BEGIN
   -- This is executed as long as the previous fetch
succeeds.
   FETCH NEXT FROM authors_cursor
END

CLOSE authors_cursor
DEALLOCATE authors_cursor
GO
```

+ Ví dụ sử dụng Cursor, giá trị cột được đưa vào biến.

```
USE pubs
```

```
GO

-- Declare the variables to store the values returned
by FETCH.
DECLARE @au_lname varchar(40), @au_fname varchar(20)


DECLARE authors_cursor CURSOR FOR
SELECT au_lname, au_fname FROM authors
WHERE au_lname LIKE "B%"
ORDER BY au_lname, au_fname

OPEN authors_cursor

-- Perform the first fetch and store the values in
variables.
-- Note: The variables are in the same order as the
columns
-- in the SELECT statement.

FETCH NEXT FROM authors_cursor
INTO @au_lname, @au_fname

-- Check @@FETCH_STATUS to see if there are any more
rows to fetch.
WHILE @@FETCH_STATUS = 0
BEGIN

   -- Concatenate and display the current values in the
variables.
   PRINT "Author: " + @au_fname + " " +  @au_lname

-- This is executed as long as the previous fetch
succeeds.
   FETCH NEXT FROM authors_cursor
   INTO @au_lname, @au_fname
END

CLOSE authors_cursor
DEALLOCATE authors_cursor
GO
```

+ Ví dụ sử dụng Scroll Cursor, con trỏ cho phép sử dụng các phương thức: LAST, PRIOR, RELATIVE, ABSOLUTE.

```
USE pubs
GO

-- Execute the SELECT statement alone to show the
-- full result set that is used by the cursor.
SELECT au_lname, au_fname FROM authors
ORDER BY au_lname, au_fname

-- Declare the cursor.
DECLARE authors_cursor SCROLL CURSOR FOR
SELECT au_lname, au_fname FROM authors
ORDER BY au_lname, au_fname

OPEN authors_cursor

-- Fetch the last row in the cursor.
FETCH LAST FROM authors_cursor

-- Fetch the row immediately prior to the current row
in the cursor.
FETCH PRIOR FROM authors_cursor

-- Fetch the second row in the cursor.
FETCH ABSOLUTE 2 FROM authors_cursor

-- Fetch the row that is three rows after the current
row.
FETCH RELATIVE 3 FROM authors_cursor

-- Fetch the row that is two rows prior to the current
row.
FETCH RELATIVE -2 FROM authors_cursor

CLOSE authors_cursor
DEALLOCATE authors_cursor
GO
```

Lệnh Delete – Xóa dữ liệu.

Sử dụng lệnh Delete để xóa dữ liệu, kết hợp cùng điều kiện để xóa một hay nhiều hàng dữ liệu trong bảng.

+ Xóa tất cả các hàng của bảng.

```
USE pubs
DELETE authors
```

+ Xóa một tập các hàng.

```
USE pubs
DELETE FROM authors
WHERE au_lname = 'McBadden'
```

+ Xóa một hàng tại vị trí con trỏ.

```
USE pubs
DELETE FROM authors
WHERE CURRENT OF complex_join_cursor
```

Trong ví dụ trên con trỏ đã được mở có tên complex_join_curcor.

+ Xóa các hàng dựa vào lệnh truy vấn khác hoặc liên kết các bảng.

```
/* SQL-92-Standard subquery */
USE pubs
DELETE FROM titleauthor
WHERE title_id IN
   (SELECT title_id
   FROM titles
   WHERE title LIKE '%computers%')

/* Transact-SQL extension */
USE pubs
DELETE titleauthor
FROM titleauthor INNER JOIN titles
   ON titleauthor.title_id = titles.title_id
WHERE titles.title LIKE '%computers%'
```

+ Xóa dữ liệu sử dụng từ khóa Top.

```
DELETE authors
FROM (SELECT TOP 10 * FROM authors) AS t1
WHERE authors.au_id = t1.au_id
```

Lệnh Truncate Table – Xóa dữ liệu toàn bảng.

Tương tự như câu lệnh Delete, lệnh Truncate Table sử dụng xóa dữ liệu toàn bảng, thao tác này giống lệnh Delete khi không có điều kiện Where nhưng lệnh Truncate Table thực hiện nhanh hơn.

```
TRUNCATE TABLE authors
```

Lệnh Go – Nhóm lệnh.

Lệnh Go không tham gia thao tác với CSDL, lệnh Go xác định nhóm các lệnh với nhau, nhóm lệnh được xác định từ vị trí đầu tiên hoặc từ từ lệnh Go trước đó đến lệnh Go tiếp theo. Khi gặp lệnh Go nhóm lệnh sẽ được gửi ngay đến SQL Server để thực hiện.

```
USE pubs
GO
DECLARE @MyMsg VARCHAR(50)
SELECT @MyMsg = 'Hello, World.'
GO -- @MyMsg is not valid after this GO ends the batch.

-- Yields an error because @MyMsg not declared in this
batch.
PRINT @MyMsg
GO

SELECT @@VERSION;
-- Yields an error: Must be EXEC sp_who if not first
statement in
-- batch.
sp_who
GO
```

Contrrol-of-Flow - Điều khiển luồng.

Tương tự như các ngôn ngữ lập trình thiết kế ứng dụng, T-SQL cho phép thiết lập kịch bản câu lệnh, cho phép sử dụng các lệnh điều khiển khối, luồng, vòng lặp, điều kiện, rẽ nhánh,… Sau đây là bảng các lệnh:

Từ khóa	Mô tả

BEGIN…END	Khối lệnh
GOTO	Lệnh nhảy
IF…ELSE	Lệnh điều kiện
RETURN	Thoát
WAITFOR	Chờ thực hiện lệnh
WHILE..BREAK..CONTINUE	Vòng lặp, thoát khỏi vòng lặp, quay lại lặp
CASE	Rẽ nhánh
DECLARE	Khai báo
PRINT	In thông báo
RAISEERROR	Trả lại mã lỗi
EXECUTE (EXEC)	Thực hiện lệnh

TRUY VẤN DỮ LỆU.

Trong trước ta đã xem xét những câu lệnh thao tác với dữ liệu như Insert, Update, Delete, phần này ta sẽ xem xét các câu lệnh khai thác truy vấn dữ liệu như Select, các phép Join,…

Lệnh Use - Chọn Cơ sở dữ liệu.

Sử dung lệnh Use để chọn CSDL trong kịch bản câu lệnh.

```
Use Pubs
```

Select - Truy vấn tất cả các cột từ một bảng.

Lệnh Select được sử dụng truy vấn dữ liệu từ một hoặc nhiều bảng, từ khung nhìn, kết quả đưa lại một tập dữ liệu gồm các hàng, cột.

```
USE Northwind
GO
SELECT *
FROM Shippers
GO
```

Order by - Truy vấn sắp xếp danh sách theo thứ tự.

ASC là sắp xếp tăng, DESC là sắp xếp giảm, khi xác định sắp xếp tăng bạn có thể không cần đặt từ khóa ASC mà hệ thống tự xác định là ASC.

```
USE Northwind
GO
SELECT *
FROM Shippers
ORDER BY CompanyName DESC
GO
```

Truy vấn một số cột, xác định thứ tự các cột.

```
USE Northwind
GO
SELECT OrderID, ProductID, UnitPrice, Quantity,
Discount
FROM [Order Details]
ORDER BY OrderID ASC
GO
```

Đổi tên các cột khi truy vấn.

```
USE Northwind
GO
SELECT OrderID as [Order ID], ProductID as [Product
ID], UnitPrice as [Unit Price], Quantity, Discount
FROM [Order Details]
ORDER BY OrderID ASC
GO
```

Lệnh Case - Phân lớp dữ liệu.

Case là câu lệnh rẽ nhánh, thường được sử dụng phân lớp dữ liệu trong câu lệnh Select.

Ví dụ sử dụng lệnh Case đơn giản:

```
USE pubs
```

```
GO
SELECT   Category =
      CASE type
         WHEN 'popular_comp' THEN 'Popular Computing'
         WHEN 'mod_cook' THEN 'Modern Cooking'
         WHEN 'business' THEN 'Business'
         WHEN 'psychology' THEN 'Psychology'
         WHEN 'trad_cook' THEN 'Traditional Cooking'
         ELSE 'Not yet categorized'
      END,
   CAST(title AS varchar(25)) AS 'Shortened Title',
   price AS Price
FROM titles
WHERE price IS NOT NULL
ORDER BY type, price
COMPUTE AVG(price) BY type
GO
```

Ví dụ sử dụng lệnh Case tìm kiếm:

```
USE pubs
GO
SELECT    'Price Category' =
      CASE
         WHEN price IS NULL THEN 'Not yet priced'
         WHEN price < 10 THEN 'Very Reasonable Title'
         WHEN price >= 10 and price < 20 THEN 'Coffee
Table Title'
         ELSE 'Expensive book!'
      END,
   CAST(title AS varchar(20)) AS 'Shortened Title'
FROM titles
ORDER BY price
GO
```

Kết quả thực hiện như sau:

```
Price Category        Shortened Title
--------------------- --------------------
```

```
Not yet priced        Net Etiquette
Not yet priced        The Psychology of Co
Very Reasonable Title The Gourmet Microwav
Very Reasonable Title You Can Combat Compu
Very Reasonable Title Life Without Fear
Very Reasonable Title Emotional Security:
Coffee Table Title    Is Anger the Enemy?
Coffee Table Title    Cooking with Compute
Coffee Table Title    Fifty Years in Bucki
Coffee Table Title    Sushi, Anyone?
Coffee Table Title    Prolonged Data Depri
Coffee Table Title    Silicon Valley Gastr
Coffee Table Title    Straight Talk About
Coffee Table Title    The Busy Executive's
Expensive book!       Secrets of Silicon V
Expensive book!       Onions, Leeks, and G
Expensive book!       Computer Phobic And
Expensive book!       But Is It User Frien

(18 row(s) affected)
```

Đặt tên cho cột.

Sử dụng dấu phẩy xác định tên cột, tối đa là 30 ký tự.

```
SELECT ‘sum’= SUM(ytd_sales) FROM titles
```

Khi cần thể hiện dấu phẩy trên giá trị hoặc tên cột ta cần sử dụng 2 dấu liền nhau. Ví dụ `‘I don’’t understand.’`

Chuỗi ký tự trong kết quả truy vấn.

Sử dụng dấu phẩy trong chuỗi ký tự.

```
SELECT ‘The publisher’’s name is’, publisher=pub_name
FROM publishers
```

Các giá trị tính toán được.

Đối các kiểu dữ liệu tính toán được sử dụng các phép toán +, -, *, /, %.

```
SELECT title_id, ytd_sales*2 FROM titles
```

Truy vấn kiểu dữ liệu Text, Image.

Để truy vấn dữ liệu Text, Image có thể sử dụng 2 lệnh Select hoặc ReadText. Khi sử dụng lệnh Select để truy vấn kiểu dữ liệu này thì chỉ truy vấn được dữ liệu có độ dài xác định trước bằng câu lệnh SET TEXTSIZE.

```
SET TEXTSIZE 25
SELECT pub_id, pr_info FROM pub_info
```

Ngầm định kích thước sử dụng cho truy vấn là 4096 (4K).

Từ khóa Distinct – Truy vấn các hàng khác nhau theo cột.

Để truy vấn các hàng dữ liệu khác nhau theo cột ta sử dụng từ khóa Distinct.

```
USE pubs
SELECT DISTINCT au_id
FROM titleauthor
```

Xác định bảng trong mệnh đề From.

```
USE pubs
SELECT p.pub_id, p.pub_name
FROM publishers p
```

Mệnh đề Where.

Mệnh đề Where xác định điều kiện các hàng được truy vấn, biểu thức trong mệnh đề Where xác định theo biểu thức logic. Các phép toán, câu lệnh xác định gồm:

- Các phép toán so sánh: =, <>, <, >, !<, !>.

- Từ khóa xác định phạm vi: Between, Not Between.

- Danh sách: In, Not In.

- Theo mẫu đinh dạng: Like, Not Like.

- Giá trị NULL: Is Null, Is Not Null.

- Các phép toán logic: And, Or.

+ *Từ khóa Between*:

```
SELECT UnitsInStock, ProductID, ProductName
FROM Northwind.dbo.Products
WHERE UnitsInStock BETWEEN 15 AND 25
ORDER BY UnitsInStock
```

+ *Từ khóa Not Between*.

```
SELECT UnitsInStock, ProductID, ProductName
FROM Northwind.dbo.Products
WHERE UnitsInStock NOT BETWEEN 15 AND 25
ORDER BY UnitsInStock
```

+ *Từ khóa In, Not In*.

```
USE pubs
SELECT au_lname, state
FROM authors
WHERE state IN ('CA', 'IN', 'MD')
```

```
USE pubs
SELECT au_lname, au_fname
FROM authors
WHERE au_id IN
   (SELECT au_id
   FROM titleauthor
```

```
   WHERE royaltyper < 50)

USE pubs
SELECT au_lname, au_fname
FROM authors
WHERE au_id NOT IN
   (SELECT au_id
   FROM titleauthor
   WHERE royaltyper < 50)
```

+ Từ khóa Like.

Từ khóa Like được sử dụng tương tự như phép so sánh, phép Like được thực hiện cho dữ liệu kiểu chuỗi, phép Like được xem là phép so sánh theo định dạng của chuỗi, việc định dạng xác định theo một số từ khóa sau:

%	Xác định bất kỳ chuỗi ký tự nào hoặc không có ký tự nào tại vị trí.
_	Một ký tự bất kỳ nào đó.
[]	Một ký tự nào đó nằm trong phạm vi, ví dụ [a-f].
[^]	Xác định một ký tự không thuộc phạm vi nào đó, ví dụ [^a-f].

Ví dụ sử dụng từ khóa Like với %:

```
USE pubs
GO
SELECT phone
FROM authors
WHERE phone LIKE '415%'
ORDER by au_lname
GO
```

Ví dụ từ khóa Not Like với %:

```
USE pubs
GO
```

```
SELECT phone
FROM authors
WHERE phone NOT LIKE '415%'
ORDER BY au_lname
GO
```

Ví dụ sử dụng từ khóa Like với mệnh đề Escape: Escape được sử dụng loại bỏ một ký tự hoặc chuỗi khỏi phép so sánh.

```
USE pubs
GO
IF EXISTS(SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES
      WHERE TABLE_NAME = 'mytbl2')
   DROP TABLE mytbl2
GO
USE pubs
GO
CREATE TABLE mytbl2
(
 c1 sysname
)
GO
INSERT mytbl2 VALUES ('Discount is 10-15% off')
INSERT mytbl2 VALUES ('Discount is .10-.15 off')
GO
SELECT c1
FROM mytbl2
WHERE c1 LIKE '%10-15!% off%' ESCAPE '!'
GO
```

Ví dụ sử dụng từ khóa Like với []:

```
USE pubs
GO
SELECT au_lname, au_fname, phone
FROM authors
WHERE au_lname LIKE '[CK]ars[eo]n'
ORDER BY au_lname ASC, au_fname ASC
```

```
GO
```

+ Giá trị NULL.

Giá trị NULL được nhập bằng cách đặt ngầm định hoặc gán theo câu lệnh. Để tìm giá trị NULL trong bảng sử dụng từ khóa Is Null hoặc Is Not Null.

```
SELECT title_id, type, advance
FROM pubs.dbo.titles
WHERE advance IS NULL
```

TẠO BẢNG BẰNG LỆNH SELECT INTO.

Lệnh Select Into truy vấn dữ liệu, dữ liệu được đưa vào một bảng mới. Nếu thuộc tính select into/bulkcopy được đặt có thể tạo bảng cố định, nếu thuộc tính không được đặt ta có thể tạo bảng tạm thời.

```
SELECT Shippers.*, Link.Address, Link.City,
                   Link.Region, Link.PostalCode
INTO NewShippers
FROM Shippers
     JOIN LinkServer.DB.dbo.Shippers AS Link
       ON (Shippers.ShipperID = Link.ShipperID)
```

LỆNH COMPUTE BY.

Khi thực hiện với các hàm tính toán SUM, AVG, MIN, MAX, COUNT thường được sử dụng với các mệnh đề GROUP BY, COMPUTE BY (không áp dụng các hàm tính toán với dữ liệu kiểu Text, Image).

+ Sử dụng Group By: Từ khóa Group By được sử dụng nhóm theo cột, có thể kết hợp các hàm tính toán.

```
USE Northwind
SELECT OrdD.ProductID AS ProdID,
       SUM(OrdD.Quantity) AS AmountSold
FROM [Order Details] AS OrdD JOIN Products as Prd
```

```
     ON OrdD.ProductID = Prd.ProductID
     AND Prd.CategoryID = 2
GROUP BY OrdD.ProductID
```

+ *Sử dụng mệnh đề Compute*: Tính toán toàn bộ giá trị.

```
USE pubs
SELECT type, price, advance
FROM titles
ORDER BY type
COMPUTE SUM(price), SUM(advance)
```

+ *Sử dụng mệnh đề Compute By*: Tính toán theo nhóm (tương tự Group By).

```
USE pubs
SELECT type, price, advance
FROM titles
ORDER BY type
COMPUTE SUM(price), SUM(advance) BY type
```

TOÁN TỬ UNION.

Toán tử Union thực hiện hợp 2 tập với nhau, phép toán này thực hiện chỉ lấy đại diện khi có hai hàng của hai tập trùng nhau.

Giả sử có 2 bảng dữ liệu như sau:

Table1			Table2	
ColumnA	ColumnB		ColumnC	ColumnD
char(4)	int		char(4)	int
-------	---		-------	---
abc	1		ghi	3
def	2		jkl	4
ghi	3		mno	5

Thực hiện toán tử Union:

```
SELECT * FROM Table1
```

```
UNION
SELECT * FROM Table2
```

Kết quả thực hiện:

```
ColumnA  ColumnB
-------  --------
abc      1
def      2
ghi      3
jkl      4
mno      5
```

Khi sử dụng từ khóa ALL toàn bộ các hàng của hai tập dữ liệu sẽ được hợp lại, không loại bỏ những hàng trùng nhau.

```
SELECT * FROM TableA
UNION ALL
(   SELECT * FROM TableB
   UNION
   SELECT * FROM TableC
)
GO
```

Sử dụng toán tử Union với Select Into:

```
USE Northwind
IF EXISTS(SELECT TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES
      WHERE TABLE_NAME = 'CustomerResults')
   DROP TABLE CustomerResults
GO
USE Northwind
SELECT ContactName, CompanyName, City, Phone INTO
CustomerResults
FROM Customers
WHERE Country IN ('USA', 'Canada')
UNION
```

```
SELECT ContactName, CompanyName, City, Phone
FROM SouthAmericanCustomers
ORDER BY CompanyName, ContactName ASC
GO
```

TRUY VẤN DỮ LIỆU TỪ NHIỀU BẢNG.

Truy vấn dữ liệu từ nhiều bảng được xác định theo quan hệ giữa các cột của các bảng với nhau. Có thể truy vấn thông qua điều kiện liên kết trong mệnh đề Where hoặc từ khóa Join.

Theo điều kiện liên kết.

Sử dụng điều kiện liên kết theo cột giữa các bảng, thông tin cần truy vấn được đặt ở nhiều bảng khác nhau, để truy vấn được các thông tin như trên phải xác định điều kiện liên kết giữa các bảng.

+ Liên kết bằng nhau.

```
SELECT P.ProductID,
       S.SupplierID,
       S.CompanyName
FROM Suppliers AS S, Products AS P
WHERE S.SupplierID = P.SupplierID
  AND P.UnitPrice > $10
  AND S.CompanyName LIKE N'F%'
```

Đối với câu lệnh truy vấn theo điều kiện liên kết nói trên, các hàng chứa giá trị Null của cột tham gia liên kết sẽ không được liệt kê, câu lệnh này tương đương với lệnh Inner Join (sẽ xem trong phần sau).

+ Liên kết không bằng nhau.

Liên kết dạng này sử dụng các phép toán so sánh >, >=, <, <=, <>, !>, !<

```
USE pubs
SELECT p.pub_name, p.state, a.au_lname, a.au_fname,
a.state
```

```
FROM publishers p, authors a
   WHERE a.state > p.state and
p.pub_name = 'New Moon Books'
ORDER BY au_lname ASC, au_fname ASC
```

+ Tự liên kết bằng nhau.

Tự liên kết trong một bảng, câu lệnh dạng này thường được sử dụng trong việc xác định những cặp giá trị nào các cột trong bảng có quan hệ với nhau theo liên kết.

```
USE pubs
SELECT au1.au_fname, au1.au_lname, au2.au_fname,
au2.au_lname
FROM authors au1, authors au2
   WHERE au1.zip = au2.zip and au1.city = 'Oakland'
ORDER BY au1.au_fname ASC, au1.au_lname ASC
```

+ Tự liên kết không bằng nhau.

```
USE pubs
SELECT au1.au_fname, au1.au_lname, au2.au_fname,
au2.au_lname
FROM authors au1, authors au2
WHERE  au1.zip = au2.zip
   AND au1.city = 'Oakland'
   AND au1.state = 'CA'
   AND au1.au_id < au2.au_id
ORDER BY au1.au_lname ASC, au1.au_fname ASC
```

+ Truy vấn dữ liệu từ nhiều hơn 2 bảng dữ liệu.

Truy vấn dạng này thực hiện điều kiện liên kết theo từng cặp các bảng với nhau.

```
USE pubs
SELECT a.au_lname, a.au_fname, t.title
FROM authors a, titleauthor ta, titles t
```

```
WHERE a.au_id = ta.au_id
 AND ta.title_id = t.title_id
 AND t.type = 'trad_cook'
 ORDER BY t.title ASC
```

+ Liên kết ngoài trái.

Như những điều kiện liên kết nói trên, những hàng có cột là Null sẽ không được đưa ra tập kết quả, câu lệnh liên kết ngoài sẽ đưa ra những hàng chứa giá trị Null nói trên. Xác định liên kết ngoài bằng toán tử *.

```
USE pubs
SELECT a.au_fname, a.au_lname, p.pub_name
FROM authors a, publishers p
WHERE a.city *= p.city
ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC
```

Bảng authors sẽ đưa ra cả những hàng có cột có cột city là Null, khi đó chưa có pub_name, kết quả như sau:

```
au_fname            au_lname                 pub_name
----------         ----------------          --------
Reginald            Blotchet-Halls           NULL
Michel              DeFrance                 NULL
Innes               del Castillo             NULL
Ann                 Dull                     NULL
Marjorie            Green                    NULL
Morningstar         Greene                   NULL
Burt                Gringlesby               NULL
Sheryl              Hunter                   NULL
Livia               Karsen                   NULL
Charlene            Locksley                 NULL
Stearns             MacFeather               NULL
Heather             McBadden                 NULL
Michael             O'Leary                  NULL
Sylvia              Panteley                 NULL
Albert              Ringer                   NULL
Anne                Ringer                   NULL
Meander             Smith                    NULL
Dean                Straight                 NULL
Dirk                Stringer                 NULL
Johnson             White                    NULL
Akiko               Yokomoto                 NULL
Abraham             Bennet           Algodata Infosystems
```

```
Cheryl               Carson                Algodata Infosystems

(23 row(s) affected)
```

+ Liên kết ngoài phải.

```
USE pubs
SELECT a.au_fname, a.au_lname, p.pub_name
FROM authors AS a, publishers AS p
WHERE a.city = p.city
ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC
```

Kết quả thực hiện:

```
au_fname             au_lname                 pub_name
-------------------- ----------------------  --------------------
Abraham              Bennet                   Algodata
Infosystems
Cheryl               Carson                   Algodata
Infosystems
NULL                 NULL                     Binnet & Hardley
NULL                 NULL                     Five Lakes
Publishing
NULL                 NULL                     GGG&G
NULL                 NULL                     Lucerne Publishing
NULL                 NULL                     New Moon Books
NULL                 NULL                     Ramona Publishers
NULL                 NULL                     Scootney Books

(9 row(s) affected)
```

Lệnh Join – Truy vấn từ nhiều bảng.

Phần trên ta đã xem xét kỹ thuật truy vấn dữ liệu từ nhiều bảng sử dụng điều kiện liên kết, tương tự như các phép toán so sánh, *=, =* SQL Server cung cấp câu lệnh Join thay thế các phép toán nói trên.

+ Inner Join – Liên kết trong.

Thay vì xác định điều kiện liên kết trong mệnh đề Where thì ở đây ta chỉ cần xác định liên kết trong mệnh đề From.

Liên kết bằng:

```
USE pubs
SELECT *
FROM authors AS a INNER JOIN publishers AS p
   ON a.city = p.city
ORDER BY a.au_lname DESC
```

Liên kết không bằng:

```
USE pubs
SELECT p.pub_name, p.state, a.au_lname, a.au_fname,
a.state
FROM publishers p INNER JOIN authors a
   ON a.state > p.state
WHERE p.pub_name = 'New Moon Books'
ORDER BY au_lname ASC, au_fname ASC
```

+ Tự liên kết trong bảng.

Tự liên kết bằng:

```
USE pubs
SELECT au1.au_fname, au1.au_lname, au2.au_fname,
au2.au_lname
FROM authors au1 INNER JOIN authors au2
   ON au1.zip = au2.zip
WHERE au1.city = 'Oakland'
ORDER BY au1.au_fname ASC, au1.au_lname ASC
```

Tự liên kết không bằng:

```
USE pubs
SELECT au1.au_fname, au1.au_lname, au2.au_fname,
au2.au_lname
FROM authors au1 INNER JOIN authors au2
   ON  au1.zip = au2.zip
WHERE au1.city = 'Oakland'
```

```
   AND au1.state = 'CA'
   AND au1.au_id < au2.au_id
ORDER BY au1.au_lname ASC, au1.au_fname ASC
```

+ Liên kết nhiều hơn 2 bảng.

```
USE pubs
SELECT a.au_lname, a.au_fname, t.title
FROM authors a INNER JOIN titleauthor ta
   ON a.au_id = ta.au_id JOIN titles t
   ON ta.title_id = t.title_id
WHERE t.type = 'trad_cook'
ORDER BY t.title ASC
```

+ Liên kết ngoài trái - LEFT OUTER JOIN.

Liên kết ngoài trái tương tự như phép toán *=.

```
USE pubs
SELECT a.au_fname, a.au_lname, p.pub_name
FROM authors a LEFT OUTER JOIN publishers p
   ON a.city = p.city
ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC
```

+ Liên kết ngoài phảii – RIGHT OUTER JOIN.

Liên kết ngoài phải tương tự như phép toán =*.

```
USE pubs
SELECT a.au_fname, a.au_lname, p.pub_name
FROM authors a RIGHT OUTER JOIN publishers p
   ON a.city = p.city
ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC
```

+ Liên kết ngoài 2 phía – FULL OUTER JOIN.

Là phép liên kết trái hoặc phải.

```
USE pubs
SELECT a.au_fname, a.au_lname, p.pub_name
FROM authors a FULL OUTER JOIN publishers p
   ON a.city = p.city
ORDER BY p.pub_name ASC, a.au_lname ASC, a.au_fname ASC
```

Kết quả như sau:

```
au_fname             au_lname                 pub_name
---------------    ------------------------  -----------------
Reginald             Blotchet-Halls               NULL
Michel               DeFrance                     NULL
Innes                del Castillo                 NULL
Ann                  Dull                         NULL
Marjorie             Green                        NULL
Morningstar          Greene                       NULL
Burt                 Gringlesby                   NULL
Sheryl               Hunter                       NULL
Livia                Karsen                       NULL
Charlene             Locksley                     NULL
Stearns              MacFeather                   NULL
Heather              McBadden                     NULL
Michael              O'Leary                      NULL
Sylvia               Panteley                     NULL
Albert               Ringer                       NULL
Anne                 Ringer                       NULL
Meander              Smith                        NULL
Dean                 Straight                     NULL
Dirk                 Stringer                     NULL
Johnson              White                        NULL
Akiko                Yokomoto                     NULL
Abraham              Bennet                  Algodata Infosystems
Cheryl               Carson                  Algodata Infosystems
NULL                 NULL                    Binnet & Hardley
NULL                 NULL                    Five Lakes Publishing
NULL                 NULL                    GGG&G
NULL                 NULL                    Lucerne Publishing
NULL                 NULL                    New Moon Books
NULL                 NULL                    Ramona Publishers
NULL                 NULL                    Scootney Books

(30 row(s) affected)
```

+ Giá trị Null và phép Join.

Giá trị Null không xác định trong phép so sánh của mệnh đề Where (chỉ sử dụng với các phép so sánh Is Null hoặc Is Not Null), trong phép Join ta có thể xác định giống nhau giữa 2 giá trị Null. Xét ví dụ sau:

Giả sử có 2 bảng dữ liệu có giá trị như sau:

```
table1                          table2
a           b                   c           d
-------     ------              -------     ------
      1        one               NULL        two
   NULL      three                  4       four
      4      join4
```

Thực hiện phép Join như sau:

```
SELECT *
FROM table1 t1 JOIN table2 t2
   ON t1.a = t2.c
ORDER BY t1.a
```

Kết quả thực hiện:

```
a           b      c           d
----------- ------ ----------- ------
4           join4  4           four

(1 row(s) affected)
```

TRUY VẤN TỔNG HỢP.

Việc sử dụng các hàm tính toán như SUM, AVG,… thường được thực hiện theo các mệnh đề WHERE, GROUP BY, HAVING. Khi xác định điều kiện

có sử dụng các hàm tính toán thì phải sử dụng mệnh đề HAVING mà không được sử dụng trong mệnh đề WHERE.

Các hàm tính toán có thể tóm tắt như sau:

SUM([ALL | DISTINCT])	Tính tổng tất cả hoặc những hàng khác nhau.
AVG([ALL | DISTINCT])	Tính trung bình tất cả hoặc những hàng khác nhau.
COUNT([ALL | DISTINCT])	Đếm số hàng tất cả hoặc những hàng khác nhau.
COUNT(*)	Đếm các hàng được lựa chọn.
MAX()	Tính giá trị lớn nhất.
MIN()	Tính giá trị nhỏ nhất.

Các hàm SUM, AVG chỉ làm việc với dữ liệu dạng số, các hàm SUM, AVG, COUNT, MAX, MIN bỏ qua giá trị Null, hàm COUNT(*) đếm cả hàng có giá trị Null.

Sử dụng hàm tính toán.

+ Tính tổng toàn bộ.

```
USE pubs
SELECT SUM(ytd_sales)
FROM titles
```

+ Tính tổng, trung bình có điều kiện.

```
USE pubs
SELECT AVG(advance), SUM(ytd_sales)
FROM titles
WHERE type = 'business'
```

Mệnh đề Group By.

Group by được thực hiện nhóm các hàng theo giá trị cột xác định, các hàm tính toán sẽ được thực hiện theo nhóm nói trên.

```
USE Northwind
SELECT OrdD.ProductID AS ProdID,
       SUM(OrdD.Quantity) AS AmountSold
FROM [Order Details] AS OrdD JOIN Products as Prd
     ON OrdD.ProductID = Prd.ProductID
     AND Prd.CategoryID = 2
GROUP BY OrdD.ProductID
```

Kết quả thực hiện như sau:

```
ProdID      AmountSold
----------- -----------
3           328
4           453
5           298
6           301
8           372
15          122
44          601
61          603
63          445
65          745
66          239
77          791

(12 row(s) affected)
```

Mệnh đề Having.

Having được sử dụng cùng với các hàm tính toán xác định điều kiện lọc các hàng, thường được kết hợp cùng mệnh đề Group By để thực hiện các hàm tính toán theo nhóm.

+ Having với hàm SUM.

```
USE pubs
SELECT pub_id, total = SUM(ytd_sales)
FROM titles
GROUP BY pub_id
```

```
HAVING SUM(ytd_sales) > 40000
```

+ Having với hàm Count.

```
USE pubs
SELECT pub_id, total = SUM(ytd_sales)
FROM titles
GROUP BY pub_id
HAVING COUNT(*) > 5
```

+ Having với mệnh đề Where.

```
SELECT pub_id, SUM(advance) AS AmountAdvanced,
       AVG(price) AS AveragePrice
FROM pubs.dbo.titles
WHERE pub_id > '0800'
  AND price >= $5
GROUP BY pub_id
HAVING SUM(advance) > $15000
   AND AVG(price) < $20
ORDER BY pub_id DESC
```

+ Having thay cho mệnh đề Where.

```
SELECT titles.pub_id, AVG(titles.price)
FROM titles INNER JOIN publishers
   ON titles.pub_id = publishers.pub_id
GROUP BY titles.pub_id
HAVING publishers.state = 'CA'
```

TRUY VẤN LỒNG NHAU.

Phần này sẽ xem xét các câu lệnh truy vấn lồng nhau, trong câu lệnh truy vấn Select có câu lệnh truy vấn Select khác trong điều kiện xác định của lệnh Select ngoài. Thông thường các câu lệnh dạng này đi cùng các từ lhóa IN, NOT IN, EXITST, NOT EXIST, ANY, ALL.

Truy vấn lồng nhau với phép bằng.

```
USE pubs
SELECT title, price
FROM titles
WHERE price =
   (SELECT price
   FROM titles
   WHERE title = 'Straight Talk About Computers')
```

Đầu tiên câu lệnh sẽ xác định hàng trong lệnh Select trong, lệnh truy vấn này phải đưa ra kết quả duy nhất.

Truy vấn với từ khóa IN.

Kiểm tra nằm trong tập các giá trị truy vấn được.

```
USE pubs
SELECT distinct pub_name
FROM publishers
WHERE pub_id IN
   (SELECT pub_id
   FROM titles
   WHERE type = 'business')
```

Hằng số nằm trong khoảng:

```
USE pubs
SELECT DISTINCT au_lname, au_fname
FROM authors
WHERE 100 IN
   (SELECT royaltyper
   FROM titleauthor
   WHERE titleauthor.au_id = authors.au_id)
```

Truy vấn với từ khóa Exist.

Kiểm tra tồn tại hàng dữ liệu truy vấn được.

```
USE pubs
SELECT DISTINCT pub_name
FROM publishers
WHERE EXISTS
   (SELECT *
   FROM titles
   WHERE pub_id = publishers.pub_id
   AND type = 'business')
```

Truy vấn với hàm All.

Kiểm tra với tất cả các hàng.

```
USE pubs
SELECT t1.type
FROM titles t1
GROUP BY t1.type
HAVING MAX(t1.advance) >= ALL
   (SELECT 2 * AVG(t2.advance)
   FROM titles t2
   WHERE t1.type = t2.type)
```

Truy vấn với hàm Any.

Kiểm tra thỏa mãn với bất kỳ hàng nào.

```
USE pubs
SELECT title
FROM titles
WHERE advance > ANY
   (SELECT advance
   FROM publishers INNER JOIN titles
   ON titles.pub_id = publishers.pub_id
      AND pub_name = 'Algodata Infosystems')
```

Truy vấn với hàm Some.

Kiểm tra với ít nhất một hàng.

```
USE pubs
SELECT t1.type
FROM titles t1
GROUP BY t1.type
HAVING MAX(t1.advance) >= SOME
   (SELECT 2 * AVG(t2.advance)
   FROM titles t2
   WHERE t1.type = t2.type)
```

Nhiều lệnh Select lồng nhau.

```
USE pubs
SELECT au_lname, au_fname
FROM authors
WHERE au_id IN
   (SELECT au_id
   FROM titleauthor
   WHERE title_id IN
      (SELECT title_id
      FROM titles
      WHERE type = 'popular_comp'))
```

UPDATE, DELETE, INSERT VỚI LỆNH TRUY VẤN LỒNG NHAU.

Việc thực hiện các lệnh thao tác với dữ liệu có thể kết hợp điều kiện truy vấn lồng nhau để xác định phạm vi dữ liệu được thao tác.

Kết hợp với lenệh Select.

```
UPDATE titles
SET price = price * 2
WHERE pub_id IN
   (SELECT pub_id
   FROM publishers
```

```
   WHERE pub_name = 'New Moon Books')
```

Kết hợp với lệnh Join.

```
UPDATE titles
SET price = price * 2
FROM titles INNER JOIN publishers ON titles.pub_id =
publishers.pub_id
   AND pub_name = 'New Moon Books'
```

Xóa dữ liệu kết hợp với lệnh Select.

```
DELETE sales
WHERE title_id IN
   (SELECT title_id
   FROM titles
   WHERE type = 'business')
```

Xóa dữ liệu với phép Join.

```
DELETE sales
FROM sales INNER JOIN titles ON sales.title_id =
titles.title_id
   AND type = 'business'
```

LỆNH READTEXT – ĐỌC TEXT, IMAGE.

Lệnh ReadText được thực hiện đọc dữ liệu kiểu Text, Image và chuyển vào một biến.

```
USE pubs
GO
DECLARE @ptrval varbinary(16)
SELECT @ptrval = TEXTPTR(pr_info)
   FROM pub_info pr INNER JOIN publishers p
```

```
      ON pr.pub_id = p.pub_id
      AND p.pub_name = 'New Moon Books'
READTEXT pub_info.pr_info @ptrval 1 25
GO
```

Ví dụ trên thực hiện đọc dữ liệu từ cột pr_info bắt đầu từ vị trí 1, độ dài 25 byte.

THAO TÁC DỮ LIỆU NGOÀI.

Nội dung phần này sẽ giới thiệu câu lệnh, kỹ thuật truy vấn dữ liệu của hệ quản trị CSDL khác hoặc Instance khác.

Lệnh OpenRowSet.

Lệnh OpenRowSet sử dụng truy nhập dữ liệu xa với nguồn dữ liệu là OLE DB, kết nối kiểu này có thể thực hiện các lệnh Insert, Update, Delete, Select với bảng dữ liệu. Quyền thực hiện trong câu lệnh thực hiện theo user kết nối trong câu lệnh.

+ OPENROWSET với lệnh SELECT và Microsoft OLE DB Provider for SQL Server.

```
USE pubs
GO
SELECT a.*
FROM OPENROWSET('SQLOLEDB','seattle1';'sa';'MyPass',
   'SELECT * FROM pubs.dbo.authors ORDER BY au_lname,
au_fname') AS a
GO
```

Ví dụ trên thực hiện kết nối đến Instance có tên seattle1, user có tên sa, mật khẩu MyPass.

+ OPENROWSET với OLE DB Provider for ODBC.

```
USE pubs
GO
SELECT a.*
FROM OPENROWSET('MSDASQL',
   'DRIVER={SQL
Server};SERVER=seattle1;UID=sa;PWD=MyPass',
```

```
   pubs.dbo.authors) AS a
ORDER BY a.au_lname, a.au_fname
GO
```

+ *Microsoft OLE DB Provider for Jet*. Lệnh dạng này được này được thực hiện kết nối đến Access.

```
USE pubs
GO
SELECT a.*
FROM OPENROWSET('Microsoft.Jet.OLEDB.4.0',
   'c:\MSOffice\Access\Samples\northwind.mdb';'admin';'
mypwd', Orders)
   AS a
GO
```

+ *OPENROWSET với INNER JOIN một bảng khác*.

```
USE pubs
GO
SELECT c.*, o.*
FROM Northwind.dbo.Customers AS c INNER JOIN
   OPENROWSET('Microsoft.Jet.OLEDB.4.0',
   'c:\MSOffice\Access\Samples\northwind.mdb';'admin';'
mypwd', Orders)
   AS o
   ON c.CustomerID = o.CustomerID
GO
```

Lệnh OpenDataSource.

Lệnh OpenDataSource thực hiện mở dữ liệu ngoài Instance, không cần đến linked_server.

+ *Kết nối đến Instance khác*.

```
SELECT   *
```

```
FROM      OPENDATASOURCE(
          'SQLOLEDB',
          'Data Source=ServerName;User
ID=MyUID;Password=MyPass'
     ).Northwind.dbo.Categories
```

+ Kết nối đến Excel.

```
SELECT *
FROM OpenDataSource( 'Microsoft.Jet.OLEDB.4.0',
  'Data Source="c:\Finance\account.xls";User
ID=Admin;Password=;Extended properties=Excel
5.0')...xactions
```

Lệnh OpenQuery.

Lệnh OpenQuery thực hiện thao tác với dữ liệu ngoài thông qua LinkedServer.

```
EXEC sp_addlinkedserver 'OracleSvr',
   'Oracle 7.3',
   'MSDAORA',
   'ORCLDB'
GO
SELECT *
FROM OPENQUERY(OracleSvr, 'SELECT name, id FROM
joe.titles')
GO
```

MỘT SỐ HÀM CƠ BẢN.

Hàm hệ thống.

DB_ID	Trả về ID của CSDL khi biết tên.
DB_NAME	Trả về tên CSDL khi biết ID.
HOST_ID	Trả về ID của máy chủ.
HOST_NAME	Trả về tên máy chủ

SUSER_ID	Trả về ID User của Server khi biết tên
SUSER_NAME	Trả về tên User của Server khi biết ID.
USER_ID	Trả về ID User khi biết tên
USER_NAME	Trả về tên User khi biết ID

Hàm thao tác với chuỗi.

+ SUBSTRING - Lấy chuỗi nhỏ trong chuỗi.

SUBSTRING **(** *expression* **,** *start* **,** *length* **)**

Sử dụng với chuỗi ký tự:

```
USE pubs
SELECT au_lname, SUBSTRING(au_fname, 1, 1)
FROM authors
ORDER BY au_lname
```

Sử dụng với text, ntext, image:

```
USE pubs
SELECT pub_id, SUBSTRING(logo, 1, 10) AS logo,
   SUBSTRING(pr_info, 1, 10) AS pr_info
FROM pub_info
WHERE pub_id = '1756'
```

+ CHARINDEX – Trả về vị trí bắt đầu một mẫu trong chuỗi.

CHARINDEX **(** *expression1* **,** *expression2* [**,** *start_location*] **)** – Tìm vị trí xuất hiện chuỗi expression1 trong expression2.

Ví dụ tìm chuỗi ‘wonderful’ trong cột notes của bảng titles:

```
USE pubs
GO
SELECT CHARINDEX('wonderful', notes)
FROM titles
WHERE title_id = 'TC3218'
GO
```

+ PATINDEX – Trả về vị trí xuất hiện của mẫu trong chuỗi.

PATINDEX **(** '*%pattern%*' **,** *expression* **)**

Ví dụ trìm vị trí xuất hiện mẫu ‘%wonderful%’:

```
USE pubs
GO
SELECT PATINDEX('%wonderful%', notes)
FROM titles
WHERE title_id = 'TC3218'
GO
```

Ví dụ trìm vị trí xuất hiện mẫu ‘%won_erful%’:

```
USE pubs
GO
SELECT PATINDEX('%won_erful%', notes)
FROM titles
WHERE title_id = 'TC3218'
GO
```

+ STR – Chuyển dữ liệu kiểu số sáng chuỗi.

STR **(** *float_expression* [**,** *length* [**,** *decimal*]] **)**

Ví dụ chuyển số sang chuỗi có độ dài 6, làm tròn sau dấu phảy 1 số.

```
SELECT STR(123.45, 6, 1)
GO
```

Kết quả là chuỗi ‘123.5’

Ví dụ sử dụng với hàm Floor lấy giá trị nguyên nhỏ hơn của một số thực:

```
SELECT STR (FLOOR (123.45), 8, 3)
GO
```

Kết quả là ‘123.000’

+ STUFF – Chèn một chuỗi vào một chuỗi khác.

Hàm Stuff thực hiện xóa chuỗi nhỏ trong một chuỗi sau đó thực hiện chèn một chuỗi mới vào vị trí bắt đầu.

STUFF **(** *character_expression* **,** *start* **,** *length* **,** *character_expression* **)**

Ví dụ:

```
SELECT STUFF('abcdef', 2, 3, 'ijklmn')
GO
```

Kết quả thực hiện:

```
aijklmnef
```

+ SOUNDEX – Trả về hàm phát âm.

Hàm Soundex sử dụng so sánh phát âm giữa 2 chuỗi, ví dụ sau sẽ cho 2 mã Soundex như nhau:

```
SELECT SOUNDEX ('Smith'), SOUNDEX ('Smythe')
```

Kết quả thực hiện:

```
S530  S530
```

+ *Defference – So sánh giá trị hàm Soundex giữa 2 chuỗi*: Giá trị trả về từ 0 đến 4, 4 là giá trị giống nhau nhất. Ví dụ sau so sánh giữa 2 chuỗi:

```
SELECT DIFFERENCE('Smithers', 'Smythers')
GO
```

Kết quả thực hiện: 4

```
SELECT DIFFERENCE('Anothers', 'Brothers')
GO
```

Kết quả thực hiện: 2

+ UNICODE – Lấy mã unicode ký tự đầu tiên trong chuỗi.

+ NCHAR – Chuyển mã unicode thành ký tự.

Các hàm DateTime.

+ GETDATE: Trả về ngày, giờ hiện tại.

+ DATEPART: Trả về giá trị ngay hoặc tháng hoặc năm của một biểu thức ngày.

DATEPART **(** *datepart* **,** *date* **)**

Giá trị datepart theo bảng sau:

Datepart	Dạng rút gọn
year	**yy, yyyy**
quarter	**qq, q**
month	**mm, m**
dayofyear	**dy, y**
day	**dd, d**
week	**wk, ww**
weekday	**dw**
hour	**hh**
minute	**mi, n**
second	**ss, s**
millisecond	**ms**

```
SELECT DATEPART(m, 0), DATEPART(d, 0), DATEPART(yy, 0)
```

+ SET DATFIRST: Đặt ngày đầu tiên trong tuần.

+ SET DATEFORMAT: Đặt định dạng kiểu DateTime để nhập dữ liệu.

```
SET DATEFORMAT mdy
GO
DECLARE @datevar datetime
SET @datevar = '12/31/98'
SELECT @datevar
GO
```

+ DAY, MONTH, YEAR: Lấy giá trị ngày, tháng, năm.

+ ISDATE: Kiểm tra xem dữ liệu có hợp lệ DateTime không.

+ DATEDIFF: Xác định độ lệch giữa 2 giá trị DateTime.

DATEDIFF **(** *datepart* **,** *startdate* **,** *enddate* **)**

Ví dụ: Xác định số ngày đã phát hành sách.

```
USE pubs
GO
SELECT DATEDIFF(day, pubdate, getdate()) AS no_of_days
FROM titles
GO
```

+ DATEADD – Xác định giá DateTime mới khi thay đổi một khoảng thời gian.

DATEADD **(** *datepart* **,** *number***,** *date* **)**

```
USE pubs
GO
SELECT DATEADD(day, 21, pubdate) AS timeframe
FROM titles
```

```
GO
```

Các hàm chuyển đổi.

+ CONVERT

+ CAST

TRANSACTION – PHIÊN GIAO DỊCH.

Transaction là một đơn vị công việc trong nó bao gồm nhiều việc nhỏ, các việc này được thực hiện thành công thì Transaction thành công, dữ liệu thay đổi trong quá trình thực hiện của Transaction sẽ được cập nhật. Nếu trong quá trình có phát sinh lỗi thì Transaction sẽ lặp lại (Roll Back hoặc Cancel), dữ liệu không được cập nhật. Một phiên giao dịch có 4 đặc tính ACID (Atomicity, Consistency, Isolation, Durability).

Atomicity – Nguyên tố: Một phiên giao dịch là một đơn vị công việc nhỏ nhất, tất cả dữ liệu thay đổi trong phiên giao dịch được thực hiện hoặc tất cả không được thực hiện.

Consistency- Nhất quán: Giao dịch sẽ không thực hiện nếu có một thao tác xung khắc về mặt logic hoặc quan hệ. Tính nhất quán rất quan trọng với mô hình ứng dụng client/server, với mô hình dạng này tại một thời điểm có thể có nhiều giao dịch thực hiện đồng thời, nếu một giao dịch nào đó không nhất quán thì tất cả các giao dịch khác sẽ thực hiện sai, dẫn đến sự vi phạm toàn vẹn dữ liệu.

Isolation – Tách biệt: Tại một thời điểm có nhiều phiên giao dịch đồng thời, các phiên giao dịch chỉ tác động với nhau khi dữ liệu được cập nhật (kết thúc phiên). Giả sử có 2 phiên giao dịch có tác động

Durability - Bền vững: Sau khi giao dịch hoàn tất, dữ liệu ở trạng thái bền vững.

Một phiên giao dịch được xác định bắt đầu, kết thúc:

Bắt đầu phiên giao dịch.

Phiên giao dịch có 3 loại: explicit transaction, implicit transaction, autocommit transaction.

Explicit transaction: Là kiểu phiên giao dịch rõ, được bắt đầu bằng lệnh BEGIN TRANSACTION, đối với phiên giao dịch phân tán thì được bắt đầu bằng lệnh

BEGIN DISTRIBUTED TRAN

[*transaction_name* | *@tran_name_variable*]

Đặt tên giao dịch:

```
DECLARE @TranName VARCHAR(20)
SELECT @TranName = 'MyTransaction'

BEGIN TRANSACTION @TranName
GO
USE pubs
GO
UPDATE roysched
SET royalty = royalty * 1.10
WHERE title_id LIKE 'Pc%'
GO

COMMIT TRANSACTION MyTransaction
GO
```

Đánh dấu trong giao dịch:

```
BEGIN TRANSACTION RoyaltyUpdate
   WITH MARK 'Update royalty values'
GO
USE pubs
GO
UPDATE roysched
   SET royalty = royalty * 1.10
   WHERE title_id LIKE 'Pc%'
GO
COMMIT TRANSACTION RoyaltyUpdate
GO
```

Autocommit transaction: Mỗi câu lệnh tự cập nhật dư liệu khi nó kết thúc, không cần câu lệnh điều khiển phiên giao dịch.

Implicit transaction: Là phiên giao dịch ẩn, đặt chế độ này thông qua hàm API hoặc lệnh SET IMPLICIT_TRANSACTIONS ON. Khi phiên giao dịch kết thúc, câu lệnh T-SQL tiếp theo sẽ khởi động phiên giao dịch mới.

```
SET IMPLICIT_TRANSACTIONS { ON | OFF }
```

Sử dụng kết hợp với Implicit transaction:

```
USE pubs
GO

CREATE table t1 (a int)
GO
INSERT INTO t1 VALUES (1)
GO

PRINT 'Use explicit transaction'
BEGIN TRAN
INSERT INTO t1 VALUES (2)
SELECT 'Tran count in transaction'= @@TRANCOUNT
COMMIT TRAN
SELECT 'Tran count outside transaction'= @@TRANCOUNT
GO

PRINT 'Setting IMPLICIT_TRANSACTIONS ON'
GO
SET IMPLICIT_TRANSACTIONS ON
GO

PRINT 'Use implicit transactions'
GO
-- No BEGIN TRAN needed here.
INSERT INTO t1 VALUES (4)
SELECT 'Tran count in transaction'= @@TRANCOUNT
COMMIT TRAN
SELECT 'Tran count outside transaction'= @@TRANCOUNT
GO

PRINT 'Use explicit transactions with
IMPLICIT_TRANSACTIONS ON'
GO
```

```
BEGIN TRAN
INSERT INTO t1 VALUES (5)
SELECT 'Tran count in transaction'= @@TRANCOUNT
COMMIT TRAN
SELECT 'Tran count outside transaction'= @@TRANCOUNT
GO

SELECT * FROM t1
GO

-- Need to commit this tran too!
DROP TABLE t1
COMMIT TRAN
GO
```

Kết thúc phiên giao dịch.

Sử dụng lệnh Commit trong phiên giao dịch.

COMMIT [TRAN [SACTION] [*transaction_name* | **@***tran_name_variable*]]

+ Commit một phiên giao dịch.

```
BEGIN TRANSACTION
USE pubs
GO
UPDATE titles
SET advance = advance * 1.25
WHERE ytd_sales > 8000
GO
COMMIT
GO
```

+ Commit nhiều phiên giao dịch lồng nhau.

```
CREATE TABLE TestTran (Cola INT PRIMARY KEY, Colb CHAR(3))
GO
BEGIN TRANSACTION OuterTran -- @@TRANCOUNT set to 1.
GO
INSERT INTO TestTran VALUES (1, 'aaa')
GO
BEGIN TRANSACTION Inner1 -- @@TRANCOUNT set to 2.
```

```
GO
INSERT INTO TestTran VALUES (2, 'bbb')
GO
BEGIN TRANSACTION Inner2 -- @@TRANCOUNT set to 3.
GO
INSERT INTO TestTran VALUES (3, 'ccc')
GO
COMMIT TRANSACTION Inner2 -- Decrements @@TRANCOUNT to 2.
-- Nothing committed.
GO
COMMIT TRANSACTION Inner1 -- Decrements @@TRANCOUNT to 1.
-- Nothing committed.
GO
COMMIT TRANSACTION OuterTran -- Decrements @@TRANCOUNT to 0.
-- Commits outer transaction OuterTran.
GO
```

Hủy bỏ và quay lại phiên giao dịch.

Sử dụng lệnh RollBack Transaction hủy bỏ những thực hiện và quay lại phiên giao dịch.

ROLLBACK [TRAN [SACTION]
[*transaction_name* | *@tran_name_variable*
| *savepoint_name* | *@savepoint_variable*]]

LOCK – KHÓA.

Khi 2 hay nhiều người cùng truy nhập đồng thời một CSDL, SQL Server sử dụng khoá để xác định hoạt động cho một người và không xác didnhj cho người khác. Khoá là việc ngăn không cho những người đọc dữ liệu mà không bị người khác sử đổi.

Hầu hết SQL Server đều khoá tự động, bạn có thể thiết tiết CSDL một cách có hiệu quả hơn bằng việc tìm hiểu về khoá và chọn khoá cho ứng dụng của bạn.

Tìm hiểu về khoá.

Khoá gồm các loại sau:

Kiểu khoá	Mô tả

Shared	Là khoá không làm thay đổi, ghi dữ liệu, dùng cho lệnh Select
Update	Khoá hoặc cho phép sửa đổi dữ liệu
Exclusive	Khoá với các thao tác Update, Insert, Delete

Một số phạm vi khoá như sau:

Tên	Mô tả
Page	Trang dữ liệu 2K hoặc trang chỉ mục Index, thường được dùng
Extent	Nhóm các trang có kích thước 8k, chỉ dùng với trường hợp xác định
Table	Cả bảng dữ liệu, gồm dữ liệu và index
Intent	Là kiểu đặc biệt để đặt kiểu khoá của trang hiện tại trên bảng

Bảng xác định hiệu lực của các kiểu khoá

	Shared	**Update**	**Exclusive**
Shared	Yes	Yes	No
Update	Yes	No	No
Exclusive	No	No	No

Ví dụ: Khi đặt chế độ khoá là Exclusive thì những phiên giao dịch khác không thể yêu cầu bất cứ loại khoá nào đến khi hoá Exclusive bị bỏ.

Xem thông tin về khoá.

Để xem thông tin về khoá đạng sử dụng trong SQL Server ta làm như sau:

- Chọn đối tượng cần xem khoá
- Thực hiện thủ tục sp_lock

Chọn kiểu khoá.

Khoá đực đặt trong các câu lệnh như: SELECT, INSERT, UPDATE, và DELETE , sau đây là bảng mô tả các kiểu khoá đối với phương thức nói trên

Tên	Mô tả
NOLOCK	Được sử dụng với câ lệnh Select, người đoc có thể đọc dữ liệu khi dữ liệu gốc khi chưa được ghi dữ liệu mới trong giao dịch đang sử dụng
HOLDLOCK	Khoa Shared được giữ đến khi phiên giao dịch được hoàn tất

	khi khoá chưa được giải phóng
UPDLOCK	Dùng để cập nhật dữ liệu của kiểu khoá Shared trong quá trình đọc bảng dữ liệu và được giữ đến khi kết thúc lệnh của phiên giao dịch. Khoa này dùng khi cập nhật dữ liệu, ngăn không cho người khác đọc đến khi phiên giao dịch cập nhật được hoàn tất
TABLOCK	Dùng khoá Shared trên một bảng dữ liệu, cho phép những người khác đọc dữ liệu nhưng ngăn không cho cập nhật
PAGELOCK	sử dụng kiểu khoá Shared phạm vi trang dữ liệu (Page), đây là loại khoá ngầm định
TABLOCKX	Dùng kiểu khoá Exlusive trên một bảng dữ liệu, ngăn người khác đọc và cập nhật dữ liệu từ bảng và giữ đến khi kết thúc lệnh, phiên giao dịch

- Cách đặt khoá như sau: Dùng lệnh SET
- Ví dụ 1:

```
USE pubs
GO
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE
GO
BEGIN TRANSACTION
SELECT au_lname FROM authors WITH (NOLOCK)
GO
```

- Vớ d? 2:

```
Select * from authors(UPDLOCK)
```

Đặt mức khoá.

Dùng đặt, điều khiển khoá trong các giao dịch của SQL Server

- Cú pháp:

```
SET TRANSACTION ISOLATION LEVEL
  { READ COMMITTED
    | READ UNCOMMITTED
    | REPEATABLE READ
    | SERIALIZABLE
  }
```

Trong đó:

- Read Committed: Dùng kiểu khoá Shared trong quá trình đọc dữ liệu

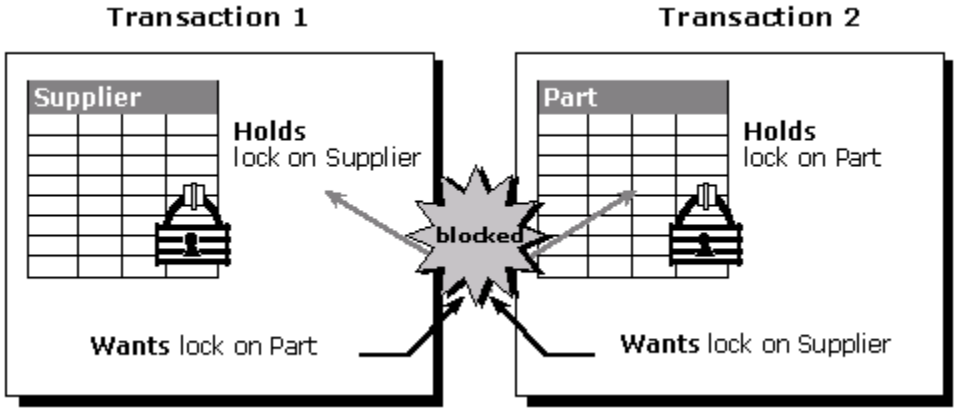
- Read Uncommitted: Không đặt khoá Shared và khoá Exclusive, có thể đọc dữ liệu gốc khi đang có phiên giao địch sửa đổi dữ liệu
- RepeateTable Read: Khoá tất cả dữ liệu đang được sử dụng trong truy vấn, ngăn những người khác sửa dữ liệu nhưng người khác có thể chèn thêm dữ liệu mới vào bảng (hàng mới)
- Serializable: Đặt khoá trong một tập dữ liệu (khoá phạm vi) ngăn không cho người khác có thể sửa, thêm hàng mới vào tập dữ liệu đến khi giao dịch kết thúc, tương ự như HoldLock trong lệnh Select

- Ví dụ:

```
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
GO
BEGIN TRANSACTION
SELECT * FROM publishers
SELECT * FROM authors
...
COMMIT TRANSACTION
```

Khoá chết (DeadLock).

Trong hệ quản trị CSDL quan hệ nói rieneg và các hệ quản trị khác nói chung, việc xuất hiện nhiều luồng dữ liệu đồng thời trong CSDL là thường xuyên xảy ra, một giao dịch có thể lấy dữ liệu từ nhiều nguồn dữ liệu khác nhau, hai giao dịch trong cùng CSDL có thể cùng chung một nguồn dữ liệu nào đó nên việc các giao dịch này đặt các mức khoá khác nhau cho các nguồn dữ liệu mà no năm giữ là không thể tránh khỏi, ví dụ tên sơ đồ sau mô tả sự giao chéo về nguồn dữ liệu trong giao dịch



Trong giao dịch 1 và 2 đều đặt các bảng dữ liệu ở mức khoá Exlusive, như vậy giao dịch 1 chỉ thực hiện được khi giao dịch 2 thực hiện xong hoặc quay lại trạng thái ban đầu, ngược lại giao dịch 2 cũng chừo giao dịch 1 thực hiện xong hoặc quay lại trạng thai ban đầu. Cứ như vây thì cả 2 giao dịch sẽ không bao giơ

kết thúc được phiên giao dịch của mình. Phần chung của khoá nói trên gọi là khoá chết, và được khoá theo khối (block).

GRAND – GÁN QUYỀN.

Lệnh Grand thực hiện gán quyền cho user hoặc role của SQL Server. Người thực hiện Grand phải có quyền được thực hiện phân quyền cho user. Có 2 hình thức gán quyền: gán quyền thực hiện câu lệnh, gán quyền thao tác với đối tượng.

Gán quyền thao tác câu lệnh.

GRANT { ALL | *statement* [**,***...n*] }
TO *security_account* [**,***...n*]

Các câu lệnh:

- CREATE DATABASE
- CREATE DEFAULT
- CREATE FUNCTION
- CREATE PROCEDURE
- CREATE RULE
- CREATE TABLE
- CREATE VIEW
- BACKUP DATABASE
- BACKUP LOG

SQL Server ngầm định một số nhóm có quyền thực hiện câu lệnh như sau:

	dbcreator	**processadmin**	**securityadmin**	**serveradmin**	**bulkadmin**
ALTER DATABASE	X				
CREATE DATABASE	X				

BULK INSERT						X
DBCC				X (1)		
DENY			X (2)			
GRANT			X (2)			
KILL		X				
RECONFIGURE				X		
RESTORE	X					
REVOKE			X (2)			
SHUTDOWN				X		

	db_ owner	**db_ datareader**	**db_ datawriter**	**db_ ddladmin**	**db_ backup operator**	**db_ security admin**
ALTER DATABASE	X			X		
ALTER FUNCTION	X			X		
ALTER PROCEDURE	X			X		
ALTER TABLE	X (1)			X		
ALTER TRIGGER	X			X		
ALTER VIEW	X (1)			X		
BACKUP	X				X	
CHECKPOINT	X				X	
CREATE DEFAULT	X			X		
CREATE FUNCTION	X			X		
CREATE INDEX	X (1)			X		
CREATE PROCEDURE	X			X		
CREATE RULE	X			X		
CREATE TABLE	X			X		
CREATE TRIGGER	X (1)			X		
CREATE VIEW	X			X		
DBCC	X				X (2)	
DELETE	X (1)		X			
DENY	X					X
DENY on object	X					
DROP	X (1)			X		
EXECUTE	X (1)					

GRANT	X					X
GRANT on object	X (1)					
INSERT	X (1)		X			
READTEXT	X (1)	X				
REFERENCES	X (1)			X		
RESTORE	X					
REVOKE	X					X
REVOKE on object	X (1)					
SELECT	X (1)	X				
SETUSER	X					
TRUNCATE TABLE	X (1)			X		
UPDATE	X (1)		X			
UPDATE STATISTICS	X (1)					
UPDATETEXT	X (1)		X			
WRITETEXT	X (1)		X			

Các user được gán quyền có thể là user của SQL Server hoặc user của Windows NT.

Ví dụ gán quyền thao tác câu lệnh cho 3 user (trong đó có 2 user của SQL Server và 1 user của Windows NT):

```
GRANT CREATE DATABASE, CREATE TABLE
TO Mary, John, [Corporate\BobJ]
```

Ví dụ gán quyền thao tác cho role và user:

```
USE pubs
GO

GRANT SELECT
ON authors
TO public
GO

GRANT INSERT, UPDATE, DELETE
ON authors
TO Mary, John, Tom
```

```
GO
```

Gán quyền thao tác đối tượng.

Là việc gán quyền cho các user hoặc role có quyền thao tác với các đối tượng của SQL Server.

Ví dụ gán quyền thao tác cho Role:

```
GRANT CREATE TABLE TO Accounting
```

Ví dụ gán quyền để gán quyền thao tác cho user khác: Ví dụ Jean là dbo của bảng Plan_data, Jean thực hiện gán quyền với chức năng GRAND_OPTION cho role accounting, Jill thuộc role nói trên và Jill gán quyền được Select cho Jack, Jack không là thành viên của Accounting.

```
/* User Jean */
GRANT SELECT ON Plan_Data TO Accounting WITH GRANT
OPTION

/* User Jill */
GRANT SELECT ON Plan_Data TO Jack AS Accounting
```

Thủ cục sp_grandlogin.

Là thủ tục thực hiện gán quyền truy nhập cho user của Windows NT hoặc nhóm user của Windows NT.

sp_grantlogin [**@loginame** =] '*login*'

Ví dụ gán quyền truy nhập SQL Server cho BobJ.

```
EXEC sp_grantlogin 'Corporate\BobJ'
```

Thủ tục sp_grandaccess.

Gán quyền khai thác cho user của SQL Server hoặc Windows NT.

sp_grantdbaccess [**@loginame** =] '*login*'
[**,**[**@name_in_db** =] '*name_in_db*' [**OUTPUT**]]

Ví dụ gán quyền khai thác cho user của Windows và lấy theo tên mới.

```
EXEC sp_grantdbaccess 'Corporate\GeorgeW', 'Georgie'
```

REVOKE – TƯỚC QUYỀN.

Revoke là câu lệnh tước quyền khai thác của user.

Tước quyền được thực hiện câu lệnh.

REVOKE { ALL | *statement* [**,***...n*] }
FROM *security_account* [**,***...n*]

Ví dụ tước quyền khai thác với 2 user:

```
REVOKE CREATE TABLE FROM Joe, [Corporate\BobJ]
```

Ví dụ tước quyền khai thác 2 câu lệnh với các user:

```
REVOKE CREATE TABLE, CREATE DEFAULT
FROM Mary, John
```

Tước quyền khai thác của user với đối tượng.

Ví dụ tước quyền thực hiện lệnh Select trong role Budget_data đối với Mary:

```
REVOKE SELECT ON Budget_Data TO Mary
```

DENY – TỪ CHỐI QUYỀN.

Là câu lệnh từ chối quyền đối với user, user chỉ thực hiện được quyền khi có chỉ định rõ ràng.

Ví dụ từ chối quyền thực hiện lệnh với các user:

```
DENY CREATE DATABASE, CREATE TABLE
TO Mary, John, [Corporate\BobJ]
```

Ví dụ gán quyền khai thác cho role, sau đó thực hiện từ chối thực hiện của các user trong role:

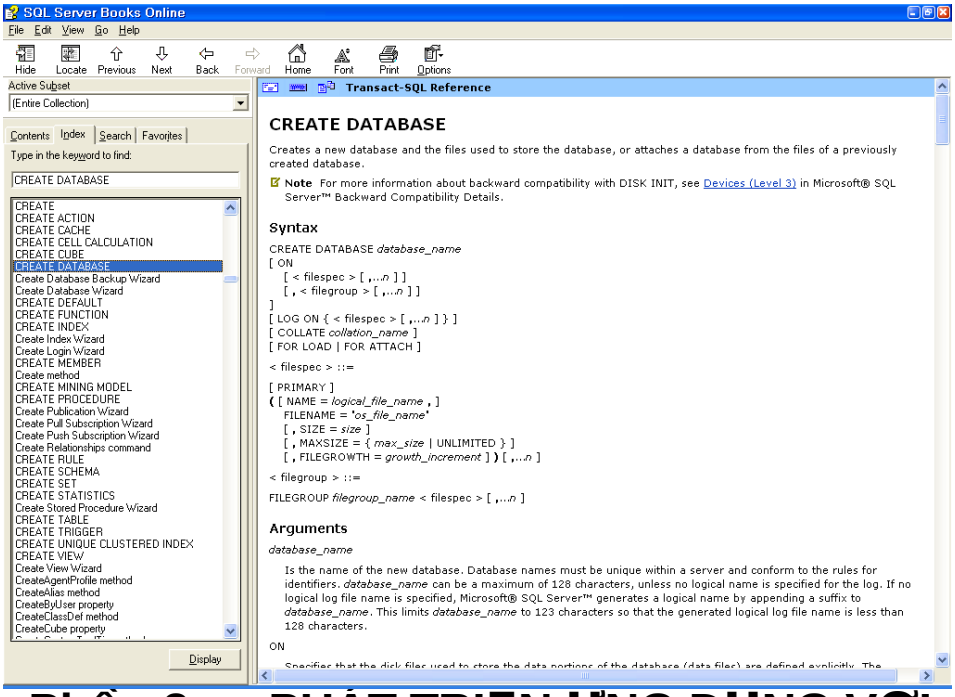
```
USE pubs
GO

GRANT SELECT
ON authors
TO public
GO

DENY SELECT, INSERT, UPDATE, DELETE
ON authors
TO Mary, John, Tom
```

Ví dụ từ chối quyền của role:

```
DENY CREATE TABLE TO Accounting
```

TRỢ GIÚP.

Trong quá trình thực hiện soạn lệnh T-SQL bạn có thể thực hiện tra cứu lệnh trong Book Online.



Phần 3. PHÁT TRIỂN ỨNG DỤNG VỚI SQL SERVER

Trong phần này ta sẽ xem xét kỹ thuật phát triển ứng dụng với SQL Server từ các ngôn ngữ lập trình (Visual Basic, C++, VBScript,…). Các ứng dụng khai thác CSDL của SQL Server thực hiện các bước sau:

+ Kết nối từ ứng dụng đến SQL Server.

+ Xây dựng cơ sở dữ liệu.

+ Thực hiện các lệnh khai thác hoặc thủ tục của SQL Server.

+ Khai thác dữ liệu thông qua công cụ có sẵn.

+ Ngắt kết nối.

GIỚI THIỆU.

Thiết kế ứng dụng là việc thực hiện tạo giao diện (API – Application Program Interface) giao tiếp với SQL Server, việc thực hiện kết nối thực hiện

thông qua các công cụ ADO, URL, OLE DB, ODBC, Embedded SQL for C, DB-Library. Khi sử dụng các công cụ kết nối dữ liệu thao tác dưới dạng bảng hoặc dạng tài liệu XML.

+ Dữ liệu dưới dạng bảng được thực hiện thông qua các công cụ kết nối ADO, OLE DB, ODBC, Embedded SQL for C, DB-Library.

+ Dữ liệu thực hiện thông qua tài liệu XML thông qua các công cụ ADO, URL, OLE DB.

KẾT NỐI VỚI SQL SERVER BẰNG ADO.

ADO viết tắt của cụm từ ActiveX Data Object là công cụ giao tiếp với dữ liệu của nhiều hệ quản trị CSDL khác nhau, SQL Server là một ví dụ cho việc giao tiếp.

ADO sử dụng với CSDL quan hệ hoặc sử dụng với CSDL đa chiều, khi đó gọi là ADO MD (ADO Multi Dimention). ADO sử dụng kết nối kiểu OLE DB hoặc các thư viện kết nối COM (Component Object Model).

OLE DB sử dụng 2 phương thức Microsoft OLE DB Provider for SQL Server (SQLOLEDB) và Microsoft OLE DB Provider for ODBC (MSDASQL).

ADO có thể thực hiện từ các ngôn ngữ lập trình Visual Basic, ASP, C++.

Cấu trúc ứng dụng sử dụng ADO.

ADO gồm các thành phần cơ bản sau: Application, ADO, OLE DB Provider, Data Source.

Thành phần	Chức năng
Application	Gọi các đối tượng, thành phần, phương thức và các thuộc tính của ADO. Thông qua các thành phần này ứng dụng sẽ gửi các câu lệnh SQL và nhận kết quả xử lý.
ADO	Quản lý việc trao đổi dữ liệu giữa ứng dụng và OLE DB
OLE DB provider	Xử lý các lệnh gọi từ ứng dụng qua ADO, kết nối với Data Source. Processes all ADO calls from the application, connects to a data source, passes SQL statements from the application to the data source, and returns results to the application.
Data source	Contains the information used by a provider to access a specific instance of data in a DBMS.

Khi thực hiện lập trình ứng dụng với SQL Server sử dụng ADO, người lập trình phải thực hiện các thao tác sau:

+ Kết nối đến nguồn dữ liệu (data source).

+ Gửi câu lệnh SQL đến nguồn dữ liệu.

+ Xử lý kết quả nhận được từ câu lệnh đã gửi.

+ Xử lý các lỗi và thông báo.

+ Ngắt kết nối đến nguồn dữ liệu.

Đối với một số ứng dụng phức tạp sử dụng ADO có thể sử dụng một số thao tác sau:

+ Sử dụng con trỏ (cursor) để điều khiển vị trí trong tập kết quả.

+ Thực hiện thủ tục lưu trữ trên Server.

+ Thực hiện hàm tự định nghĩa trên Server.

+ Quản lý các phép truy vấn mà có nhiều tập kết quả.

+ Yêu cầu kết thúc hoặc lặp lại một phiên giao dịch.

+ Quản lý các thao tác với kiểu dữ liệu lớn (text, image).

+ Thực hiện các thao tác với XML sử dụng phép truy vấn XPath.

Kết nối đến SQL Server.

Để kết nối đến SQL Server, các công việc cơ bản cần thực hiện như sau:

+ Cấu hình kết nối.

+ Thiết lập hoặc ngắt kết nối đến nguồn dữ liệu.

+ Xác định OLE DB provider.

+ Thực hiện truy vấn.

+ Quản lý các phiên làm việc trên kết nối.

Khi sử dụng SQLOLEDB ta phải thực hiện đặt các thuộc tính sau cho kết nối:

+ **Initial Catalog:** Xác định CSDL.

+ **Data Source:** Xác định tên Server.

+ **Integrated Security**: Xác định chế độ xác thực, nếu là **SSPI chế độ xác thực là** Windows Authentication, hoặc xác định **User ID, Password** của chế độ xác thực SQL Server Authentication.

Ví dụ thực hiện kết nối đến SQL Server đặt từng thuộc tính tiêng biệt từ Visual Basic:

```
' Initialize variables.
Dim cn As New ADODB.Connection
. . .
Dim ServerName As String, DatabaseName As String, _
   UserName As String, Password As String

' Put text box values into connection variables.
ServerName = txtServerName.Text
DatabaseName = txtDatabaseName.Text
UserName = txtUserName.Text
Password = txtPassword.Text

' Specify the OLE DB provider.
cn.Provider = "sqloledb"

' Set SQLOLEDB connection properties.
cn.Properties("Data Source").Value = ServerName
cn.Properties("Initial Catalog").Value = DatabaseName

' Decision code for login authorization type:
' Windows NT or SQL Server authentication.
If optWinNTAuth.Value = True Then
    cn.Properties("Integrated Security").Value = "SSPI"
Else
    cn.Properties("User ID").Value = UserName
    cn.Properties("Password").Value = Password
End If

' Open the database.
cn.Open
```

Ví dụ kết nối đến SQL Server sử dụng chuỗi kết nối:

```
' Initialize variables.
Dim cn As New ADODB.Connection
Dim provStr As String

' Specify the OLE DB provider.
cn.Provider = "sqloledb"

' Specify connection string on Open method.
ProvStr =
"Server=MyServer;Database=northwind;Trusted_Connection=
yes"
cn.Open provStr
```

Ví dụ kết nối sử dụng ODBC:

```
Dim cn As New ADODB.Connection

cn.ConnectionTimeout = 100
' DSN connection. You can use variables for the
parameters.
cn.Open "MyDataSource", "sa", "MyPassword"
' Alternative syntax follows:
' cn.Open "DSN=DataSourceName;UID=sa;PWD=Password;"

cn.Close
```

Ví dụ kết nối xác định Driver của SQL Server:

```
Dim cn As New ADODB.Connection

' Connection to SQL Server without using ODBC data
source.
cn.Open "Driver={SQL
Server};Server=Server1;Uid=SA;Pwd=;Database=northwind"

cn.Close
```

Thực hiện truy vấn.

Thực hiện truy vấn sử dụng đối tượng Command.

cmd.Execute(*NumRecords, Parameters, Options***)**

Đối tượng Command có thể thực hiện nhiều kiểu câu lệnh (Select, Update, Insert, Delete, Create, Drop), đối với lệnh Select kết quả thực hiện là một recordset.

Set *rs* = cmd.Execute(*NumRecords, Parameters, Options***)**

Kiểu lệnh thực hiện trong Command được xác định theo option của lệnh, gồm một số kiểu sau:

Tên kiểu	Mô tả
adCmdFile	Tên file chứa đối tượng recordset
adCmdStoreProc	Stored procedure
adCmdTable	Tên bảng
adCmdTableDirect	Tên bảng mà các cột được truy vấn
adCmdText	Câu lệnh SQL
adCmdUnknown	Chưa xác định
adCmdUnspecified	Chưa xác định tham số cho lệnh

Thực hiện truy vấn thông qua đối tượng connection.

```
Dim cn As New ADODB.Connection
. . .
Dim rs As New ADODB.Recordset

cmd1 = txtQuery.Text
Set rs = cn.Execute(cmd1)
```

Thực hiện truy vấn có sử dụng tham số.

Khi thực hiện các thủ tục có tham số truyền vào các ứng dụng phải truyền tham số, trong phần này sẽ giới thiệu một ví dụ sử dụng đối tượng parameter.

Tạo thủ tục:

```
USE NORTHWIND
GO
drop proc myADOParaProc
GO
```

```
CREATE PROC myADOParaProc
@categoryid int(4)
AS
SELECT * FROM products WHERE categoryid = @categoryid
GO
```

Sử dụng đối tượng parameter truyền tham số là số nguyên xác định categoryID:

```
Dim cn As New ADODB.Connection
Dim cmd As New ADODB.Command
Dim rs As New ADODB.Recordset
Dim prm As ADODB.Parameter
Dim fld As ADODB.Field
Dim provStr As String

' Connect using the SQLOLEDB provider.
cn.Provider = "sqloledb"

' Specify connection string on Open method.
provStr =
"Server=MyServer;Database=northwind;Trusted_Connection=yes"
cn.Open provStr

' Set up a command object for the stored procedure.
Set cmd.ActiveConnection = cn
cmd.CommandText = "myADOParaProc"
cmd.CommandType = adCmdStoredProc
cmd.CommandTimeout = 15

' Set up a new parameter for the stored procedure.
Set prm = Cmd.CreateParameter("CategoryID", adInteger,
adParamInput, 4, 7)
Cmd.Parameters.Append prm

' Create a recordset by executing the command.
Set rs = cmd.Execute
Set Flds = rs.Fields

' Print the values for all rows in the result set.
While (Not rs.EOF)
   For Each fld in Flds
      Debug.Print fld.Value
   Next
```

```
   Debug.Print ""
    rs.MoveNext
Wend

' Close recordset and connection.
rs.Close

cn.Close
```

Đối tượng Recordset.

Sử dụng đối tượng Recordset lưu trữ kết quả của lệnh Select.

```
Dim cn As New ADODB.Connection
Dim rs As ADODB.Recordset
. . .
cmd1 = txtQuery.Text
Set rs = New ADODB.Recordset
rs.Open cmd1, cn
rs.MoveFirst
. . .
' Code to loop through result set(s)
```

Đối tượng Field.

Sử dụng đối tượng field là các cột của Recordset, thông qua nó ta có thể láy giá trị, thuộc tính của cột.

```
Dim rs As New ADODB.Recordset
Dim fld As ADODB.Field
Dim cn As ADODB.Connection
Dim cmdText As String

cn.Provider = "sqloledb"
cn.Properties("Data Source").Value = "MyServerName"
cn.Properties("Initial Catalog").Value = "northwind"
cn.Properties("Integrated Security").Value = "SSPI"
cn.Open

cmdText = "select * from authors"
```

```
rs.Open cmdText, cn
Set Flds = rs.Fields
Dim TotalCount As Integer
TotalCount = Flds.Count

For Each fld In Flds
    Debug.Print fld.Name
    Debug.Print fld.Type
    Debug.Print fld.Value
Next
rs.Close
```

Sử dụng con trỏ.

Khi sử dụng đối tượng Recordset của ADO, ta có thể sử dụng nhiều kiểu con trỏ khác nhau xác định kiểu khóa, điều khiển vị trí,...

```
Dim rs As New ADODB.Recordset
. . .
rs.Open "SELECT * FROM titles", , adOpenDynamic,
adLockOptimistic

rs.Close
```

COn trỏ nói trên gồm những thuộc tính cơ bản sau: **CursorType**, **CursorLocation**, **LockType**, **CacheSize**.

Thuộc tính	Mô tả
CursorType	- **adOpenForwardOnly:** Ngầm định Xác định kiểu con trỏ được sử dụng: - **adOpenForwardOnly:** Chỉ đọc, chỉ có thể cập nhật dữ liệu trên hàng dữ liệu hiện thời. - **adOpenStatic:** Trạng thái tĩnh, khi mở kiểu này hệ thống sẽ cung cấp một ảnh dữ liệu (snapshot), dữ liệu thay đổi trên bảng cơ sở sẽ không được thể hiện trên snapshot dạng này. - **adOpenKeyset:** Theo vị trí tùy chọn, khi di chuyển hàng cập nhật con trỏ kiểu này sẽ chiếu đến hàng dữ liệu cơ sở, hàng dữ liệu được khó và bạn có thể cập nhật, lấy dữ liệu từ hàng cơ sở**.** - **adOpenDynamic:** Động, con trỏ kiểu này gần giống keyset cursor , nhưng con trỏ kiểu này phản ảnh những thay đổi

	trên bảng cơ sở.
CursorLocation	**-adUseServer:** Ngầm định. - **adUseClient:** Nếu đặt là ta chỉ có thể mở ở trạng thái tĩnh.
LockType	**-adLockReadOnly:** Ngầm định. Xác định kiểu khóa trong quá trình cập nhật dữ liệu (**adLockPessimistic**, **adLockOptimistic**, **adLockBatchOptimistic**).
CacheSize	Ngầm định: 1 Xác định số hàng đặt trong bộ đệm hoặc đọc trong một thời điểm.

Các phương thức dịch chuyển hàng dữ liệu.

Khi sử dụng đối tượng Recordset bạn có thể dịch chuyển vị trí của hàng dữ liệu bằng các phương thức **MoveFirst**, **MoveLast**, **MoveNext**, **MovePrevious.** Đánh dấu vị trí theo phương thức **Bookmark,** phương thức **clone** để tạo một bản sao recordset.

Quản lý phiên làm việc.

Trong phần câu lệnh T-SQL ta đã xem xét việc điều khiển một hpiên làm việc (transaction), tuy nhiên ta có thể sử dụng đối tượng connection của ADO để điều khiển trực tiếp phiên làm việc như trong kịch bản lệnh nói trên bằng việc sử dụng các phương thức **BeginTrans**, **CommitTrans**, **RollbackTrans**. Xét ví dụ sau:

```
Dim cn As New ADODB.Connection
Dim rs As New ADODB.Recordset

. . .
' Open connection.
cn.Open

' Open titles table.
rs.Open "SELECT * FROM titles", Cn, adOpenDynamic,
adLockPessimistic
. . .
' Begin the transaction.
rs.MoveFirst
cn.BeginTrans
```

```
' User loops through the recordset making changes.
. . .
' Ask if the user wants to commit all the changes made.
If MsgBox("Save all changes?", vbYesNo) = vbYes Then
   cn.CommitTrans
Else
   cn.RollbackTrans
End If
```

Thực hiện các lệnh DDL.

Để thực hiện các lệnh DDL như CREATE TABLE, DROP TABLE, ALTER TABLE. Bạn có thể sử dụng đối tượng command của ADO, xét ví dụ sau:

```
Dim Cn As New ADODB.Connection
Dim Cmd As New ADODB.Command

' If the ADOTestTable does not exist, go to AdoError.
On Error GoTo AdoError

' Connect using the SQLOLEDB provider.
cn.Provider = "sqloledb"
cn.Properties("Data Source").Value = "MyServerName"
cn.Properties("Initial Catalog").Value = "northwind"
cn.Properties("Integrated Security").Value = "SSPI"
cn.Open

' Set up command object.
Set Cmd.ActiveConnection = Cn
Cmd.CommandText = "DROP TABLE ADOTestTable"
Cmd.CommandType = adCmdText
Cmd.Execute

Done:
    Cmd.CommandText = "SET NOCOUNT ON"
    Cmd.Execute
    Cmd.CommandText = "CREATE TABLE ADOTestTable (id
int, name char(100))"
    Cmd.Execute
    Cmd.CommandText = "INSERT INTO ADOTestTable
values(1, 'Jane Doe')"
    Cmd.Execute
    Cn.Close
```

```
Exit Sub

AdoError:
      Dim errLoop As Error
      Dim strError As String

      ' Enumerate Errors collection and display
properties of
      ' each Error object.
      Set Errs1 = Cn.Errors
      For Each errLoop In Errs1
            Debug.Print errLoop.SQLState
            Debug.Print errLoop.NativeError
            Debug.Print errLoop.Description
      Next

      GoTo Done

End Sub
```

Quản lý dữ liệu kiểu lớn – Text, image.

Dữ liệu kiểu text, ntext, image là kiểu dữ liệu phức tạp, việc quản lý, khai thác không được thực hiện thông thường, ADO hỗ trợ các phương thức riêng để thực hiện.

Thay vì độc, cập nhật dữ liệu trực tiếp thì dữ liệu kiểu này được thao tác theo đoạn (chunk) bằng cách sử dụng các phương thức **AppendChunk**, **GetChunk**.

Trước khi thực hiện bạn phải đặt tham số bằng cách thực hiện lệnh sau:

```
EXEC sp_dboption 'pubs', 'Select into/bulkcopy', 'True'
```

Xét ví dụ sau trên CSDL Pubs:

- Copy bảng pub_info sang bảng mới

```
USE pubs
SELECT * INTO pub_info_x
   FROM pub_info
GO
```

- Thực hiện chèn dữ liệu vào bảng:

```
Public Sub AppendChunkX()

   Dim cn As ADODB.Connection
   Dim rstPubInfo As ADODB.Recordset
   Dim strCn As String
   Dim strPubID As String
   Dim strPRInfo As String
   Dim lngOffset As Long
   Dim lngLogoSize As Long
   Dim varLogo As Variant
   Dim varChunk As Variant

   Const conChunkSize = 100

   ' Open a connection.
   Set cn = New ADODB.Connection
   strCn = "Server=srv;Database=pubs;UID=sa;Pwd=;"

   cn.Provider = "sqloledb"
   cn.Open strCn

   'Open the pub_info_x table.
   Set rstPubInfo = New ADODB.Recordset
   rstPubInfo.CursorType = adOpenDynamic
   rstPubInfo.LockType = adLockOptimistic
   rstPubInfo.Open "pub_info_x", cn, , , adCmdTable

   'Prompt for a logo to copy.
   strMsg = "Available logos are : " & vbCr & vbCr

   Do While Not rstPubInfo.EOF
      strMsg = strMsg & rstPubInfo!pub_id & vbCr & _
        Left(rstPubInfo!pr_info,
         InStr(rstPubInfo!pr_info, ",") - 1) & vbCr &
vbCr
      rstPubInfo.MoveNext
   Loop

   strMsg = strMsg & "Enter the ID of a logo to copy:"
   strPubID = InputBox(strMsg)

   ' Copy the logo to a variable in chunks.
   rstPubInfo.Filter = "pub_id = '" & strPubID & "'"
   lngLogoSize = rstPubInfo!logo.ActualSize
   Do While lngOffset < lngLogoSize
```

```
      varChunk = rstPubInfo!logo.GetChunk(conChunkSize)
      varLogo = varLogo & varChunk
      lngOffset = lngOffset + conChunkSize
   Loop

   ' Get data from the user.
   strPubID = Trim(InputBox("Enter a new pub ID:"))
   strPRInfo = Trim(InputBox("Enter descriptive
text:"))

   ' Add a new record, copying the logo in chunks.
   rstPubInfo.AddNew
   rstPubInfo!pub_id = strPubID
   rstPubInfo!pr_info = strPRInfo
   lngOffset = 0   ' Reset offset.

   Do While lngOffset < lngLogoSize
      varChunk = LeftB(RightB(varLogo, lngLogoSize - _
        lngOffset),conChunkSize)
      rstPubInfo!logo.AppendChunk varChunk
      lngOffset = lngOffset + conChunkSize
   Loop

   rstPubInfo.Update

   ' Show the newly added data.
   MsgBox "New record: " & rstPubInfo!pub_id & vbCr & _
     "Description: " & rstPubInfo!pr_info & vbCr & _
     "Logo size: " & rstPubInfo!logo.ActualSize

   rstPubInfo.Close
   cn.Close

End Sub
```

Kết nối từ ASP.

Trong ác ví dụ saiu đây thực hiện làm việc với SQL Server từ ASP, sử dụng ngôn ngữ lập trình VBScript, để làm được ví dụ này bạn đọc phải có kiến thức về thiết kế Web site (HTML).

Thiết kế form kết nối:

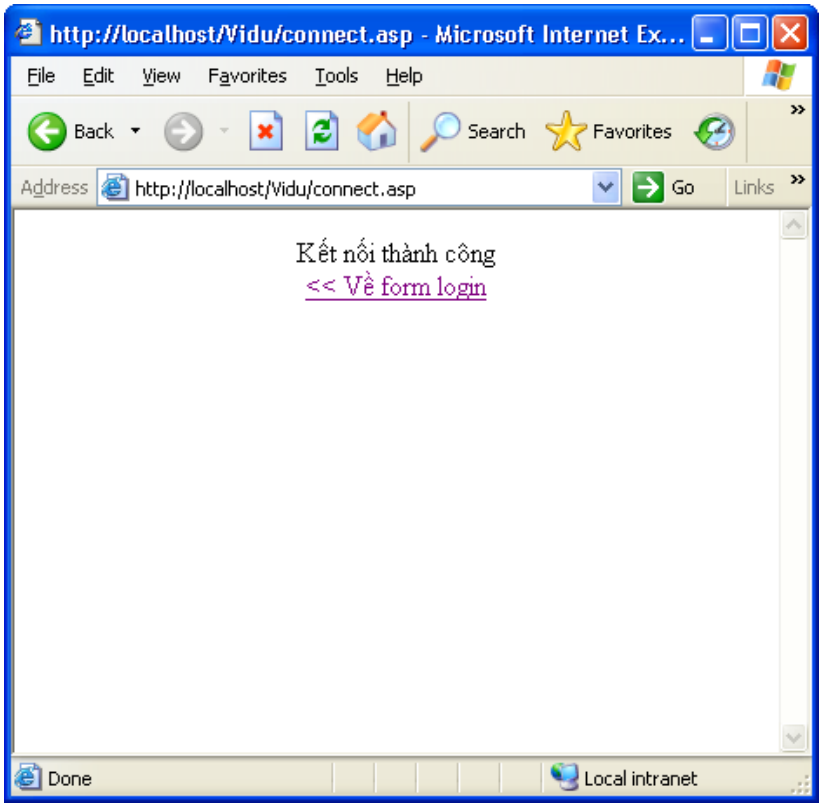
```
<%@LANGUAGE="VBSCRIPT" CODEPAGE="65001"%>
<html>
<head>
<title>Login SQL Server example</title>
<meta   http-equiv="Content-Type"   content="text/html;
charset=utf-8">
<style type="text/css">
<!--
.style1 {
    font-size: 24px;
    font-weight: bold;
}
-->
</style>
</head>

<body>
<p align="center" class="style1">Login SQL Server</p>
<p> </p>
<form name="frmlogin">
<table width="100%" border="0">
  <tr>
        <td  width="40%"><div  align="right">User  name
</div></td>
    <td width="60%">
      <input name="txtUser" type="text" id="txtUser">
    </td>
  </tr>
  <tr>
    <td><div align="right">Password </div></td>
        <td><input  name="txtPassword"  type="password"
id="txtPassword"></td>
  </tr>
  <tr>
    <td><div align="right">Server name </div></td>
            <td><input   name="txtServer"   type="text"
id="txtServer"></td>
  </tr>
</table>
  <div align="center">
     <input name="cmdLogin" type="button" id="cmdLogin"
value="Login">
```

```
     
   <input type="reset" name="cmdReset" value="Reset">
  </div>
</form>
</body>
</html>
<Script language=vbscript>
sub cmdLogin_Onclick
    frmlogin.method="post"
    frmlogin.action="connect.asp"
    frmlogin.submit
end sub
</script>
```

Tập tin connect.asp: Là tập tin được gọi từ form login.asp, thực hiện nhận tham số của form login.asp, kết nối đến SQL Server.

```
<%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>
<%
dim username, password, servername, txt
    username=request.Form("txtUser")
```

```
    password = request.Form("txtPassword")
    servername=request.Form("txtServer")
    txt= "Provider=SQLOLEDB; "
    txt=txt & " Data Source=" & servername & ";"
    txt=txt & " Initial Catalog=pubs; "
    txt=txt & " User ID=" & username & ";"
    txt=txt & " PWD=" & password

    Set cn=Server.CreateObject("ADODB.Connection")
    cn.Open txt
    %>
<div    align="center"><span    class="style1">K&#7871;t
n&#7889;i th&agrave;nh c&ocirc;ng</span>
<%
    cn.close
%>
<br>
<a href="Login.asp<< V&#7873; form login</a>
</div>
```



Liệt kê danh sách.

Để liệt kê danh sách (có thể lấy dữ liệu bằng cách truy vấn trực tiếp hoặc thông qua khung nhìn - view) trước hết phải tạo một recordset lưu trữ kết quả truy vấn, từ recordset ta có thể lấy dữ liệu và đặt vào vị trí tương ứng cần thiết.

+ *Khai báo Recordset*:

```
Set rs=Server.CreateObject("ADODB.Recordset")
rs.ActiveConnection =cn
rs.Source ="Select * from Authors"
rs.Open
```

+ *Lấy giá trị*:

```
Rs.fields(“au_id”)
```

+ *Đóng Recordset*:

```
Rs.close
```

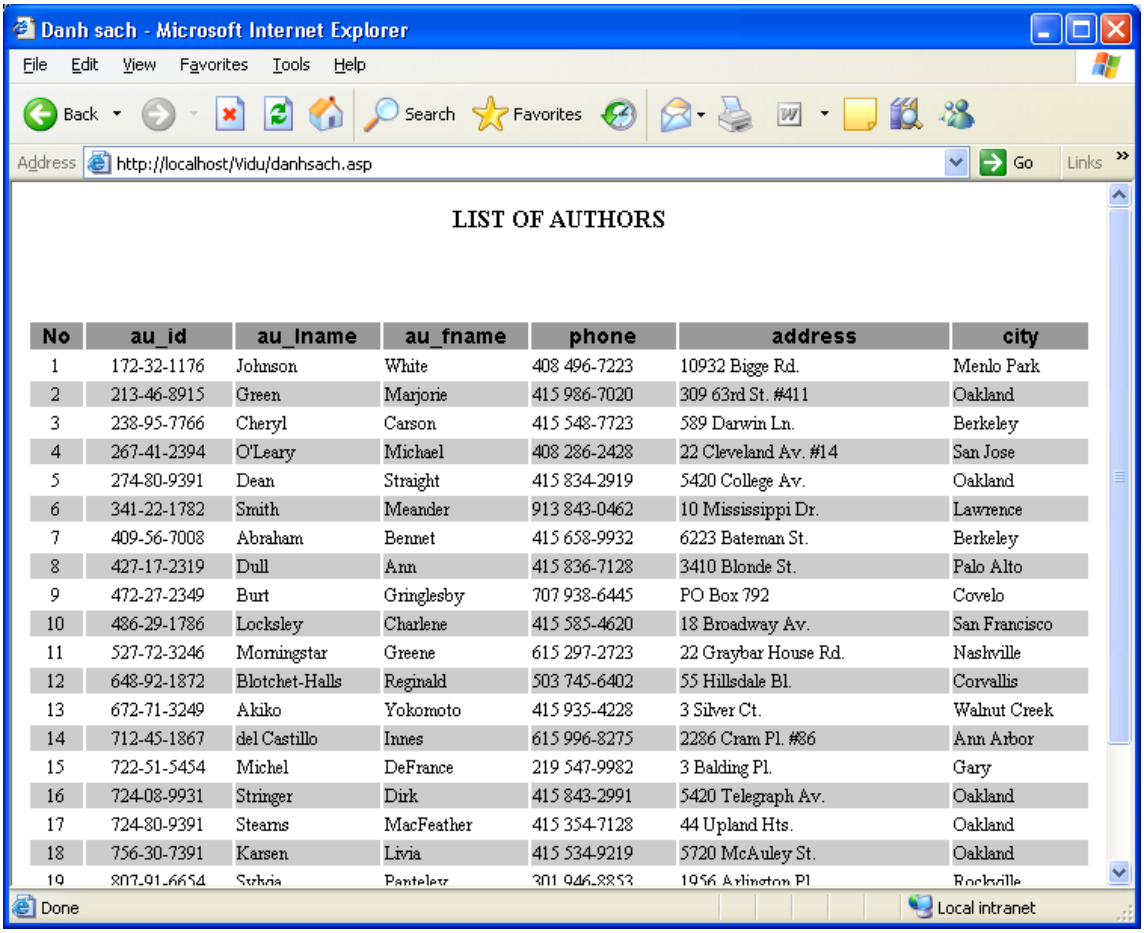
+ *Ví dụ liệt kê danh sách bằng cách truy vấn trực tiếp*:

```
<%@LANGUAGE="VBSCRIPT" CODEPAGE="65001"%>
<html>
<head>
<title>Danh sach</title>
<meta   http-equiv="Content-Type"   content="text/html;
charset=utf-8">
<style type="text/css">
<!--
.style5 {
    font-family: Arial, Helvetica, sans-serif;
    font-weight: bold;
    font-size: 14px;
}
.style6 {font-family: Arial, Helvetica, sans-serif}
.style7 {
    font-size: 14px;
    font-weight: bold;
}
```

```
.style9  {font-family:  Arial,  Helvetica,  sans-serif;
font-size: 14px; }
.style12 {font-size: 12px}
-->
</style>
</head>
<%
dim username, password, servername, txt
    username="sa"
    password = ""
    servername="TDCong"
    txt= "Provider=SQLOLEDB; "
    txt=txt & " Data Source=" & servername & ";"
    txt=txt & " Initial Catalog=pubs; "
    txt=txt & " User ID=" & username & ";"
    txt=txt & " PWD=" & password
    Set cn=Server.CreateObject("ADODB.Connection")
    cn.Open txt

    Set rs=Server.CreateObject("ADODB.Recordset")
    rs.ActiveConnection =cn
    rs.Source ="Select * from Authors"
    rs.Open
%>
<body>
<p align="center"><strong>LIST OF AUTHORS
</strong></p>
<p> </p>
<table width="100%"  border="0.2">
  <tr bgcolor="#999999">
                <td    width="5%"><div    align="center"
class="style5">No</div></td>
     <td width="14%"><div align="center" class="style6
style7">au_id</div></td>
                <td    width="14%"><div    align="center"
class="style9"><strong>au_lname</strong></div></td>
                <td    width="14%"><div    align="center"
class="style9"><strong>au_fname</strong></div></td>
                <td    width="14%"><div    align="center"
class="style9"><strong>phone</strong></div></td>
                <td    width="26%"><div    align="center"
class="style9"><strong>address</strong></div></td>
```

```
                <td    width="13%"><div    align="center"
class="style9"><strong>city</strong></div></td>
  </tr>
  <%
    i=0
  do while not rs.eof and not rs.bof
     i=i+1
    if i mod 2<>0 then
  %>
  <tr bgcolor="#FFFFFF">
     <td><div align="center"><span class="style12"><%=i
%></span></div></td>
       <td><div  align="center"><span  class="style12"><
%=rs.fields("au_id")%></span></div></td>
      <td><span class="style12"><%=rs.fields("au_fname")
%></span></td>
      <td><span class="style12"><%=rs.fields("au_lname")
%></span></td>
     <td><span class="style12"><%=rs.fields("Phone")%></
span></td>
      <td><span  class="style12"><%=rs.fields("Address")
%></span></td>
        <td><span  class="style12"><%=rs.fields("City")
%></span></td>
  </tr>
  <%
   else
  %>
  <tr bgcolor="#CCCCCC">
     <td><div align="center"><span class="style12"><%=i
%></span></div></td>
       <td><div  align="center"><span  class="style12"><
%=rs.fields("au_id")%></span></div></td>
      <td><span class="style12"><%=rs.fields("au_lname")
%></span></td>
      <td><span class="style12"><%=rs.fields("au_fname")
%></span></td>
     <td><span class="style12"><%=rs.fields("Phone")%></
span></td>
      <td><span  class="style12"><%=rs.fields("Address")
%></span></td>
```

```
          <td><span  class="style12"><%=rs.fields("City")
%></span></td>
  </tr>
  <%
    end if
    rs.movenext
  loop
  %>
</table>
</body>
<%
    rs.close
%>
</html>
```



No	au_id	au_lname	au_fname	phone	address	city
1	172-32-1176	Johnson	White	408 496-7223	10932 Bigge Rd.	Menlo Park
2	213-46-8915	Green	Marjorie	415 986-7020	309 63rd St. #411	Oakland
3	238-95-7766	Cheryl	Carson	415 548-7723	589 Darwin Ln.	Berkeley
4	267-41-2394	O'Leary	Michael	408 286-2428	22 Cleveland Av. #14	San Jose
5	274-80-9391	Dean	Straight	415 834-2919	5420 College Av.	Oakland
6	341-22-1782	Smith	Meander	913 843-0462	10 Mississippi Dr.	Lawrence
7	409-56-7008	Abraham	Bennet	415 658-9932	6223 Bateman St.	Berkeley
8	427-17-2319	Dull	Ann	415 836-7128	3410 Blonde St.	Palo Alto
9	472-27-2349	Burt	Gringlesby	707 938-6445	PO Box 792	Covelo
10	486-29-1786	Locksley	Charlene	415 585-4620	18 Broadway Av.	San Francisco
11	527-72-3246	Morningstar	Greene	615 297-2723	22 Graybar House Rd.	Nashville
12	648-92-1872	Blotchet-Halls	Reginald	503 745-6402	55 Hillsdale Bl.	Corvallis
13	672-71-3249	Akiko	Yokomoto	415 935-4228	3 Silver Ct.	Walnut Creek
14	712-45-1867	del Castillo	Innes	615 996-8275	2286 Cram Pl. #86	Ann Arbor
15	722-51-5454	Michel	DeFrance	219 547-9982	3 Balding Pl.	Gary
16	724-08-9931	Stringer	Dirk	415 843-2991	5420 Telegraph Av.	Oakland
17	724-80-9391	Stearns	MacFeather	415 354-7128	44 Upland Hts.	Oakland
18	756-30-7391	Karsen	Livia	415 534-9219	5720 McAuley St.	Oakland
19	807-91-6654	Sylvia	Panteley	301 946-8853	1956 Arlington Pl	Rockville

KẾT NỐI VỚI SQL SERVER BẰNG SQL-DMO.

SQL DMO viết tắt của cụm từ SQL Distributed Management Objects, sử dụng thư viện liên kết động (dll) để kết nối đến SQL Server.

SQL DMO thực hiện liên kết nhúng (OLE Automation), các đối tượng SQL Server được thực hiện nhúng các đối tượng của SQL Server vào ứng dụng, khai thác các đối tượng thông qua thuộc tính, sự kiện và các phương thức làm việc của nó.

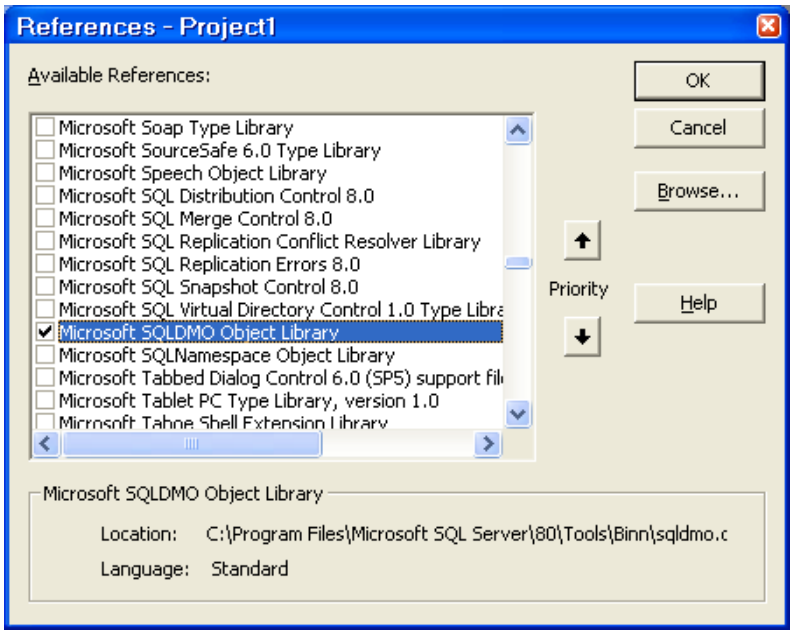
SQL DMO hỗ trợ phát triển ứng dụng từ ngôn ngữ lập trình Visual Basic, C++, khi đóng gói các thư viện liên kết động sẽ được đóng gói cùng, cài dặt ứng dụng thư viện sẽ được cài đặt trong Windows, nên khi chạy ứng dụng bạn không cần thiết lập môi trường Client Connectivity.

Các tập tin cơ bản cho SQL DMO: sqldmo.dll, sqldmo80.hlp, sqldmo.rll, sqldmo.h (C++), sqldmoid.h (C++), sqldmo.sql. Trong phần này sẽ giới thiệu kỹ thuật thiết kế ứng dụng từ Visual Basic 6.0.

Khai báo thư viện trong project.

- Vào menu Project -> References

- Chọn Microsoft SQL DMO Object Library -> Ok



Khai báo đối tượng.

Sau khi thực hiện khai báo thư viện trong project, ta có thể khai báo biến kiểu đối tượng (object) hoặc kiểu đối tượng của SQL DMO.

Ví dụ khai báo biến kiểu SQL Server:

```
Dim oSQLServer As SQLDMO.SQLServer
```

Kết nối đến SQL Server.

Để kết nối đến SQL Server ta sử dụng phương thức kết nối của đối tượng SQL Server, có 3 tham số Servername, LoginName, Pasword.

```
Dim oSQLServer As SQLDMO.SQLServer
    Set oSQLServer = New SQLDMO.SQLServer
        oSQLServer.Connect  "ServerName",  "LoginName",
"Pasword"
```

Thực hiện lại kết nối:

Trong nhiều trường hợp bạn muốn ngắt kết nối hiện tại và thực hiện lại kết nối để lấy trạng thái SQL Server hiện thời (tương tự động tác làm tươi – Reresh).

```
    oSQLServer.DisConnect

    oSQLServer.ReConnect
```

Làm việc với các đối tượng.

SQL DMO tạo đối tượng kế thừa từ những đối tượng con của nó, ví dụ SQL Server kế thừa từ các đối tượng Database <- Table <- Column,…

Xác định biến với CSDL:

```
Dim oDatabase as new SQLDMO.Database
Set oDatabase = oSQLServer.Databases("Northwind")
```

Lấy danh sách tên các CSDL vào hộp thoại:

```
Dim nDatabase as Integer
For nDatabase = 1 to oSQLServer.Databases.Count
    Combo1.AddItem oSQLServer.Databases(nDatabase).Name
```

```
Next nDatabase
```

Các đối tượng đều được kế thà từ các đối tượng con, các đối tượng con tạo thành một tập hợp, tập hợp nói trên có thể thực hiện các phương thức Add, Remove,…với từng đối tượng.

Ví dụ remove bảng khỏi CSDL:

```
oServer.Databases("Northwind").Tables.Remove("Orders",
"anne")
```

Thực hiện lệnh SQL:

Các đối tượng (SQL Server, Database) có thể thực hiện các lệnh SQL thông qua các phương thức ExecuteImmediate và ExecuteWithResults.

Ví dụ thực hiện lệnh thao tác:

```
oSQLServer.ExecuteImmediate “Create Database Example”
```

Ví dụ thực hiện lệnh truy vấn:

```
Dim rs As QueryResults
      Set  rs  =  oDatabase.ExecuteWithResults("Select  *
from Authors")
```

Ví dụ lấy dữ liệu từ một truy vấn:

```
    For i = 1 To rs.Rows
        For j = 1 To rs.Columns
            MsgBox rs.GetColumnString(i, j)
        Next j
    Next i
```

Các phương thức thực hiện kết nối có thể hỗ trợ theo từng ngôn ngữ lập trình, hỗ trợ nhiều trong việc lập trình từ Visual Basic, ASP, C, C++. Bạn có thể tham khảo các ví dụ trong thư mục Sample của SQL Server. Các ví dụ sẽ đề cập nhiều ngôn ngữ lập trình, nhiều sự kiện khác nhau.