

HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU – SQL SERVER

Nguyễn Thị Tuyết – Khoa CNTT
Đại học Quy Nhơn
tuyetdhqn@gmail.com

NỘI DUNG

- Bài 1: Tổng quan về SQL Server
- Bài 2: Bảng dữ liệu - Table
- Bài 3: Truy vấn dữ liệu
- Bài 4: Câu lệnh hợp
- Bài 5: Khung nhìn – View
- Bài 6: Thủ tục thường trú
- Bài 7: Hàm do người dùng định nghĩa
- Bài 8: Trigger
- Bài 9: Bảo mật và phân quyền người dùng
- Bài 10: Nhập và Xuất CSDL.
- Bài 11: Sao lưu CSDL

Thời lượng & Tài liệu tham khảo

- 45 tiết: 30 tiết lý thuyết+30tiết thực hành
 - Thi kiểm tra trên máy.
 - Giáo trình SQL Server - DHQN
 - Tập bài giảng của GV
-

Bài 1: Tổng quan về SQL Server

1. Giới thiệu Hệ QTCSDL-SQL Server
 2. Đặc tính của SQL Server
 3. Các chức năng của Hệ QTCSDL
 4. Hoạt động của một Hệ QLCSDL
 5. Vai trò của con người
 6. Các bước xây dựng CSDL
 7. Các thành phần của SQL Server
 8. Quản trị SQL Server
-

1. Giới thiệu Hệ QTCSDL-SQL Server

a. Hệ QTCSDL quan hệ là gì?

Mỗi cơ sở dữ liệu quan hệ là một tập hợp dữ liệu được tổ chức trong những bảng hai chiều có quan hệ với nhau.

Một RDBMS có nhiệm vụ:

- Lưu trữ và tạo dữ liệu sẵn có trong các bảng.
- Duy trì quan hệ giữa các bảng trong cơ sở dữ liệu.
- Bảo đảm tích hợp dữ liệu bằng cách tạo các qui tắc quản lý giá trị dữ liệu.
- Khôi phục mọi dữ liệu trong trường hợp hệ thống có sự cố.

1. Giới thiệu Hệ QTCSDL-SQL Server

b. SQL (*Struct query language*) Server là hệ thống quản trị cơ sở dữ liệu quan hệ (Relational DataBase Management System- RDBMS) sử dụng các lệnh SQL để trao đổi dữ liệu giữa Client và Server.

2. Đặc tính của SQL Server

- Cho phép quản trị một hệ CSDL lớn (vài tera byte), có tốc độ xử lý dữ liệu nhanh.
- Cho phép nhiều người cùng khai thác trong một thời điểm đối với một CSDL và toàn bộ quản trị CSDL (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).

3. CÁC CHỨC NĂNG CỦA HỆ QUẢN TRỊ CSDL

Một hệ QT CSDL có các chức năng cơ bản sau:

a) Cung cấp môi trường tạo lập CSDL:

Thông qua ngôn ngữ định nghĩa dữ liệu, người dùng khai báo kiểu và các cấu trúc dữ liệu thể hiện thông tin, khai báo các ràng buộc trên dữ liệu được lưu trữ trong CSDL.

3. CÁC CHỨC NĂNG CỦA HỆ QTCSDL

b) Cung cấp môi trường cập nhật và khai thác dữ liệu

Ngôn ngữ để người dùng diễn tả yêu cầu cập nhật hay tìm kiếm, kết xuất thông tin gọi là ngôn ngữ thao tác dữ liệu.

Gồm:

- Cập nhật: Nhập, sửa, xóa dữ liệu;
- Khai thác: Tìm kiếm và kết xuất dữ liệu.

Ngôn ngữ được dùng phổ biến là SQL (Structured Query Language).

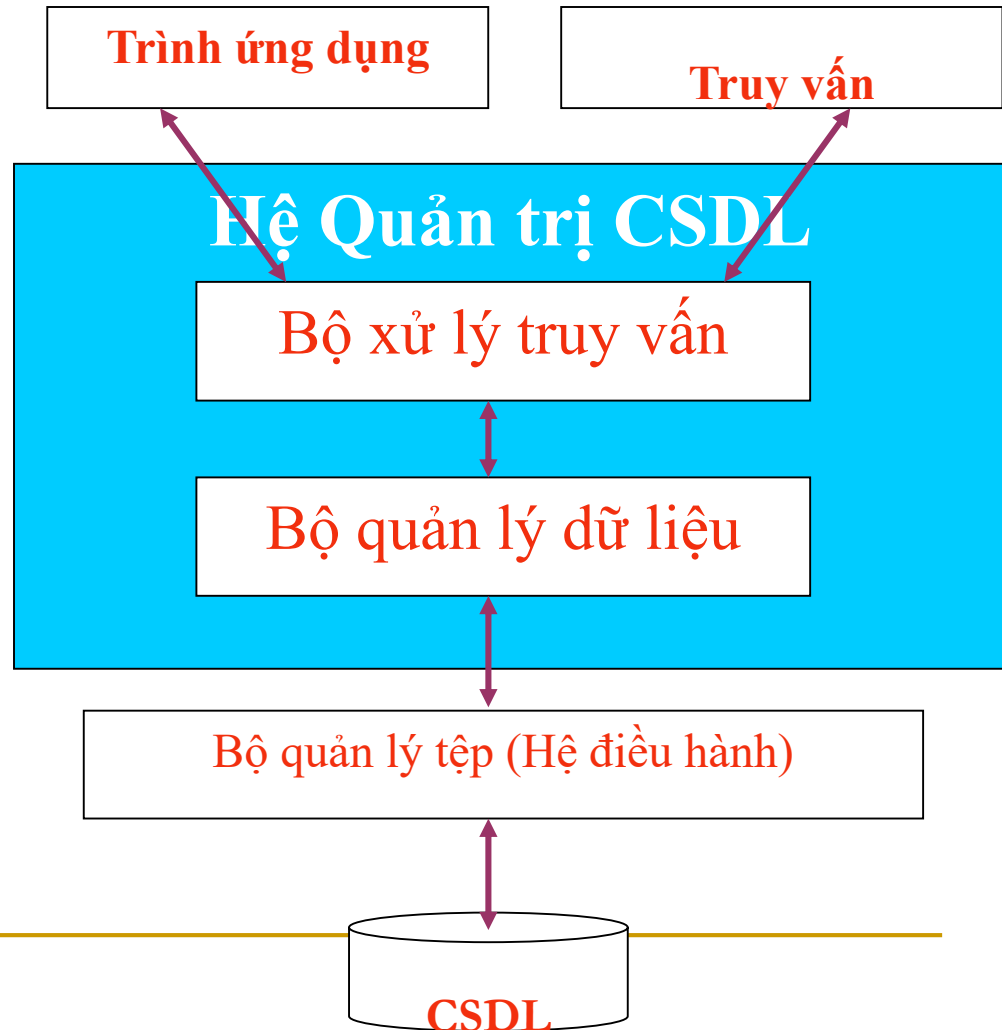
3. CÁC CHỨC NĂNG CỦA HỆ QTCSDL

c) Cung cấp công cụ kiểm soát, điều khiển việc truy cập vào CSDL:

- Phát hiện và ngăn chặn sự truy cập không được phép;
 - Duy trì tính nhất quán của dữ liệu;
 - Tổ chức và điều khiển các truy cập đồng thời;
 - Khôi phục CSDL khi có sự cố phần cứng hay phần mềm;
-

4. HOẠT ĐỘNG CỦA MỘT HỆ QUẢN TRỊ CSDL

Một hệ QTCSDL là một phần mềm phức tạp gồm nhiều thành phần (môđun), mỗi thành phần có chức năng cụ thể, trong đó có 2 thành phần chính là bộ xử lý truy vấn và bộ quản lý dữ liệu. Một số chức năng của hệ QTCSDL được hỗ trợ bởi hệ điều hành.



5. VAI TRÒ CỦA CON NGƯỜI KHI LÀM VIỆC VỚI HỆ QTCSDL

Có ba vai trò của con người liên quan đến hoạt động của hệ QTCSDL:

- + Người quản trị CSDL
 - + Người lập trình ứng dụng
 - + Người dùng
-

5. VAI TRÒ CỦA CON NGƯỜI KHI LÀM VIỆC VỚI HỆ QTCSDL

a. Người quản trị CSDL

Là một hay một nhóm người được trao quyền điều hành hệ CSDL

- + Quản lí các tài nguyên của CSDL;
- + Tổ chức hệ thống: phân quyền truy cập cho người dùng, đảm bảo an ninh cho hệ CSDL;
- + Nâng cấp hệ CSDL: bổ sung, sửa đổi để cải tiến chế độ khai thác, nâng cao hiệu quả sử dụng;
- + Bảo trì CSDL: thực hiện các công việc bảo vệ và khôi phục hệ CSDL.

5. VAI TRÒ CỦA CON NGƯỜI KHI LÀM VIỆC VỚI HỆ QTCSDL

b. Người lập trình ứng dụng

Là người có nhiệm vụ xây dựng các chương trình ứng dụng hỗ trợ khai thác thông tin từ CSDL trên cơ sở các công cụ mà hệ quản trị CSDL cung cấp.

c. Người dùng

Là người có nhu cầu khai thác thông tin từ CSDL. Người dùng thường được phân thành từng nhóm, mỗi nhóm có một quyền hạn nhất định để truy cập và khai thác CSDL

6. CÁC BƯỚC XÂY DỰNG CSDL

Bước 1: Khảo sát

- + Tìm hiểu các yêu cầu của công tác quản lí.
 - + Xác định và phân tích mối liên hệ các dữ liệu cần lưu trữ.
 - + Phân tích các chức năng cần có của hệ thống khai thác thông tin, đáp ứng các yêu cầu đặt ra.
 - + Xác định khả năng phần cứng, phần mềm có thể khai thác, sử dụng.
-

6. CÁC BƯỚC XÂY DỰNG CSDL

Bước 2: Thiết kế

- + Thiết kế CSDL.
- + Lựa chọn hệ quản trị để triển khai.
- + Xây dựng hệ thống chương trình ứng dụng.

Bước 3: Kiểm thử

- + Nhập dữ liệu cho CSDL.
 - + Tiến hành chạy thử các chương trình ứng dụng.
-

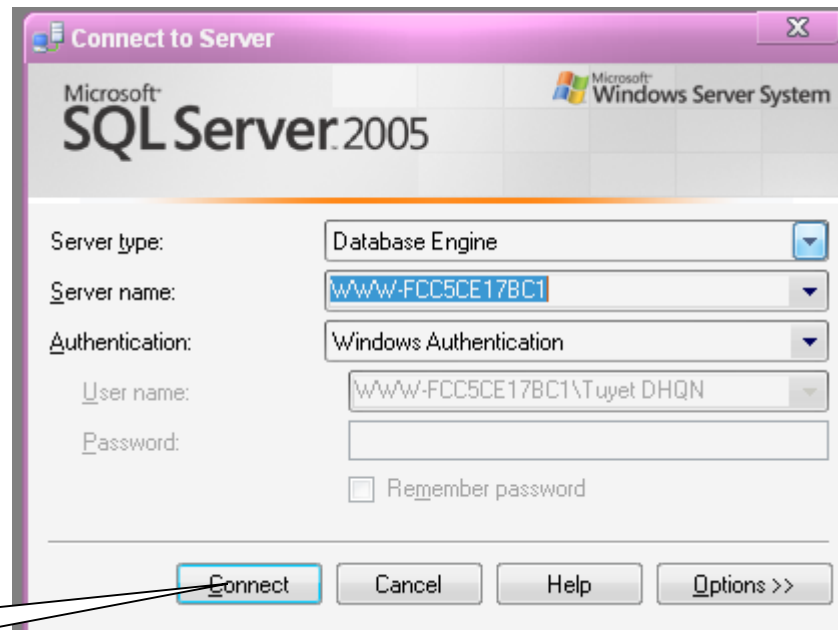
7. Các thành phần của SQL server

- Database : CSDL của SQL server
- Tập tin log : tập tin lưu trữ những chuyển tác của SQL
- Tables : bảng dữ liệu
- Filegroups : tập tin nhóm
- Diagrams : sơ đồ quan hệ
- Views : khung nhìn (hay bảng ảo) số liệu dự trên bảng
- Stored procedure : thủ tục và hàm nội
- User defined function : hàm do người dùng định nghĩa
- Users : người sử dụng cơ sở dữ liệu
- Roles : các quy định vai trò và chức năng trong HT
- Rules : những quy tắc
- Defaults : các giá trị mặc nhiên
- User-defined data types: kiểu dữ liệu do người dùng tự định nghĩa

8. Quản trị SQL Server

a. Khởi động

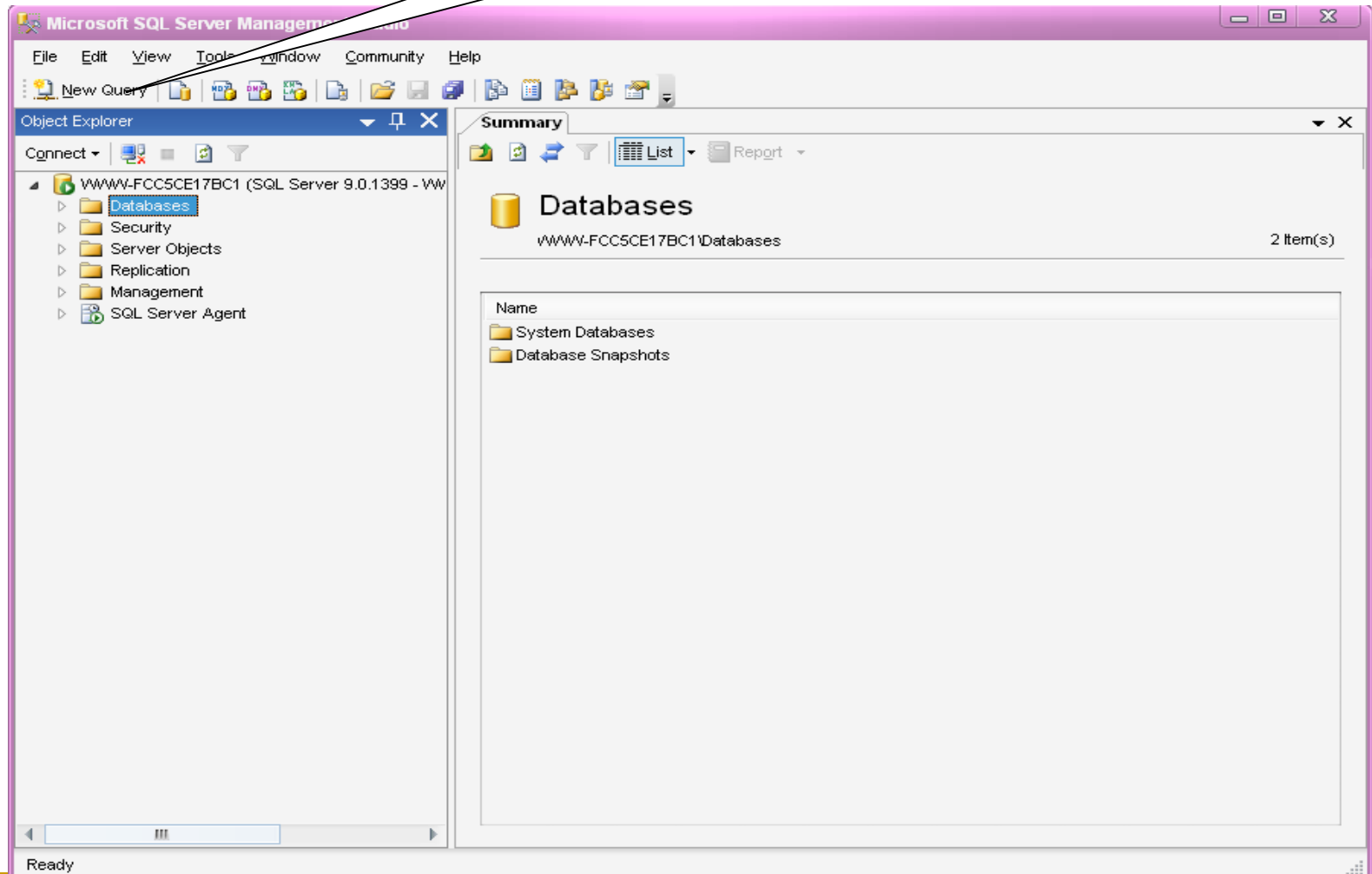
- ❑ Start → Programs → Microsoft SQL Server 2008 → SQL Server Management Studio



Chọn

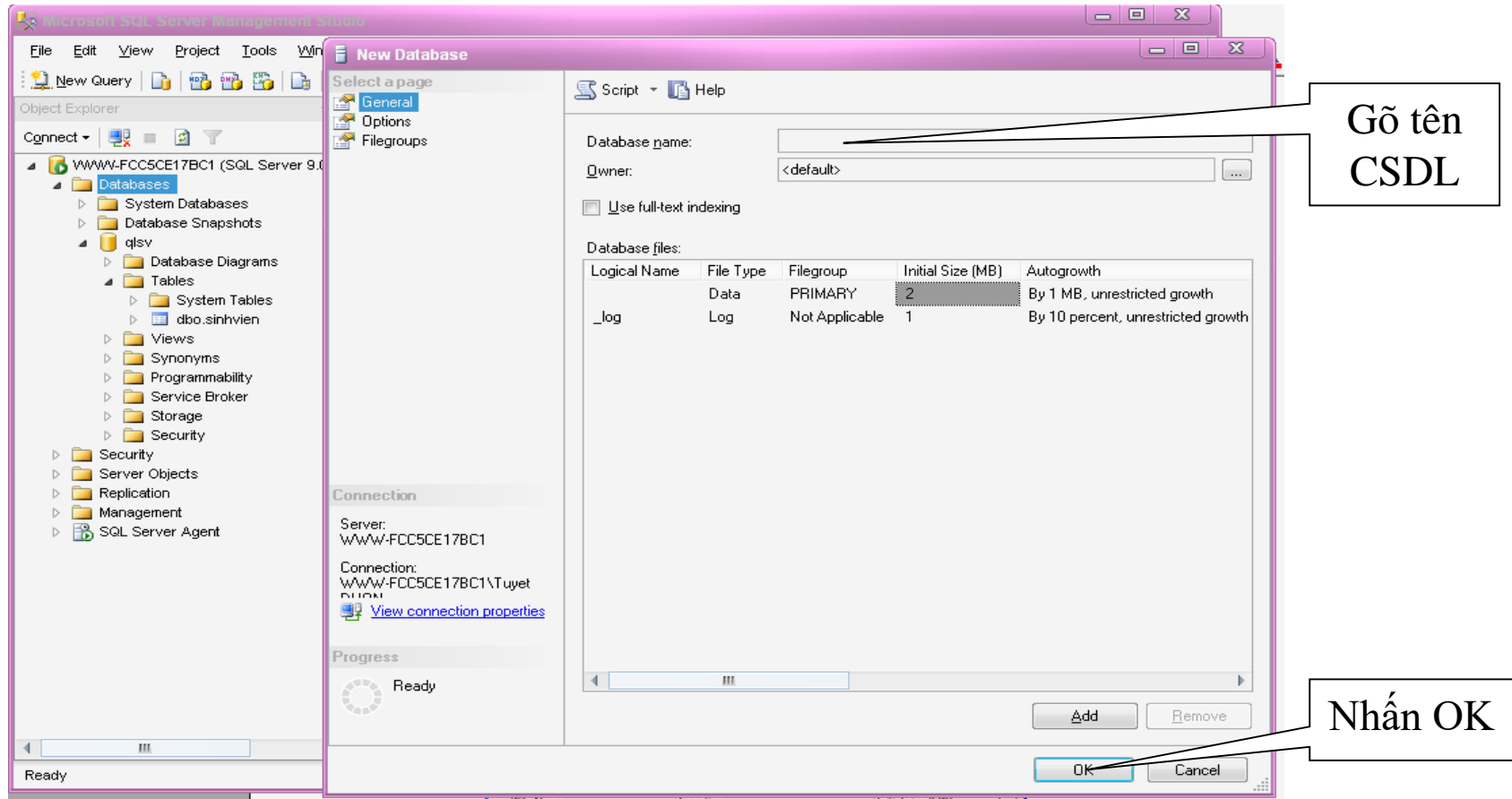
a. Khởi động

Cửa sổ lệnh (nhập chọn)

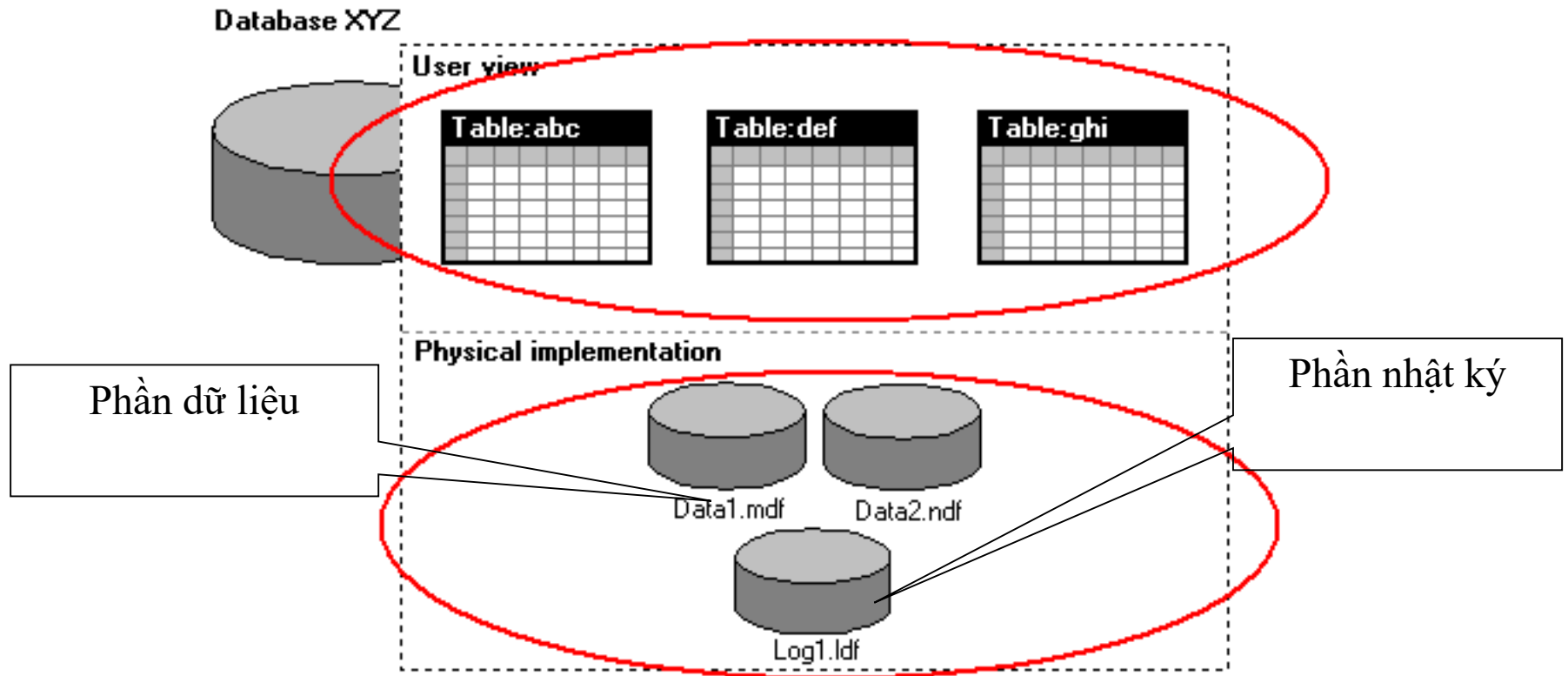


b. Tạo CSDL

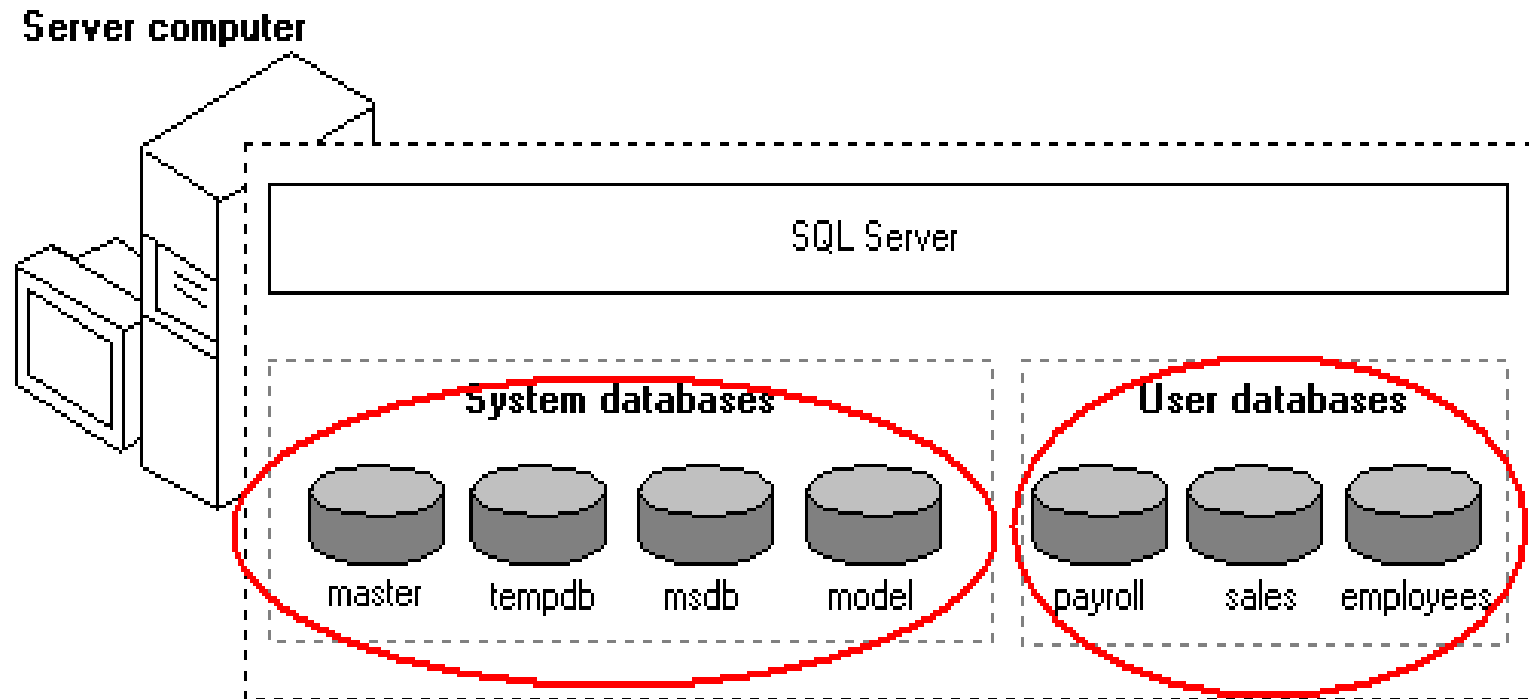
Chuột phải tại Database → New Database



c. Quản trị CSDL



c. Quản trị CSDL



c. Quản trị CSDL: CSDL 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.
-

d. Tạo CSDL

- Tạo CSDL: trước khi tạo CSDL ta phải thực hiện phân tích các thông tin liên quan mục đích sử dụng CSDL cho bài toán của mình: Tên CSDL, các table, ràng buộc,... tuân theo các chuẩn CSDL.
- Tạo CSDL có 2 cách: theo công cụ wizard và câu lệnh SQL: ***CREATE DATABASE tencsdl***
 - Một câu lệnh có thể viết trên nhiều dòng, nhiều câu lệnh có thể viết trên một dòng.
 - Từ khóa: màu xanh dương, chú thích: xanh lá, hàm: hồng,....
 - Thực hiện câu lệnh bằng cách bôi đen rồi nhấn F5,
 - chú ý câu lệnh đang thực hiện trên Database nào.

d. Tạo CSDL (TT)

```
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 )
```

Bài 2: BẢNG DỮ LIỆU – TABLE

1. Chuẩn trong CSDL
2. Thiết kế bảng dữ liệu
3. Kiểu dữ liệu
4. Ràng buộc dữ liệu
5. Tạo bảng dữ liệu
6. Sửa, xóa cấu trúc bảng
7. Nhập dữ liệu vào bảng
8. Cập nhật dữ liệu
9. Xóa dữ liệu
10. Các hàm thông dụng trong SQL

1. Chuẩn trong CSDL

Chuẩn 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.

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

Ví dụ: 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 nêu 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

1. Chuẩn trong CSDL (tiếp)

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

1. Chuẩn trong CSDL (tiếp)

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.

Id_sach	Id_Sinhvien	Ngày mượn	Số lượng đã mượn

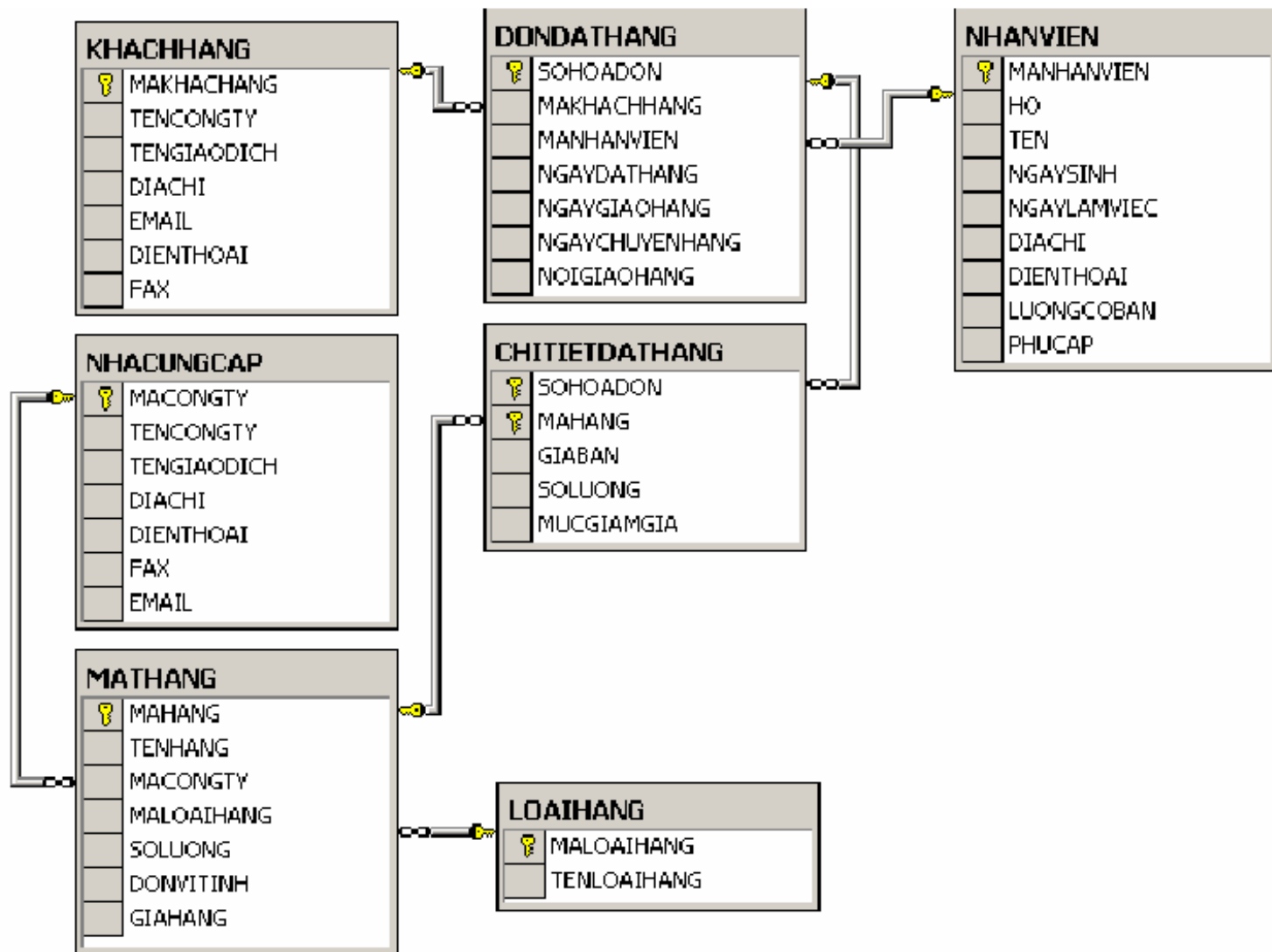
2. THIẾT KẾ BẢNG DỮ LIỆU.

- 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,...

2. THIẾT KẾ BẢNG DỮ LIỆU.

Trước khi bắt tay vào thiết lập bảng dữ liệu ta phải xác định 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.



3. Kiểu dữ liệu

Binary: dữ liệu dạng số ở hệ hexa, gồm Binary, Varbinary, Image.

Text: kiểu ký tự, chứa chữ cái, ký hiệu, số, gồm:

- 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.

3. Kiểu dữ liệu (tiếp)

Date/Time: Kiểu dữ liệu ngày, thời gian, ngày và thời gian:

- Date: Ngày
- Time: Giờ
- DateTime: Đầy đủ cả ngày và thời gian.
- SmallDateTime: .

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

- Int, smallint, tinyint, bigint: Số nguyên
- Float, real, decimal, numeric: Số thực.
- số thứ tự: <kiểu> identity(1,1)

Monetary: Tiền tệ:

- Money, Smallmoney.

Bit: Kiểu số 0, 1.

LOAIHANG			
	Column Name	Data Type	Allow Nulls
🔑	MaLoaiHang	char(2)	<input type="checkbox"/>
	TenLoaiHang	nvarchar(30)	<input checked="" type="checkbox"/>

NHACUNGCAP			
	Column Name	Data Type	Allow Nulls
🔑	MaCongTy	char(3)	<input type="checkbox"/>
	TenCongTy	nvarchar(50)	<input checked="" type="checkbox"/>
	TenGiaoDich	nvarchar(20)	<input checked="" type="checkbox"/>
	DiaChi	nvarchar(50)	<input checked="" type="checkbox"/>
	DienThoai	varchar(15)	<input checked="" type="checkbox"/>
	Fax	varchar(15)	<input checked="" type="checkbox"/>
	Email	varchar(30)	<input checked="" type="checkbox"/>

MATHANG			
	Column Name	Data Type	Allow Nulls
🔑	MaHang	char(4)	<input type="checkbox"/>
	TenHang	nvarchar(30)	<input checked="" type="checkbox"/>
	MaCongTy	char(3)	<input checked="" type="checkbox"/>
	MaLoaiHang	char(2)	<input checked="" type="checkbox"/>
	SoLuong	int	<input checked="" type="checkbox"/>
	DonViTinh	nvarchar(10)	<input checked="" type="checkbox"/>
	GiaHang	numeric(10, 2)	<input checked="" type="checkbox"/>



4. 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.

4. Ràng buộc dữ liệu

■ Toàn vẹn dữ liệu

Kiểu toàn vẹn	Công cụ trong SQL Server
Entry integrity (Toàn vẹn thực thể)	Ràng buộc Primary key Ràng buộc Unique Cột Identity
Domain integrity (Toàn vẹn theo miền)	Giá trị ngầm định Default Ràng buộc khóa ngoài Foreign Key Ràng buộc Check Thuộc tính NOT NULL
Referential integrity (Toàn vẹn tham chiếu)	Ràng buộc Foreign Key Ràng buộc Check
User-defined integrity (Toàn vẹn do người dùng định nghĩa)	Rules Stored procedures Triggers

4. Ràng buộc dữ liệu (tiếp)

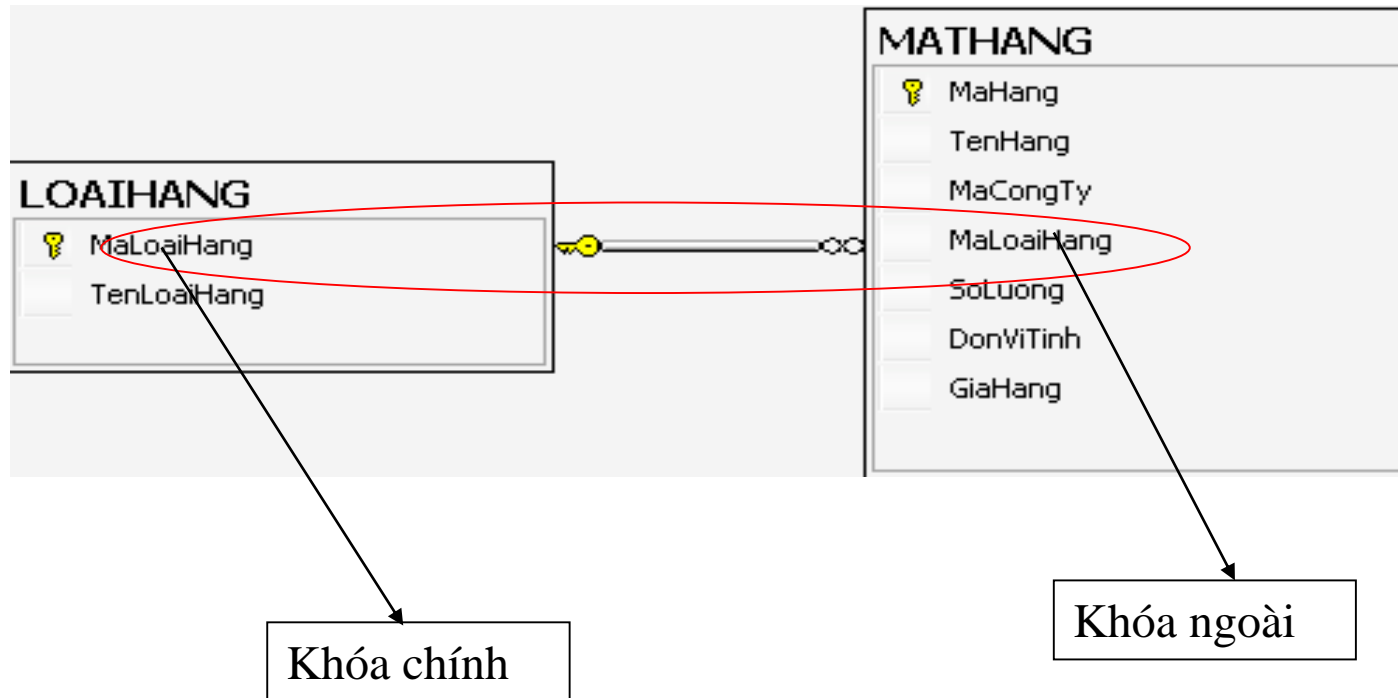
- *Khóa chính – Primary Key.*

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

- *Khóa ngoài.*

- Cột dữ liệu là khóa ngoài nếu có quan hệ với khóa chính ở bảng khác, 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 tên, cùng kiểu dữ liệu, cùng kích thước.
- Bảng bị tham chiếu phải được định nghĩa trước.

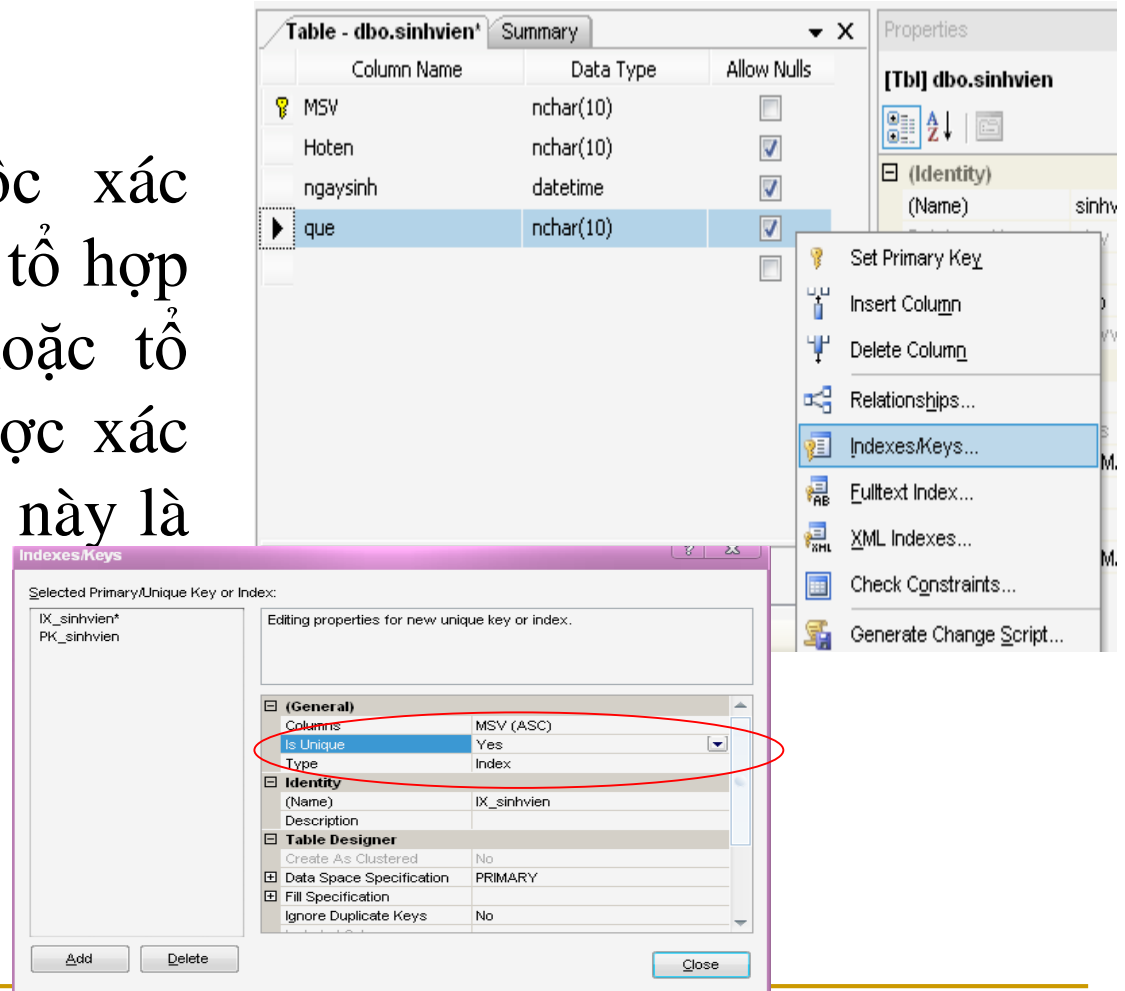
4. Ràng buộc dữ liệu (tiếp)



4. Ràng buộc dữ liệu (tiếp)

■ *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.

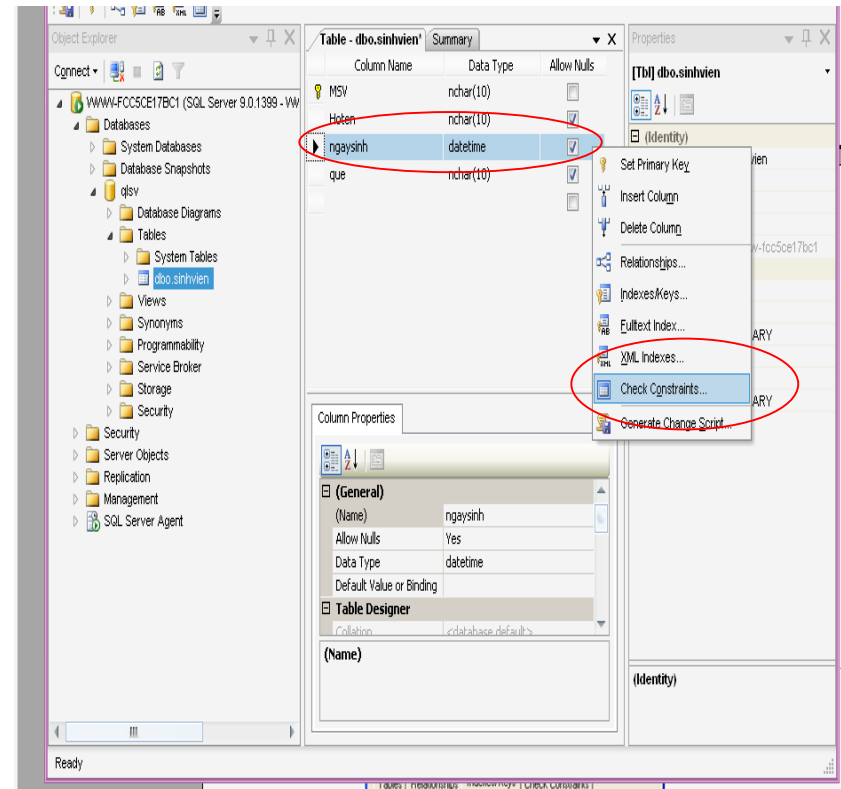


4. Ràng buộc dữ liệu (tiếp)

■ *Ràng buộc Check.*

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

Nhấn chuột phải → Chọn Check Constraints



4. Ràng buộc dữ liệu (tiếp)

Tạo ràng buộc Check bằng công cụ

Check Constraints

Selected Check Constraint:
CK_sinhvien*

Editing properties for new check constraint. The 'Expression' property needs to be filled in before the new check constraint will be accepted.

(General)
Expression

Identity
(Name) CK_sinhvien
Description

Table Designer
Check Existing Data On Creation Yes
Enforce For INSERTs And UPDATEs Yes
Enforce For Replication

Thêm, xóa ràng buộc

Add Delete

sinhvien

sinhvien	qlsv
dbo	www-fcc5ce17bc1
signer	
umn	Yes
ta Space	PRIMARY
No	
Column	
Filegroup	PRIMARY

Nhấn vào dấu ...

Check Constraint Expression

Expression:

Gõ biểu thức điều kiện ràng buộc

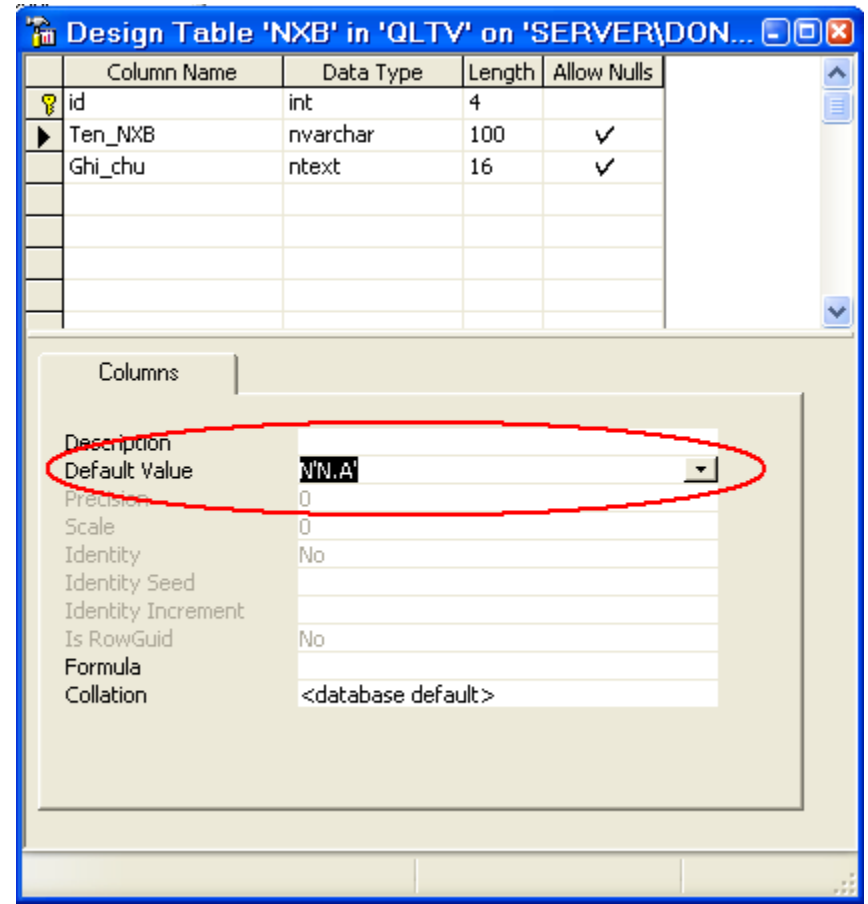
OK Cancel

Table Designer
Collation <default>
(Name)

4. Ràng buộc dữ liệu (tiếp)

- *Giá trị ngầm định – Default.*

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



5. TẠO BẢNG DỮ LIỆU

- Bảng mới được tạo ra sử dụng với mục đích gì và có vai trò như thế nào trong cơ sở dữ liệu.
- Cấu trúc của bảng bao gồm những trường (cột) nào, mỗi một trường có ý nghĩa như thế nào trong việc biểu diễn dữ liệu, kiểu dữ liệu của mỗi trường là gì và trường đó có cho phép nhận giá trị NULL hay không.
- Những trường nào sẽ tham gia vào khóa chính của bảng. Bảng có quan hệ với những bảng khác hay không và nếu có thì quan hệ như thế nào.
- Trên các trường của bảng có tồn tại những ràng buộc về khuôn dạng, điều kiện hợp lệ của dữ liệu hay không; nếu có thì sử dụng ở đâu và như thế nào.

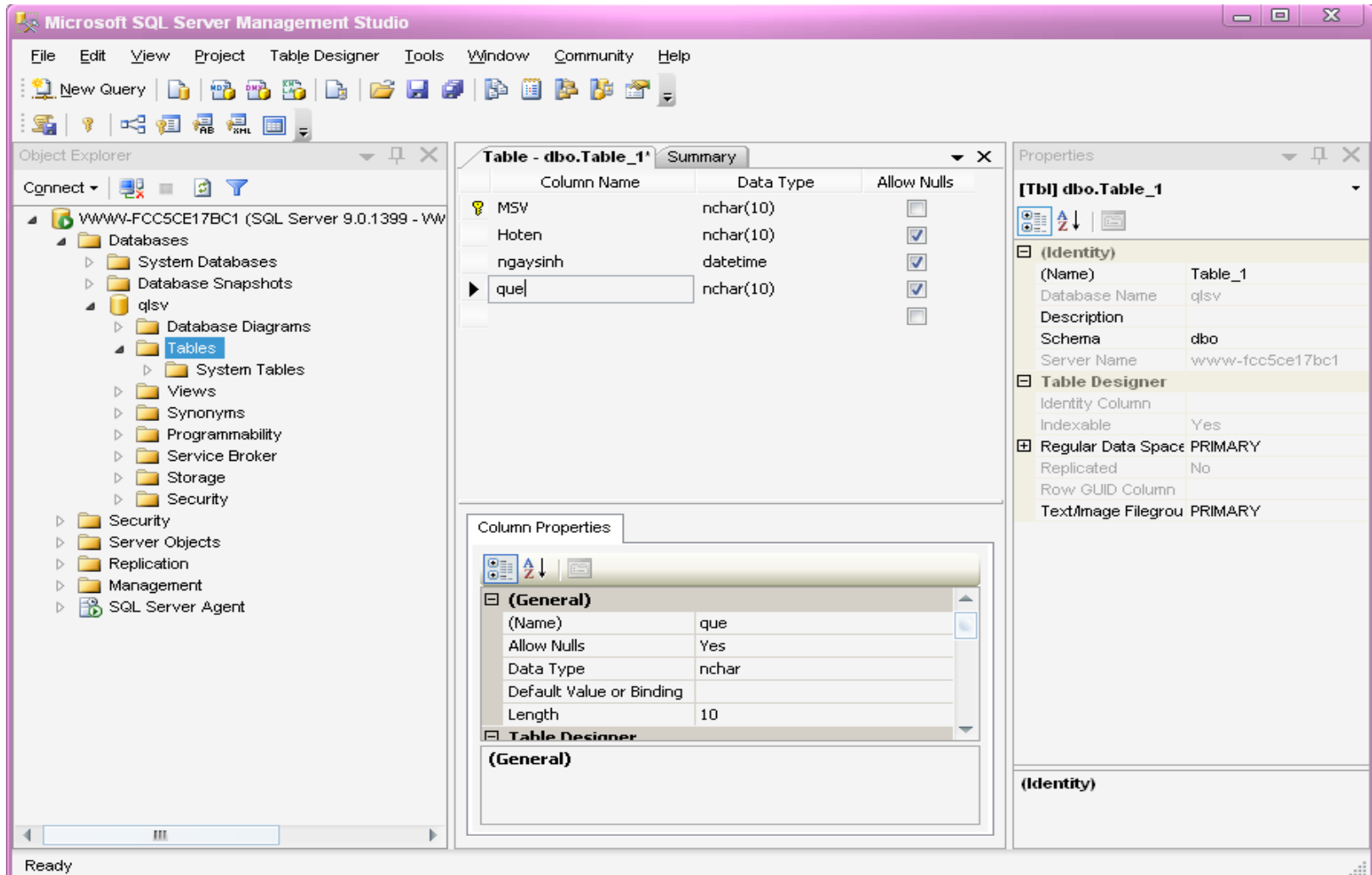
5. TẠO BẢNG DỮ LIỆU (tiếp)

■ Tạo bảng 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 nhấn chuột -> nhấn chuột phải -> chọn Set primary key.

5. Tạo bảng dữ liệu.



5. TẠO BẢNG DỮ LIỆU

■ Tạo bảng bằng câu lệnh

CREATE TABLE *tên_bảng*

(*tên_cột kieu_du_lieu các_ràng_buộc*
 [,...,*tên_cột kieu_du_lieu các_ràng_buộc*])

Ví dụ: Câu lệnh dưới đây định nghĩa bảng NHANVIEN với các trường MANV(mã nhân viên), HOTEN (họ và tên), NGAYSINH (ngày sinh của nhân viên), DIENTHOAI (điện thoại) và HSLUONG (hệ số lương)

5. TẠO BẢNG DỮ LIỆU

■ Tạo bảng bằng câu lệnh

CREATE TABLE nhanvien

(manv	NVARCHAR(10)	NOT NULL,
Hoten	NVARCHAR(50)	NOT NULL,
Ngaysinh	DATETIME	NULL,
Dienthoai	NVARCHAR(10)	NULL,
Hsluong	DECIMAL(3,2)	DEFAULT(2.16))

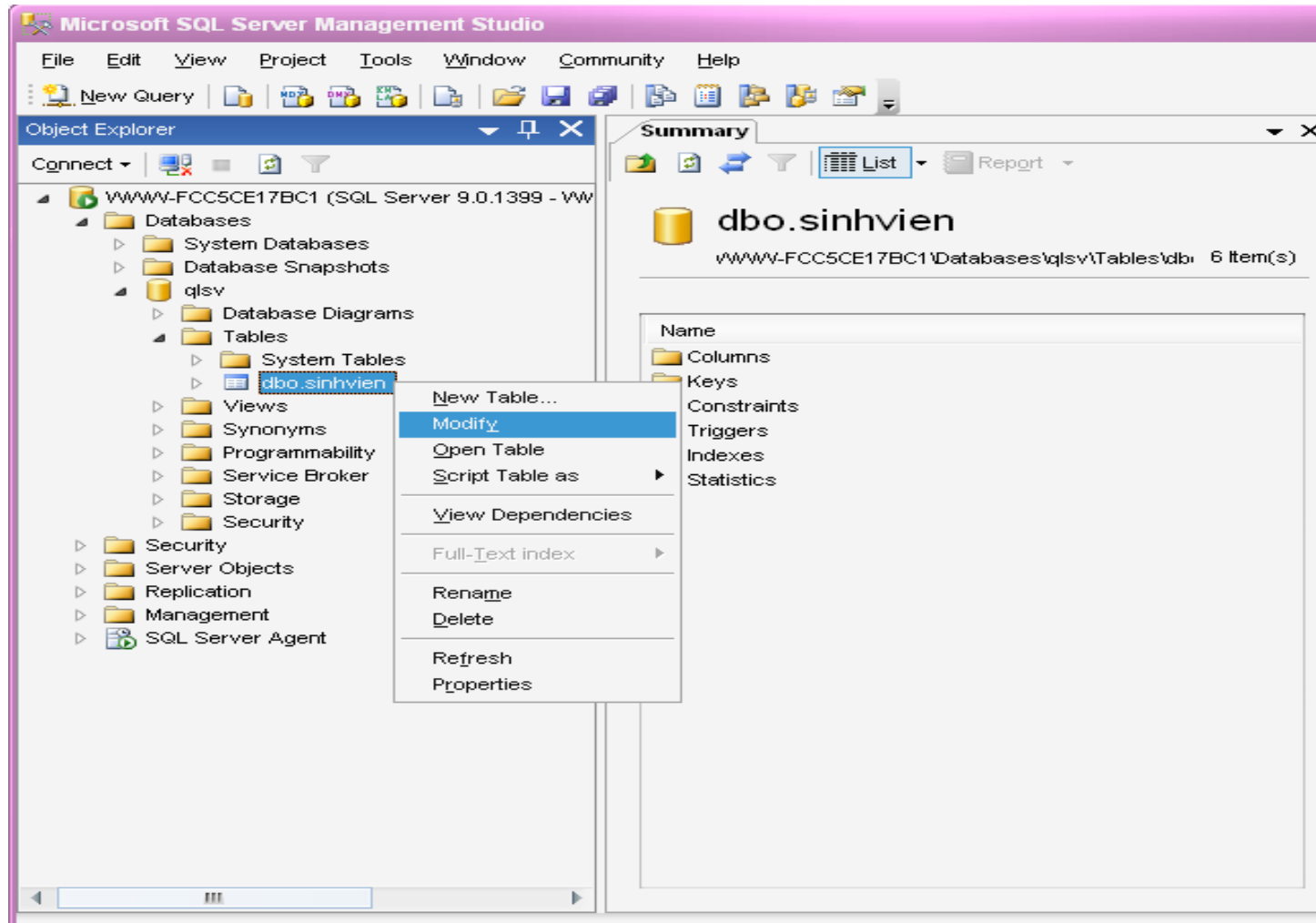
6. SỬA, XÓA CẤU TRÚC BẢNG

6.1. Sửa cấu trúc bảng

❖ Sử dụng công cụ.

- Chọn bảng cần sửa đổi của CSDL đang dùng.
- Nhấp chuột phải -> chọn Design.
- Thực hiện sửa cấu trúc bảng.

6. SỬA, XÓA CẤU TRÚC BẢNG



6. SỬA, XÓA CẤU TRÚC BẢNG

6.1. Sửa cấu trúc bảng

❖ *Sử dụng câu lệnh.*

- Thêm ràng buộc Check

[CONSTRAINT *tên_ràng_buộc*] CHECK (*điều_kiện*)

- Thêm ràng buộc Khóa

[CONSTRAINT *tên_ràng_buộc*] PRIMARY KEY [(*danh_sách_cột*)]

- Thêm ràng buộc Unique

[CONSTRAINT *tên_ràng_buộc*] UNIQUE [(*danh_sách_cột*)]

- Thêm ràng buộc FOREIGN KEY

[CONSTRAINT *tên_ràng_buộc*] FOREIGN KEY [(*danh_sách_cột*)]

REFERENCES *tên_bảng_tham_chiếu* (*danh_sách_cột_tham_chiếu*)

[ON DELETE CASCADE | NO ACTION | SET NULL | SET
DEFAULT]

[ON UPDATE CASCADE | NO ACTION | SET NULL | SET
DEFAULT]

6. SỬA, XÓA CẤU TRÚC BẢNG

6.1. Sửa cấu trúc bảng

❖ *Sử dụng câu lệnh*

- Nếu bảng Nhanvien chưa tạo trong CSDL thì:

CREATE TABLE nhanvien

(manv CHAR(10)

CONSTRAINT pk_manv PRIMARY KEY,

hoten NVARCHAR(50) NOT NULL,

Ngaysinh DATE

CONSTRAINT chk_ngaysinh

CHECK (year(getdate())-YEAR(NGAYSINH)>=22),

Dienthoai CHAR(11) NULL,

CONSTRAINT unique_nv UNIQUE(dienthoai),

Hsluong DECIMAL(3,2) DEFAULT(2.16))

6. SỬA, XÓA CẤU TRÚC BẢNG

6.1. Sửa cấu trúc bảng

❖ *Sử dụng câu lệnh*

- Nếu bảng Nhanvien đã tạo trong CSDL thì sử dụng cú pháp của câu lệnh ALTER TABLE như sau:

ALTER TABLE *tên_bảng*

ADD *định_nghĩa_cột* /

ALTER COLUMN *tên_cột* *kiểu_dữ_liệu* [NULL / NOT NULL] / DROP COLUMN *tên_cột* /

ADD CONSTRAINT *tên_ràng_buộc*

định_nghĩa_ràng_buộc / DROP CONSTRAINT
tên_ràng_buộc / NOCHECK CONSTRAINT *tên_rb*

6.1. SỬA CẤU TRÚC BẢNG

Giả sử ta có hai bảng DONVI và NHANVIEN với định nghĩa như sau:

- CREATE TABLE donvi
(Madv INT NOT NULL PRIMARY KEY,
Tendv NVARCHAR(30)NOT NULL)
- CREATE TABLE nhanvien
(Manv NVARCHAR(10) NOT NULL,
Hoten NVARCHAR(30) NOT NULL,
Ngaysinh DATETIME,
Diachi CHAR(30) NOT NULL)
- ALTER TABLE nhanvien
ADD dienthoai NVARCHAR(6)
CONSTRAINT chk_nhanvien_dienthoai
CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9][0-9]')
- ALTER TABLE nhanvien
ADD madv INT NULL

6.1. SỬA CẤU TRÚC BẢNG

- Định nghĩa lại kiểu dữ liệu của cột DIACHI trong bảng NHANVIEN và cho phép cột này chấp nhận giá trị NULL:

```
ALTER TABLE nhanvien
```

```
ALTER COLUMN diachi NVARCHAR(100) NULL
```

- Xóa cột ngày sinh khỏi bảng NHANVIEN:

```
ALTER TABLE nhanvien
```

```
DROP COLUMN ngaysinh
```

- Định nghĩa khoá chính (ràng buộc PRIMARY KEY) cho bảng NHANVIEN là cột MANV:

```
ALTER TABLE nhanvien
```

```
ADD CONSTRAINT pk_nhanvien PRIMARY KEY(manv)
```

6.1. SỬA CẤU TRÚC BẢNG

- Định nghĩa khoá ngoài cho bảng NHANVIEN trên cột MADV tham chiếu đến cột MADV của bảng DONVI:

```
ALTER TABLE nhanvien
```

```
ADD CONSTRAINT fk_nhanvien_madv
```

```
FOREIGN KEY(madv) REFERENCES donvi(madv)
```

```
ON DELETE CASCADE ON UPDATE CASCADE
```

- Xóa bỏ ràng buộc kiểm tra số điện thoại của nhân viên

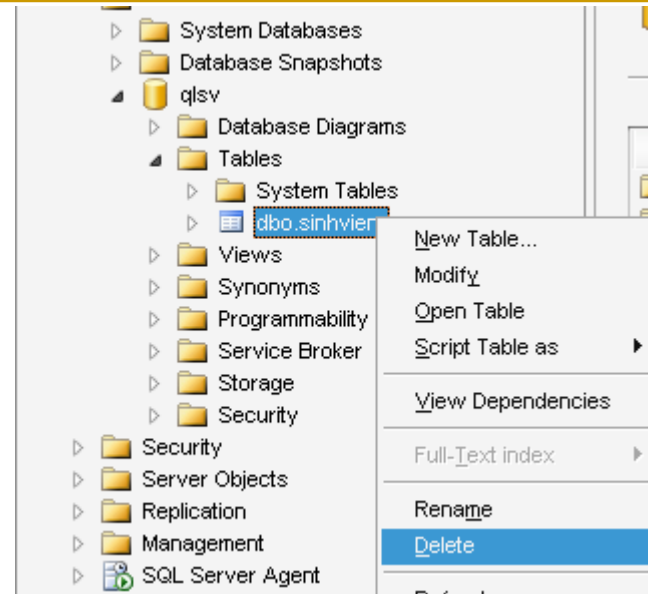
```
ALTER TABLE nhanvien
```

```
DROP CONSTRAINT CHK_NHANVIEN_DIENHONAI
```


6.2. Xóa bảng

■ Sử dụng công cụ.

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



✗ 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

➡ Phải xóa ràng buộc trước khi xóa bảng.

6.2. Xóa bảng

- *Sử dụng lệnh. DROP TABLE tên_bảng*
 - ❑ Xóa bỏ ràng buộc fk_nhanvien_madv khỏi bảng NHANVIEN:

ALTER TABLE nhanvien

DROP CONSTRAINT fk_nhanvien_madv

- ❑ Xóa bảng DONVI:

DROP TABLE *donvi*

7. Nhập dữ liệu vào bảng

■ *Sử dụng công cụ.*

- Chọn bảng dữ liệu
 - Nhấp chuột phải -> Edit top 200 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.

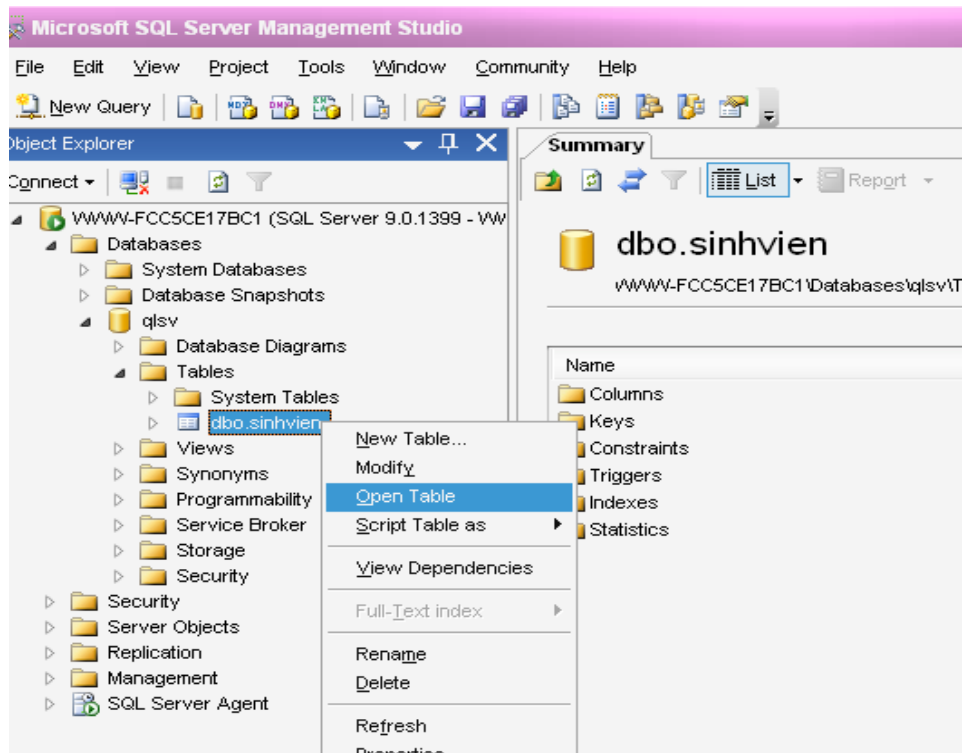


Table - dbo.sinhvien		Summary			
	MSV	Hoten	ngaysinh	que	diem
▶	a	h	12/2/2010 12:0...	NULL	NULL
	b	k	NULL	NULL	0
*	NULL	NULL	NULL	NULL	NULL

7. Nhập dữ liệu vào bảng

- *Sử dụng câu lệnh.*

- INSERT INTO *tên_bảng*[(*danh_sách_cột*)]

VALUES(*danh_sách_giá_trị*)

- Có thể không cần ghi *danh_sách_cột* sau *tên_bảng* nếu *danh_sách_giá_trị* phải đủ và theo thứ tự trong bảng.
 - Nếu cột dữ liệu hỗ trợ Unicode thì trước giá trị nhập vào bạn phải thêm kèm ký tự N

7. Nhập dữ liệu vào bảng

■ Sử dụng lệnh

- ❑ Insert into NXB(Ten, Dia_chi)
values(N'Kim Đồng', N'Hà Nội')

- ❑ INSERT INTO nhanvien

VALUES('NV01','Le Van A', '2/4/75', '896963', 2.14)

- ❑ INSERT INTO nhanvien(manv,hoten)

VALUES('NV02','Mai Thi B')

- ❑ INSERT INTO nhanvien(manv,hoten,dienthoai)

VALUES('NV03','Tran Thi C','849290')

7. Nhập dữ liệu vào bảng

■ *INSERT vào bảng lấy giá trị từ bảng khác*

- Các giá trị dữ liệu được bổ sung vào bảng không được chỉ định tường minh mà thay vào đó là một câu lệnh SELECT truy vấn dữ liệu từ bảng khác.

- Cú pháp

INSERT INTO tên_bảng[(danh_sách_cột)]
câu_lệnh_SELECT

- Ví dụ: INSERT INTO luusinhvien
 SELECT hodem,ten,ngaysinh
 FROM sinhvien
 WHERE noisinh like '%Huế%'

8. Cập nhật dữ liệu bằng câu lệnh UPDATE

UPDATE tên_bảng -- bảng được cập nhật
SET tên_cột = biểu_thức [..., tên_cột = biểu_thức]
[FROM danh_sách_bảng] -- các bảng khác có liên quan
[WHERE điều_kiện] --đk cập nhật, đk liên kết
Ví dụ: cập nhật lại số TC của các môn học có số TC
lớn hơn 4 -> 4

```
UPDATE monhoc  
SET sotc = 4  
WHERE sotc >4
```


8. Cập nhật dữ liệu bằng câu lệnh UPDATE

Ví dụ: Cập nhật tiền phòng, nếu phòng loại A thì 100\$, nếu phòng loại B thì 70\$, các loại phòng còn lại là 50\$

SOPHONG	LOAIPHONG	SONGAY	TIENPHONG
101	A	5	
202	B	5	
101	A	2	
102	C	3	

SOPHONG	LOAIPHONG	SONGAY	TIENPHONG
101	A	5	500
202	B	5	350
101	A	2	200
102	C	3	150

UPDATE nhatkypdong

SET tienphong=songay*CASE

WHEN loaiphong='A' THEN 100

WHEN loaiphong='B' THEN 70

ELSE 50

END

8. Cập nhật dữ liệu bằng câu lệnh UPDATE

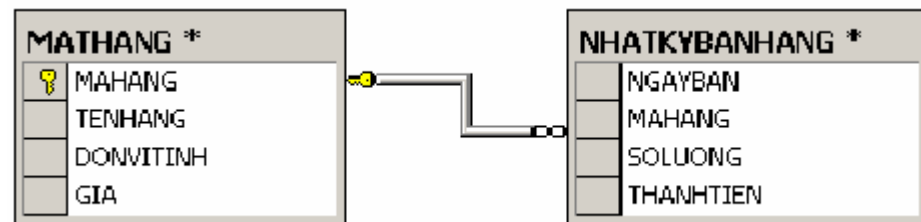
Ví dụ: Cập nhật cột thành tiền = giá* số lượng

UPDATE nhattybanhang

SET thanh tien = soluong*gia

FROM mathang

WHERE nhattybanhang.mahang =
mathang.mahang



8. Cập nhật dữ liệu bằng câu lệnh UPDATE

*Câu lệnh UPDATE với truy vấn con

Tương tự như trong câu lệnh SELECT, truy vấn con có thể được sử dụng trong mệnh đề WHERE của câu lệnh UPDATE nhằm chỉ định điều kiện đối với các dòng dữ liệu cần cập nhật dữ liệu.

Ví dụ: Câu lệnh trên có thể được viết như sau:

UPDATE nhattybanhang

*SET thanh tien = soluong*gia*

FROM mathang

WHERE mathang.mahang =(SELECT mathang.mahang

FROM mathang

WHERE mathang.mahang=nhattybanhang.mahang)

9. Xóa dữ liệu bằng câu lệnh DELETE

DELETE FROM tên_bảng -- bảng bị xóa dữ liệu
[FROM danh_sách_bảng] -- các bảng có liên quan khác
[WHERE điều_kiện] -- đk xóa, đk kết nối bảng

Ví dụ: Câu lệnh dưới đây xóa khỏi bảng SINHVIEN những sinh viên sinh tại Huế

```
DELETE FROM sinhvien  
WHERE noisinh LIKE N'%Huế%'
```

Ví dụ: Xóa dữ liệu khi điều kiện liên quan đến nhiều bảng, ví dụ xóa ra khỏi bảng SINHVIEN những sinh viên Ngành Hóa dầu

```
DELETE FROM sinhvien  
FROM dsnganh  
WHERE dsnganh.manganh=sinhvien.manganh  
AND tennganh=N'Hóa dầu'
```

9. Sử dụng truy vấn con trong câu lệnh DELETE

ví dụ: xóa khỏi bảng LOP những lớp không có sinh viên nào học

```
DELETE FROM lop
```

```
WHERE malop NOT IN (SELECT DISTINCT  
malop FROM sinhvien)
```

Ví dụ: Xóa toàn bộ dữ liệu trong bảng

```
DELETE FROM diemthi
```

có tác dụng tương tự với câu lệnh

```
TRUNCATE TABLE diemthi
```

10. Các hàm thông dụng trong SQL

- Hàm thống kê: Avg, min, max, count, sum
- Hàm xử lý chuỗi: ltrim, rtrim, left, right,...
- Hàm thời gian: getdate, day, month, year,...
- Hàm toán học: square, sqrt, round, rand, ceiling...
 - ❑ `select ceiling(100*rand())`—sinh ngẫu nhiên các số 1..100
 - ❑ `select round(rand()*100,0)`
 - ❑ `select ceiling(6*rand()+4)`

Bài 3: Truy vấn dữ liệu

1. Câu lệnh truy vấn tổng quát
 2. Các mệnh đề trong câu lệnh truy vấn
 3. Kết nối các bảng
 4. Các loại kết nối
 5. Truy vấn con
-

1. Truy vấn dữ liệu

Câu lệnh SELECT tổng quát:

```
SELECT [DISTINCT][TOP n] <danh_sách_chọn_các_cột>  
[INTO <tên_bảng_mới>]  
FROM <danh_sách_bảng/khung_nhìn>  
[WHERE <các_điều_kiện_ràng_buộc>]  
[GROUP BY <danh_sách_cột>]  
[HAVING <điều_kiện_bắt_buộc_dựa_trên_GROUP_BY>]  
[ORDER BY <cột_sắp_xếp>]  
[COMPUTE danh_sách_hàm_gộp [BY danh_sách_cột]]
```


1. Truy vấn dữ liệu trên 1 bảng

a. Câu lệnh *SELECT* với mệnh đề *FROM*

Mệnh đề *FROM* chỉ tên một bảng hay những bảng có quan hệ cần truy vấn thông tin.

SELECT tentruong --chọn các trường

FROM tenbang --từ các bảng

Dấu * cho phép lọc mẫu tin với tất cả các trường trong bảng.

1. Truy vấn dữ liệu trên 1 bảng

b. Câu lệnh *SELECT* với mệnh đề *WHERE*

Dùng *WHERE* để tạo điều kiện cần lọc mẫu tin theo tiêu chuẩn nào đó.

SELECT *

FROM tenbang

WHERE biểu thức điều kiện

Các phép toán trong biểu thức điều kiện: >, <, >=, <=, =, !=, <>, !>, !<, or, and, not, between...and..., like, in

Ký tự đại diện	ý nghĩa
%	Chuỗi ký tự bất kỳ gồm không hoặc nhiều ký tự
_	Ký tự đơn bất kỳ
[]	Ký tự đơn bất kỳ trong giới hạn được chỉ định (ví dụ [a-f]) hay một tập (ví dụ [abcdef])
[^]	Ký tự đơn bất kỳ không nằm trong giới hạn được chỉ định (ví dụ [^a-f] hay một tập (ví dụ [^abcdef])).

1. Truy vấn dữ liệu trên 1 bảng

b. Câu lệnh SELECT với mệnh đề WHERE

*Ví dụ: SELECT * FROM monhoc
 WHERE sotc>2

*Ví dụ: SELECT hodem,ten FROM sinhvien
WHERE hodem LIKE 'Lê%' AND ten LIKE '[AB]%'

*Trong mệnh đề WHERE, để kiểm tra giá trị của một cột có giá trị NULL hay không, ta sử dụng cách viết:

WHERE tên_cột IS NULL

WHERE tên_cột IS NOT NULL

1. Truy vấn dữ liệu trên 1 bảng

c. Câu lệnh SELECT với mệnh đề ORDER BY

- Thông thường khi truy vấn, kết quả hiển thị sắp xếp theo thứ tự Alphabet, nhưng bạn cũng có thể sắp xếp theo tiêu chuẩn bất kỳ nhờ ORDER BY.

ORDER BY columnname DESC –sx giảm

ORDER BY columnname ASC – sx tăng

- Số cột sắp xếp tối đa là 16, thứ tự sắp xếp từ trái qua phải
- Có thể chỉ định số thứ tự của cột sau ORDER BY

1. Truy vấn dữ liệu trên 1 bảng

*c. Câu lệnh **SELECT** với mệnh đề **ORDER BY***

- Ví dụ:

```
SELECT hodem,ten,gioitinh,  
YEAR(GETDATE())-YEAR(ngaysinh) AS tuoi  
FROM sinhvien  
WHERE ten='Bình'  
ORDER BY gioitinh,tuoi  
có kết quả là:
```

HODEM	TEN	GIOITINH	TUOI
Nguyễn Thị	Bình	0	23
Hoàng Văn	Bình	1	21
Châu Văn Quốc	Bình	1	21
Nguyễn Thanh	Bình	1	22
Nguyễn Đình	Bình	1	22
Nguyễn Công	Bình	1	25

1. Truy vấn dữ liệu trên 1 bảng

*d. Câu lệnh **SELECT** với mệnh đề **GROUP BY***

Khi truy vấn mẫu tin của một hay nhiều bảng, thông thường có những nghiệp vụ thuộc trường nào đó có cùng giá trị ta dùng **GROUP BY** để nhóm các giá trị đó lại

- Ví dụ:

Select mahang, sum(soluong)

From banhang

Group by mahang

1. Truy vấn dữ liệu trên 1 bảng

e. Câu lệnh SELECT với mệnh đề AS

Khi thay đổi tên trường trong truy vấn ta dùng AS

Ví dụ:

```
SELECT 'Mã lớp' = malop, tenlop 'Tên  
lớp', khoa AS 'Khoá'  
FROM lop
```

1. Truy vấn dữ liệu trên 1 bảng

f. Câu lệnh SELECT với mệnh đề INTO

Câu lệnh SELECT ... INTO có tác dụng tạo một bảng mới có cấu trúc và dữ liệu được xác định từ kết quả của truy vấn. Bảng mới được tạo ra sẽ có số cột bằng số cột được chỉ định trong danh sách chọn và số dòng sẽ là số dòng kết quả của truy vấn.

1. Truy vấn dữ liệu trên 1 bảng

*f. Câu lệnh **SELECT** với mệnh đề **INTO***

Ví dụ: Câu lệnh dưới đây truy vấn dữ liệu từ bảng SINHVIEN và tạo một bảng TUOISV bao gồm các trường HODEM, TEN và TUOI

```
SELECT hodem, ten, YEAR (GETDATE ( ) ) -  
    YEAR (ngaysinh) AS tuoi  
INTO tuoi  
FROM sinhvien
```

1. Truy vấn dữ liệu trên 1 bảng

g. Câu lệnh SELECT với mệnh đề Top N

Cho phép lấy ra chỉ một số mẫu tin theo một tiêu chuẩn hay sắp xếp nào đó.

Ví dụ: hiển thị họ tên và ngày sinh của 5 sinh viên đầu tiên trong danh sách

```
SELECT TOP 5 hodem, ten, ngaysinh  
FROM sinhvien
```

Ví dụ: hiển thị họ tên và ngày sinh của 10% số lượng sinh viên hiện có trong bảng SINHVIEN

```
SELECT TOP 10 PERCENT hodem, ten, ngaysinh  
FROM sinhvien
```

1. Truy vấn dữ liệu trên 1 bảng

h. Câu lệnh SELECT với DISTINCT

Khi ta có một hay nhiều bảng kết nối với nhau, sẽ xảy ra trùng lặp nhiều mẫu tin. Nếu chỉ cần lấy ra một mẫu tin trong những mẫu tin đó bạn sử dụng SELECT với DISTINCT

Ví dụ: SELECT khoa FROM lop

KHOA
24
24
24
24
25
25
25
25
26
26

KHOA
24
25
26

SELECT DISTINCT khoa FROM lop

1. Truy vấn dữ liệu trên 1 bảng

i. Sử dụng cấu trúc CASE trong danh sách chọn

Cấu trúc CASE được sử dụng trong danh sách chọn nhằm thay đổi kết quả của truy vấn tùy thuộc vào các trường hợp khác nhau. Cấu trúc này có cú pháp như sau:

CASE biểu_thức

WHEN biểu_thức_kiểm_tra THEN kết_quả

[...]

ELSE kết_quả_của_else]

END

1. Truy vấn dữ liệu trên 1 bảng

i. Sử dụng cấu trúc CASE trong danh sách chọn

Ví dụ: Để hiển thị giới tính là Nam hay Nữ

```
SELECT masv, hodem, ten,  
       CASE  
         WHEN gioitinh=1 THEN 'Nam'  
         ELSE 'Nữ'  
       END AS gioitinh  
FROM sinhvien
```

1. Truy vấn dữ liệu trên 1 bảng

Câu lệnh select với mệnh đề Having

■ Điều kiện hiển thị dựa trên Group by

SELECT mahang, sum(soluong)

FROM chitietdathang

group BY mahang

having count(mahang)>2

1. Truy vấn dữ liệu trên 1 bảng

j. Câu lệnh select với mệnh đề Compute dùng để thống kê

SELECT mahang, soluong

FROM chitietdathang

ORDER BY mahang

--COMPUTE SUM(soluong) BY mahang

COMPUTE count(mahang), SUM(soluong)

☞ Chú ý Compute... by
dùng để thống kê theo từng nhóm

Results			Messages		
	mahang	soluong			
1	DC01	1000			
2	DC02	1000			
3	DC03	1000			
4	DT01	2			
5	DT04	2			
6	DT05	20			
7	MM01	80			
8	MM01	30			
9	MM01	30			
10	MM02	20			
11	MM02	30			
12	TP01	5			
13	TP02	5			
14	TP03	5			
15	TP03	1000			
16	TP03	200			
17	TP06	50			
18	TP07	100			
	cnt	sum			
1	18	4579			

2. Truy vấn dựa trên nhiều bảng

MÀKHÓA	TENKHOA	DIENTHOAI
DHT01	Khoa Toán cơ - Tin học	054822407
DHT02	Khoa Công nghệ thông tin	054826767
DHT03	Khoa Vật lý	054823462
DHT04	Khoa Hoá học	054823951
DHT05	Khoa Sinh học	054822934
DHT06	Khoa Địa lý - Địa chất	054823837

MALOP	TENLOP	KHOA	HEDAOTAO	NAMNHAPHOC	MÀKHÓA
C24101	Toán K24	24	Chính quy	2000	DHT01
C24102	Tin K24	24	Chính quy	2000	DHT02
C24103	Lý K24	24	Chính quy	2000	DHT03
C24301	Sinh K24	24	Chính quy	2000	DHT05
C25101	Toán K25	25	Chính quy	2001	DHT01
C25102	Tin K25	25	Chính quy	2001	DHT02
C25103	Lý K25	25	Chính quy	2001	DHT03
C25301	Sinh K25	25	Chính quy	2001	DHT05
C26101	Toán K26	26	Chính quy	2002	DHT01
C26102	Tin K26	26	Chính quy	2002	DHT02

Cần biết mã lớp, tên lớp của các lớp thuộc khoa công nghệ thông tin, ta làm như sau:

2. Truy vấn dựa trên nhiều bảng

- Chọn ra dòng trong bảng KHOA có tên khoa là *Khoa Công nghệ thông tin* từ đó xác định được mã khoa (MAKHOA) là *DHT02*
- Tìm kiếm trong bảng LOP những dòng có giá trị trường MAKHOA là *DHT02*

Câu lệnh như sau:

```
SELECT malop,tenlop
```

```
FROM khoa,lop
```

```
WHERE khoa.makhoa = lop.makhoa
```

AND

```
Tenkhoa = 'Khoa Công nghệ Thông tin'
```

MAKHOA	TENKHOA	DIENTHOAI
DHT01	Khoa Toán cơ - Tin học	054822407
DHT02	Khoa Công nghệ thông tin	054826767
DHT03	Khoa Vật lý	054823462
DHT04	Khoa Hóa học	054823951
DHT05	Khoa Sinh học	054822934
DHT06	Khoa Địa lý - Địa chất	054823637

MALOP	TENLOP	KHOA	HEDAOTAO	NAMNHAPHOC	MAKHOA
C24101	Toán K24	24	Chính quy	2000	DHT01
C24102	Tin K24	24	Chính quy	2000	DHT02
C24103	Lý K24	24	Chính quy	2000	DHT03
C24301	Sinh K24	24	Chính quy	2000	DHT05
C25101	Toán K25	25	Chính quy	2001	DHT01
C25102	Tin K25	25	Chính quy	2001	DHT02
C25103	Lý K25	25	Chính quy	2001	DHT03
C25301	Sinh K25	25	Chính quy	2001	DHT05
C26101	Toán K26	26	Chính quy	2002	DHT01
C26102	Tin K26	26	Chính quy	2002	DHT02

MALOP	TENLOP
C24102	Tin K24
C25102	Tin K25
C26102	Tin K26

3. Kết nối các bảng

- Một câu lệnh nối thực hiện lấy các dòng dữ liệu trong các bảng tham gia truy vấn, so sánh giá trị của các dòng này trên một hoặc nhiều cột được chỉ định trong điều kiện nối và kết hợp các dòng thỏa mãn điều kiện thành những dòng kết quả truy vấn.
- Cần phải xác định các yếu tố sau:
 - Những cột nào cần hiển thị trong kết quả truy vấn
 - Những bảng nào có tham gia vào truy vấn
 - Điều kiện để thực hiện phép nối giữa các bảng dữ liệu là gì

3. Kết nối các bảng

■ Danh sách chọn trong kết nối

Việc sử dụng tên các cột trong danh sách chọn có thể là:

- Tên của một số cột nào đó trong các bảng có tham gia vào truy vấn. Nếu tên cột trong các bảng trùng tên nhau thì tên cột phải được viết dưới dạng

tên_bảng.tên_cột

- Dấu sao (*) được sử dụng trong danh sách chọn khi cần hiển thị tất cả các cột của các bảng tham gia truy vấn.

tên_bảng.*

3. Kết nối các bảng

■ Mệnh đề FROM trong kết nối

Sau mệnh đề FROM của câu lệnh nối là danh sách tên các bảng (hay khung nhìn) tham gia vào truy vấn. Nếu ta sử dụng dấu * trong danh sách chọn thì thứ tự của các bảng liệt kê sau FROM sẽ ảnh hưởng đến thứ tự các cột được hiển thị trong kết quả truy vấn.

3. Kết nối các bảng

■ Mệnh đề WHERE trong kết nối

Khi hai hay nhiều bảng được nối với nhau, ta phải chỉ định điều kiện để thực hiện nối ngay sau mệnh đề WHERE.

Điều kiện nối được biểu diễn dưới dạng biểu thức logic so sánh giá trị dữ liệu giữa các cột của các bảng tham gia truy vấn.

Ví dụ:

```
SELECT masv, hodem, ten, sinhvien.malop, tenlop,  
       tenkhoa
```

```
FROM sinhvien, lop, khoa
```

```
WHERE sinhvien.malop = lop.malop AND  
      lop.makhoa=khoa.makhoa
```

4. Các loại kết nối

a. Kết nối bằng (kết nối tự nhiên)

Kết nối bằng (equi-join) là một phép nối trong đó giá trị của các cột được sử dụng để nối được so sánh với nhau dựa trên tiêu chuẩn bằng và tất cả các cột trong các bảng tham gia nối đều được đưa ra trong kết quả.

Ví dụ: Câu lệnh dưới đây thực hiện kết nối bằng giữa hai bảng LOP và KHOA

```
SELECT *  
FROM lop, khoa  
WHERE lop.makhoa=khoa.makhoa
```

4. Các loại kết nối

a. Kết nối bằng và kết nối tự nhiên

Trong kết nối tự nhiên, điều kiện nối giữa hai bảng chính là điều kiện bằng giữa khoá ngoài và khoá chính của hai bảng; Và trong danh sách chọn của câu lệnh chỉ giữ lại một cột trong hai cột tham gia vào điều kiện của phép nối.

Ví dụ: Để thực hiện phép nối tự nhiên, câu lệnh trong ví dụ trên được viết lại như sau

```
SELECT lop.*,tenkhoa,dienthoai  
FROM lop,khoa  
WHERE lop.makhoa=khoa.makhoa
```

4. Các loại kết nối

b. Phép nối với các điều kiện bổ sung

Trong các câu lệnh nối, ngoài điều kiện của phép nối được chỉ định trong mệnh đề WHERE còn có thể chỉ định các điều kiện tìm kiếm dữ liệu khác (điều kiện chọn). Thông thường, các điều kiện này được kết hợp với điều kiện nối thông qua toán tử AND.

Ví dụ: Câu lệnh dưới đây hiển thị họ tên và ngày sinh của các sinh viên *Khoa Công nghệ Thông tin*

```
SELECT hodem,ten,ngaysinh  
FROM sinhvien,lop,khoa  
WHERE tenkhoa='Khoa Công nghệ Thông tin'  
AND sinhvien.malop = lop.malop AND  
lop.makhoa = khoa.makhoa
```


4. Các loại kết nối

c. Phép tự nối qua bí danh

Phép tự nối là phép nối mà trong đó điều kiện nối được chỉ định liên quan đến các cột của cùng một bảng. Trong trường hợp này, sẽ có sự xuất hiện tên của cùng một bảng nhiều lần trong mệnh đề FROM và do đó các bảng cần phải được đặt bí danh.

Ví dụ: *SELECT a.ho,a.ten,b.ho,b.ten,b.ngaysinh
FROM nhanvien a INNER JOIN nhanvien b
ON day(a.ngaysinh)=day(b.ngaysinh) AND
a.manhanvien<>b.manhanvien*

Chú ý: Trong trường hợp chưa thiết lập quan hệ cho các bảng ta sử dụng INNER JOIN

4. Các loại kết nối

d. Phép nối ngoài (outer-join)

Trong các phép nối đã đề cập ở trên, chỉ những dòng có giá trị trong các cột được chỉ định thoả mãn điều kiện kết nối mới được hiển thị trong kết quả truy vấn, và được gọi là *phép nối trong* (inner join).

Tuy nhiên, đôi khi ta cũng cần giữ lại những thông tin này bằng cách cho phép những dòng không thoả mãn điều kiện nối cũng có mặt trong kết quả của truy vấn. Để làm điều này, ta có thể sử dụng *phép nối ngoài* (outer-join).

4. Các loại kết nối

d. Phép nối ngoài (outer-join) gồm

- **Phép nối ngoài trái** (ký hiệu: $\ast=$, LEFT JOIN):
Phép nối này hiển thị trong kết quả truy vấn tất cả các mẫu tin bảng bên trái tồn tại ứng với những mẫu tin bảng bên phải không tồn tại
- **Phép nối ngoài phải** (ký hiệu: $=\ast$, RIGHT JOIN):
Phép nối này hiển thị trong kết quả truy vấn tất cả các dòng dữ liệu của bảng nằm bên phải cho dù dữ liệu của bảng bên trái không tồn tại

4. Các loại kết nối

Ví dụ:

Bảng DONVI

MADV	TENDV
1	Doi ngoai
2	Hanh chinh
3	Ke toan
4	Kinh doanh

Bảng NHANVIEN

HOTEN	MADV
Thanh	1
Hoa	2
Nam	2
Vinh	1
Hung	5
Phuong	NULL

SELECT *

FROM nhanvien,donvi

WHERE nhanvien.madv=donvi.madv

Hoặc:SELECT *

FROM nhanvien

INNER JOIN donvi

ON nhanvien.madv=donvi.madv

HOTEN	MADV	MADV	TENDV
Thanh	1	1	Doi ngoai
Hoa	2	2	Hanh chinh
Nam	2	2	Hanh chinh
Vinh	1	1	Doi ngoai

4. Các loại kết nối

Ví dụ: Nếu thực hiện phép nối ngoài trái giữa bảng NHANVIEN và bảng DONVI

SELECT *

FROM nhanvien,donvi

WHERE nhanvien.madv*=donvi.madv

HOTEN	MADV	MADV	TENDV
Thanh	1	1	Doi ngoai
Hoa	2	2	Hanh chinh
Nam	2	2	Hanh chinh
Vinh	1	1	Doi ngoai
Hung	5	NULL	NULL
Phuong	NULL	NULL	NULL

Viết lại câu lệnh trên bằng cách khác???

4. Các loại kết nối

Ví dụ: Câu lệnh thực hiện phép nối ngoài phải giữa hai bảng NHANVIEN và DONVI

SELECT *

FROM nhanvien RIGHT JOIN donvi

ON nhanvien.madv=donvi.madv

HOTEN	MADV	MADV	TENDV
Thanh	1	1	Doi ngoai
Vinh	1	1	Doi ngoai
Hoa	2	2	Hanh chinh
Nam	2	2	Hanh chinh
NULL	NULL	3	Ke toan
NULL	NULL	4	Kinh doanh

So sánh kết quả???

4. Các loại kết nối

e. FULL JOIN được sử dụng khi không cần quan tâm đến dữ liệu hiện ra cho dù dữ liệu bảng bên trái có tồn tại tương ứng dữ liệu bảng bên phải hay không (hoặc ngược lại)

Bài 4: Câu lệnh hợp (UNION)

1. Định nghĩa: Câu lệnh hợp được sử dụng trong trường hợp ta cần gộp kết quả của hai hay nhiều truy vấn thành một tập kết quả duy nhất.

2. Cú pháp như sau:

Câu_lệnh_select_1

UNION [ALL] *Câu_lệnh_select_2*

[UNION [ALL] *Câu_lệnh__select_3*]

...

[UNION [ALL] *Câu_lệnh_select_n*]

[ORDER BY *cột_sắp_xếp*]

[COMPUTE *danh_sách_hàm_gộp* [BY *danh_sách_cột*]]

Bài 4: Câu lệnh hợp (UNION)

3. Ví dụ: có 2 bảng

SELECT A,B FROM Table1

UNION

SELECT D,E FROM table2

A	B	C
a	1	10
b	2	20
c	3	30
d	4	40
a	5	50
b	6	60

D	E
a	1
b	2
d	3
e	4

A	B
a	1
a	5
b	2
b	6
c	3
d	3
d	4
e	4

→
Kết quả

Mặc định kết quả sẽ bỏ đi những dòng dữ liệu giống nhau, nếu muốn giữ lại những dòng dữ liệu giống nhau ta dùng UNION ALL.

Bài 4: Câu lệnh hợp (UNION)

4. Các nguyên tắc khi sử dụng UNION:

- Danh sách cột trong các truy vấn thành phần phải có cùng số lượng.
- Các cột tương ứng trong tất cả các bảng, hoặc tập con bất kỳ các cột được sử dụng trong bản thân mỗi truy vấn thành phần phải cùng kiểu dữ liệu.
- Các cột tương ứng trong bản thân từng truy vấn thành phần của một câu lệnh UNION phải xuất hiện theo thứ tự như nhau. Nguyên nhân là do phép hợp so sánh các cột từng cột một theo thứ tự được cho trong mỗi truy vấn.
- Khi các kiểu dữ liệu khác nhau được kết hợp với nhau trong câu lệnh UNION, chúng sẽ được chuyển sang kiểu dữ liệu cao hơn (nếu có thể được).
- Tiêu đề cột trong kết quả của phép hợp sẽ là tiêu đề cột được chỉ định trong truy vấn đầu tiên.

Bài 4: Câu lệnh hợp (UNION)

4. Các nguyên tắc khi sử dụng UNION:

- Truy vấn thành phần đầu tiên có thể có INTO để tạo mới một bảng từ kết quả của chính phép hợp.
- Mệnh đề ORDER BY và COMPUTE dùng để sắp xếp kết quả truy vấn hoặc tính toán các giá trị thống kê chỉ được sử dụng ở cuối câu lệnh UNION. Chúng không được sử dụng trong bất kỳ truy vấn thành phần nào.
- Mệnh đề GROUP BY và HAVING chỉ có thể được sử dụng trong bản thân từng truy vấn thành phần. Chúng không được phép sử dụng để tác động lên kết quả chung của phép hợp.
- Phép toán UNION có thể được sử dụng bên trong câu lệnh INSERT.
- Phép toán UNION không được sử dụng trong câu lệnh CREATE VIEW.

Bài 5: KHUNG NHÌN – VIEW

1. Khái niệm
 2. Tạo View
 3. Một số quy định
 4. Sửa đổi View
 5. Xóa khung nhìn
-

Bài 5: KHUNG NHÌN – VIEW

1. Khái niệm

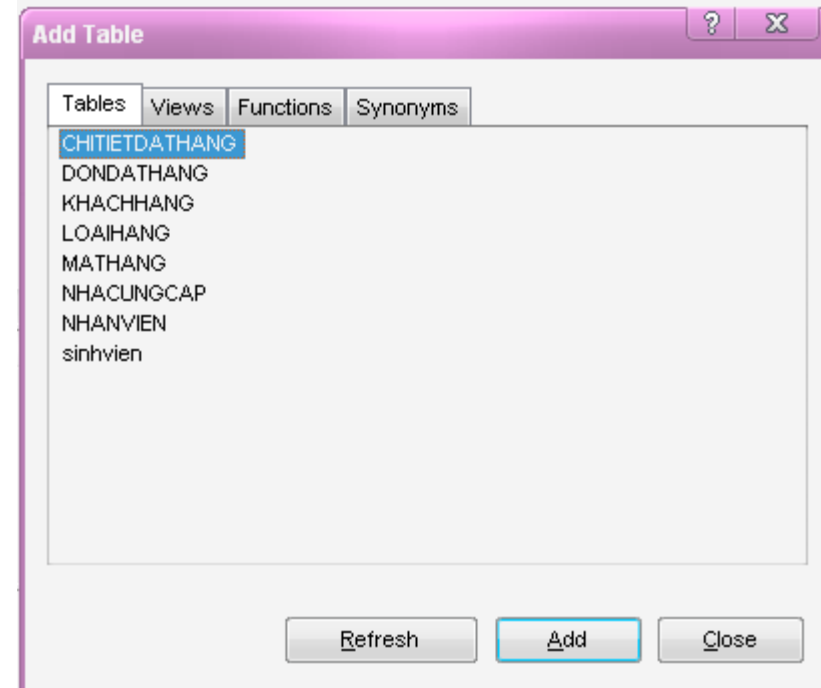
- * Khung nhìn (View) là một bảng tạm thời, có cấu trúc như một bảng, khung nhìn là đối tượng thuộc CSDL.
- * Khung nhìn được tạo ra từ câu lệnh truy vấn dữ liệu (lệnh Select), truy vấn từ một hoặc nhiều bảng dữ liệu.
- * Khung nhìn được sử dụng khai thác dữ liệu như một bảng dữ liệu, chia sẻ nhiều người dùng, an toàn trong khai thác, không ảnh hưởng dữ liệu gốc.

Bài 5: KHUNG NHÌN – VIEW

2. Tạo view

a. Bằng công cụ

- Chuột phải tại View->new view
- Add bảng vào
- Thao tác trên cửa sổ theo kịch bản
- Thực thi (!)
- Lưu view



Bài 5: Khung nhìn

WWW-FCC5CE17BC1...- SQLQuery1.sql* View - dbo.View_1' Summary

NHANVIEN

- ☐ * (All Columns)
- ☒ MaNhanVien
- ☐ Ho
- ☐ Ten
- ☐ NgaySinh

DONDATHANG

- ☐ * (All Columns)
- ☐ SoHoaDon
- ☐ MaKhachHang
- ☐ MaNhanVien
- ☒ NgayDatHang

Column	Alias	Table	Output	Sort Type	Sort Order	Filter	Or...
MaNhanVien		NHANVIEN	<input checked="" type="checkbox"/>			LIKE 'H%'	
NgayDatHang		DONDATHANG...	<input checked="" type="checkbox"/>				

```
SELECT dbo.NHANVIEN.MaNhanVien, dbo.DONDATHANG.NgayDatHang
FROM   dbo.NHANVIEN INNER JOIN
       dbo.DONDATHANG ON dbo.NHANVIEN.MaNhanVien = dbo.DONDATHANG.MaNhanVien
WHERE  (dbo.NHANVIEN.MaNhanVien LIKE 'H%')
```

MaNhanVien	NgayDatHang
H001	9/20/2007 12:0...
H002	9/20/2007 12:0...
H003	9/20/2007 12:0...

Bài 5: Khung nhìn

■ Tạo khung nhìn bằng câu lệnh

CREATE VIEW tên_khung_nhìn[(danh_sách_tên_cột)] AS
câu_lệnh_SELECT

Ví dụ:

```
create view xemnhanviencomaH as  
SELECT NHANVIEN.MaNhanVien, NgayDatHang  
FROM NHANVIEN INNER JOIN  
DONDATHANG ON NHANVIEN.MaNhanVien =  
DONDATHANG.MaNhanVien  
WHERE (NHANVIEN.MaNhanVien LIKE 'H%')
```

Xem view: `select * from xemnhanviencomaH`

Bài 5: Khung nhìn

3. Một số quy định

- Tên khung nhìn và tên cột trong khung nhìn, cũng giống như bảng, phải tuân theo qui tắc định danh.
- Không thể qui định ràng buộc và tạo chỉ mục cho khung nhìn.
- Câu lệnh SELECT với mệnh đề COMPUTE ... BY không được sử dụng để định nghĩa khung nhìn.
- Phải đặt tên cho các cột của khung nhìn trong các trường hợp sau đây:
 - Trong kết quả của câu lệnh SELECT có ít nhất một cột được sinh ra bởi một biểu thức (tức là không phải là một tên cột trong bảng cơ sở) và cột đó không được đặt tiêu đề.
 - Tồn tại hai cột trong kết quả của câu lệnh SELECT có cùng tiêu đề cột.

Bài 5: Khung nhìn

4. Sửa đổi khung nhìn

- Câu lệnh ALTER VIEW được sử dụng để định nghĩa lại khung nhìn hiện có nhưng không làm thay đổi các quyền đã được cấp phát cho người sử dụng trước đó. Câu lệnh này sử dụng tương tự như câu lệnh CREATE VIEW và có cú pháp như sau:

*ALTER VIEW tên_khung_nhìn
[(danh_sách_tên_cột)] AS
Câu_lệnh_SELECT*

Bài 5: Khung nhìn

5. Xoá khung nhìn

- Khi một khung nhìn không còn sử dụng, ta có thể xoá nó ra khỏi cơ sở dữ liệu thông qua câu lệnh:

DROP VIEW *tên_khung_nhìn*

- Nếu một khung nhìn bị xoá, toàn bộ những quyền đã cấp phát cho người sử dụng trên khung nhìn cũng đồng thời bị xoá. Do đó, nếu ta tạo lại khung nhìn thì phải tiến hành cấp phát lại quyền cho người sử dụng.

Ví dụ : **DROP VIEW xemnhanviencomaH**

Bài 6: Thủ tục lưu trữ (Stored Procedure)

1. Khái niệm
 2. Phân loại thủ tục
 3. Tạo thủ tục
 4. Các câu lệnh điều khiển
 5. Lời gọi thủ tục
 6. Giá trị trả về trong thủ tục
 7. Tham số với giá trị mặc định
 8. Sửa, xóa thủ tục
-

Bài 6: Thủ tục thường trú (Stored Procedure)

1. Khái niệm:

- Thủ tục là một chương trình con cho các ứng dụng truy cập vào một hệ thống quản lý cơ sở dữ liệu quan hệ (Wikipedia)
- Thủ tục là một đối tượng của CSDL, nó được tạo ra từ một kịch bản câu lệnh SQL
- Thủ tục có nhiều ưu điểm so với thực hiện câu lệnh SQL từ các máy khách:
 - + Lập trình theo module
 - + Thực hiện nhanh hơn
 - + Làm giảm lưu lượng trên mạng
 - + An ninh bảo mật hơn

Bài 6: Thủ tục thường trú (Stored Procedure)

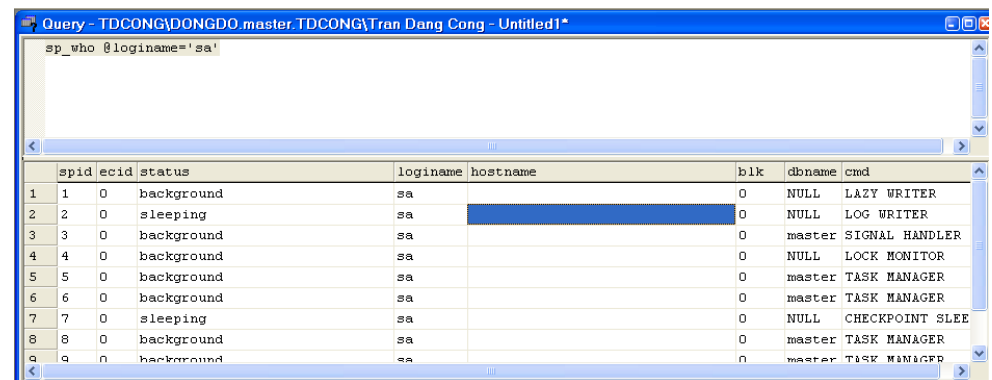
2. Phân loại thủ tục: có 5 loại

❑ System Stored Procedure.

Là thủ tục được lưu trữ trong CSDL Master, được bắt đầu bằng chữ **sp_** 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'`



	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 SLEEP
8	8	0	background	sa		0	master	TASK MANAGER
9	9	0	background	sa		0	master	TASK MANAGER

Bài 6: Thủ tục thường trú (Stored Procedure)

2. Phân loại thủ tục: có 5 loại

❑ **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 Procedure 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.

Bài 6: Thủ tục thường trú (Stored Procedure)

2. Phân loại thủ tục: có 5 loại

❑ **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 để gửi mail, thủ tục xp_cmdshell dùng để thực hiện lệnh của DOS (xp_cmdshell 'dir c:\').

❑ **Remote Stored Procedure:**

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

Bài 6: Thủ tục thường trú (Stored Procedure)

3. Tạo Thủ tục

***CREATE PROCEDURE/PROC tên_thủ_tục [(danh_sách_tham_số)]
[WITH RECOMPILE/ENCRYPTION/RECOMPILE/ENCRYPTION]
AS***

Các_câu_lệnh_của_thủ_tục

Trong đó:

- tên_thủ_tục: phải theo qui tắc định danh, không được vượt quá 128 ký tự.
 - danh_sách_tham_số: nếu thủ tục có nhiều tham số thì các khai báo phân cách nhau bởi dấu phẩy. Một tham số tối thiểu phải bao gồm hai phần:
 - ☐ tên tham số được bắt đầu bởi dấu @.
 - ☐ kiểu dữ liệu của tham số
- Ví dụ:** @mahang nvarchar(10)

Bài 6: Thủ tục thường trú (Stored Procedure)

***CREATE PROCEDURE/PROC tên_thủ_tục
[(danh_sách_tham_số)]***

***[WITH RECOMPILE / WITH ENCRYPTION / RECOMPILE /
ENCRYPTION]***

AS

Các_câu_lệnh_của_thủ_tục

Trong đó:

- WITH RECOMPILE: yêu cầu SQL Server biên dịch lại thủ tục mỗi khi được gọi.
- WITH ENCRYPTION: yêu cầu SQL Server mã hóa thủ tục.
- các_câu_lệnh_của_thủ_tục: Tập hợp các câu lệnh sử dụng trong nội dung thủ tục. Các câu lệnh này có thể đặt trong cặp từ khoá BEGIN...END hoặc có thể không.

Bài 6: Thủ tục thường trú (Stored Procedure)

4. Các câu điều khiển

- Lệnh gán: SET, SELECT,
- Lệnh in: SELECT, PRINT
- Lệnh ghép: BEGIN...END
- IF ... ELSE

IF <BIỂU THỨC LOGIC>

CÂU LỆNH SQL/BEGIN CÂU LỆNH END

ELSE

CÂU LỆNH SQL/BEGIN CÂU LỆNH END

- CASE: cho phép nhận một giá trị từ nhiều lựa chọn

CASE <Biểu thức logic>

WHEN <Biểu thức > THEN <Câu lệnh>

...

[ELSE <Câu lệnh>]

END

Bài 6: Thủ tục thường trú (Stored Procedure)

4. Các câu lệnh điều khiển

- *WHILE*

WHILE <Biểu thức logic>

Câu lệnh SQL

Hoặc BEGIN

Câu lệnh SQL

[BREAK]

[CONTINUE]

END

Trong cấu trúc *WHILE* có thể dùng *WAITFOR DELAY/TIME* <TIME> để tạm dừng một thời gian *TIME* trước khi xử lý tiếp các phát biểu sau đó

Bài 6: Thủ tục thường trú (Stored Procedure)

Ví dụ

```
CREATE PROC sp_LenDanhSachDiem(  
    @mamonhoc          NVARCHAR(10),  
    @tenmonhoc          NVARCHAR(50),  
    @sodvht             SMALLINT,  
    @malop              NVARCHAR(10))  
AS  
BEGIN  
    INSERT INTO monhoc  
    VALUES( @mamonhoc, @tenmonhoc, @sodvht)  
    INSERT INTO diemthi(mamonhoc,masv)  
    SELECT @mamonhoc,masv  
        FROM sinhvien  
        WHERE malop=@malop  
END
```

Bài 6: Thủ tục thường trú (Stored Procedure)

5. Lời gọi thủ tục

tên_thủ_tục **[danh_sách_các_đối_số]**

Exec tên_thủ_tục **[danh_sách_các_đối_số]**

Ví dụ: Lời gọi thủ tục ở ví dụ trên có thể viết như sau:

sp_LenDanhSachDiem 'Tl-005', 'Cơ sở dữ liệu', 5,
 'C24102' hoặc

sp_LenDanhSachDiem @malop='C24102',
 @tenmonhoc='Cơ sở dữ liệu',
 @mamonhoc='TI-005',
 @sodvht=5

Bài 6: Thủ tục thường trú (Stored Procedure)

6. Giá trị trả về trong thủ tục

- Tham số chứa giá trị trả về có thêm chữ OUT.
- Lệnh gọi có thêm từ khóa EXECUTE
- Khai báo biến phụ bằng từ khóa DECLARE
- Ví dụ:

```
CREATE PROC t_cong( @a int, @b int, @c int out)  
AS Select @c=@a+@b
```

- Lệnh gọi

```
declare @tong int  
select @tong=0  
execute t_cong 12,7, @tong out  
select @tong
```

Bài 6: Thủ tục thường trú (Stored Procedure)

7. Tham số với giá trị mặc định

```
CREATE PROC t_macdinh( @tenlop NVARCHAR(30)=NULL,  
@noisinh NVARCHAR(100)='Huế') AS  
BEGIN
```

```
    IF @tenlop IS NULL
```

```
        SELECT hodem,ten FROM sinhvien INNER JOIN lop ON  
        sinhvien.malop=lop.malop
```

```
            WHERE noisinh=@noisinh
```

```
    ELSE
```

```
        SELECT hodem,ten FROM sinhvien INNER JOIN lop ON  
        sinhvien.malop=lop.malop
```

```
            WHERE noisinh=@noisinh AND tenlop=@tenlop
```

```
END
```

t_macdinh

tìm những sv sinh ở Huế

~~t_macdinh @tenlop='Tin K29'~~ ~~tìm những sv Tink29 sinh ở Huế~~

Bài 6: Thủ tục thường trú (Stored Procedure)

8. 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: Alter Procedure

Xóa: Drop Procedure

Bài 7: HÀM DO NGƯỜI DÙNG ĐỊNH NGHĨA

- Hàm do người dùng định nghĩa về mặt nào đó thì nó giống như thủ tục ở bài trước ta đã tìm hiểu (gồm các câu lệnh SQL kết hợp lại). Nhưng hàm trả lại kết quả thông qua tên hàm còn thủ tục thì không chính vì vậy mà trong thủ tục có tham số OUTPUT để lấy kết quả trả về.

■ CREATE FUNCTION [tên chủ sở hữu.]tên_hàm
([danh_sách_tham_số (gán giá trị default nếu cần)])

RETURNS (kiểu_trả_về_của_hàm)

AS

BEGIN

các_câu_lệnh_của_hàm

END

Bài 7: HÀM DO NGƯỜI DÙNG ĐỊNH

NGHĨA

■ Hàm trả về bảng dữ liệu

Nhìn chung là giống View table nhưng được truyền tham số theo ý muốn

```
CREATE FUNCTION tên_hàm([danh_sách_tham_số])
```

```
RETURNS TABLE
```

```
AS
```

1 câu RETURN (1 câu lệnh_select)

- Chú ý: Trong phần thân của hàm chỉ có duy nhất một câu lệnh RETURN xác định giá trị trả về của hàm thông qua duy nhất một câu lệnh SELECT. Ngoài ra, không sử dụng bất kỳ câu lệnh nào khác trong phần thân của hàm.

Bài 7: HÀM DO NGƯỜI DÙNG ĐỊNH NGHĨA

*Tạo view

```
CREATE VIEW    sinhvien_k25  
AS  
SELECT masv,hodem,ten,ngaysinh  
FROM sinhvien INNER JOIN lop  
ON sinhvien.malop=lop.malop  
WHERE khoa=25
```

*Thực thi view SELECT * FROM sinhvien_K25

Bài 7: HÀM DO NGƯỜI DÙNG ĐỊNH NGHĨA

*Tạo hàm

```
CREATE FUNCTION funcXemSV(@khoa SMALLINT)
RETURNS TABLE
AS
RETURN(SELECT masv,hodem,ten,ngaysinh
FROM sinhvien INNER JOIN lop
ON sinhvien.malop=lop.malop
WHERE khoa=@khoa)
```

*Thực thi hàm `SELECT * FROM funcXemSV(25)`

Bài 7: HÀM DO NGƯỜI DÙNG ĐỊNH NGHĨA

■ Trong trường hợp cần phải sử dụng đến nhiều câu lệnh trong phần thân của hàm, ta sử dụng cú pháp như sau để định nghĩa hàm:

```
■ CREATE FUNCTION tên_hàm([danh_sách_tham_số])  
RETURNS @biến_bảng TABLE định_nghĩa_bảng  
AS  
BEGIN  
    các_câu_lệnh_trong_thân_hàm  
RETURN  
END
```

Bài 7: HÀM DO NGƯỜI DÙNG ĐỊNH

NGHĨA

- Khi định nghĩa hàm dạng này cần lưu ý một số điểm sau:
 - Cấu trúc của bảng trả về bởi hàm được xác định dựa vào định nghĩa của bảng trong mệnh đề RETURNS. Biến *@biến_bảng* trong mệnh đề RETURNS có phạm vi sử dụng trong hàm và được sử dụng như là một tên bảng.
 - Câu lệnh RETURN trong thân hàm không chỉ định giá trị trả về. Giá trị trả về của hàm chính là các dòng dữ liệu trong bảng có tên là *@biếnbảng* được định nghĩa trong mệnh đề RETURNS

Bài 7: HÀM DO NGƯỜI DÙNG ĐỊNH NGHĨA

```
create function xem_b(@mlh char(4))
returns @bang table (th nvarchar(30),sl int, gh int)
as
begin
    if @mlh=' '
        insert into @bang
        select tenhang,soluong,giahang from mathang
    else
        insert into @bang
        select tenhang,soluong,giahang
        from mathang
        where maloaihang=@mlh
return
end
```

Bài 8: **TRIGGER**

1. Khái niệm
 2. Những trường hợp sử dụng Trigger
 3. Đặc điểm của Trigger
 4. Tạo Trigger
 5. Sử dụng mệnh đề IF UPDATE trong trigger
-

Bài 8: TRIGGER

1. 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 thường 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.

Bài 8: TRIGGER

2. NHỮNG TRƯỜNG HỢP SỬ DỤNG TRIGGER.

- ❑ Khi các biện pháp toàn vẹn dữ liệu như Constraint, rule,... không bảo đảm. Khác với các công cụ bảo 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.
- ❑ Khi thay đổi dây 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).

Bài 8: TRIGGER

3. 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 template 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 Table, 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).

Bài 8: TRIGGER

4. Tạo Trigger

- Trigger tạo ra sẽ được áp dụng đối với bảng nào?
- Trigger được kích hoạt khi câu lệnh nào được thực thi trên bảng: INSERT, UPDATE, DELETE?
- Trigger sẽ làm gì khi được kích hoạt?
- Cú pháp như sau:

```
CREATE TRIGGER tên_trigger  
ON tên_bảng  
FOR {[INSERT][,][UPDATE][,][DELETE]} AS  
    [IF UPDATE(tên_cột)  
    [AND UPDATE(tên_cột)|OR UPDATE(tên_cột)]  
    các_câu_lệnh_của_trigger
```

Bài 8: TRIGGER

■ Ví dụ:

```
CREATE TRIGGER trg_nhatkybanhang_insert
ON nhatkybanhang
FOR INSERT AS
    UPDATE mathang
    SET mathang.soluong=mathang.soluong-
        inserted.soluong
    FROM mathang INNER JOIN inserted
        ON mathang.mahang=inserted.mahang
```

Bài 8: TRIGGER

Chuẩn SQL định nghĩa hai bảng logic INSERTED và DELETED để sử dụng trong các trigger. Cấu trúc của hai bảng này tương tự như cấu trúc của bảng mà trigger tác động. Dữ liệu trong hai bảng này tùy thuộc vào câu lệnh tác động lên bảng làm kích hoạt trigger; cụ thể trong các trường hợp sau:

- * Khi câu lệnh DELETE được thực thi trên bảng, các dòng dữ liệu bị xóa sẽ được sao chép vào trong bảng DELETED. Bảng INSERTED trong trường hợp này không có dữ liệu.
- * Dữ liệu trong bảng INSERTED sẽ là dòng dữ liệu được bổ sung vào bảng gây nên sự kích hoạt đối với trigger bằng câu lệnh INSERT. Bảng DELETED trong trường hợp này không có dữ liệu.
- * Khi câu lệnh UPDATE được thực thi trên bảng, các dòng dữ liệu cũ chịu sự tác động của câu lệnh sẽ được sao chép vào bảng DELETED, còn trong bảng INSERTED sẽ là các dòng sau khi đã được cập nhật.

Bài 8: TRIGGER

5. Sử dụng mệnh đề IF UPDATE trong trigger

- Thay vì chỉ định một trigger được kích hoạt trên một bảng, ta có thể chỉ định trigger được kích hoạt và thực hiện những thao tác cụ thể khi việc thay đổi dữ liệu chỉ liên quan đến một số cột nhất định nào đó của cột. Trong trường hợp này, ta sử dụng mệnh đề IF UPDATE trong trigger. IF UPDATE không sử dụng được đối với câu lệnh DELETE.

Bài 8: TRIGGER

Ví dụ: Xét lại ví dụ với hai bảng MATHANG và NHATKYBANHANG, trigger dưới đây được kích hoạt khi ta tiến hành cập nhật cột SOLUONG cho một bản ghi của bảng NHATKYBANHANG (lưu ý là chỉ cập nhật đúng một bản ghi)

```
CREATE TRIGGER trg_nhatkybanhang_update_soluong
ON nhatkybanhang
FOR UPDATE AS
    IF UPDATE(soluong)
        UPDATE mathang
        SET  mathang.soluong = mathang.soluong -
(inserted.soluong-deleted.soluong)
        FROM (deleted INNER JOIN inserted ON
        deleted.stt = inserted.stt) INNER JOIN mathang
        ON mathang.mahang = deleted.mahang
```

Bài 8: TRIGGER

Mệnh đề IF UPDATE có thể xuất hiện nhiều lần trong phần thân của trigger. Khi đó, mệnh đề IF UPDATE nào đúng thì phần câu lệnh của mệnh đề đó sẽ được thực thi khi trigger được kích hoạt.

```
CREATE TABLE R(  
  A INT, B INT, C INT)  
và trigger trgRtest cho bảng R:  
CREATE TRIGGER trgRtest  
  ON R  
  FOR UPDATE AS  
    IF UPDATE(A)  
      Print 'A updated'  
    IF UPDATE(C)  
      Print 'C'
```

Giao tác

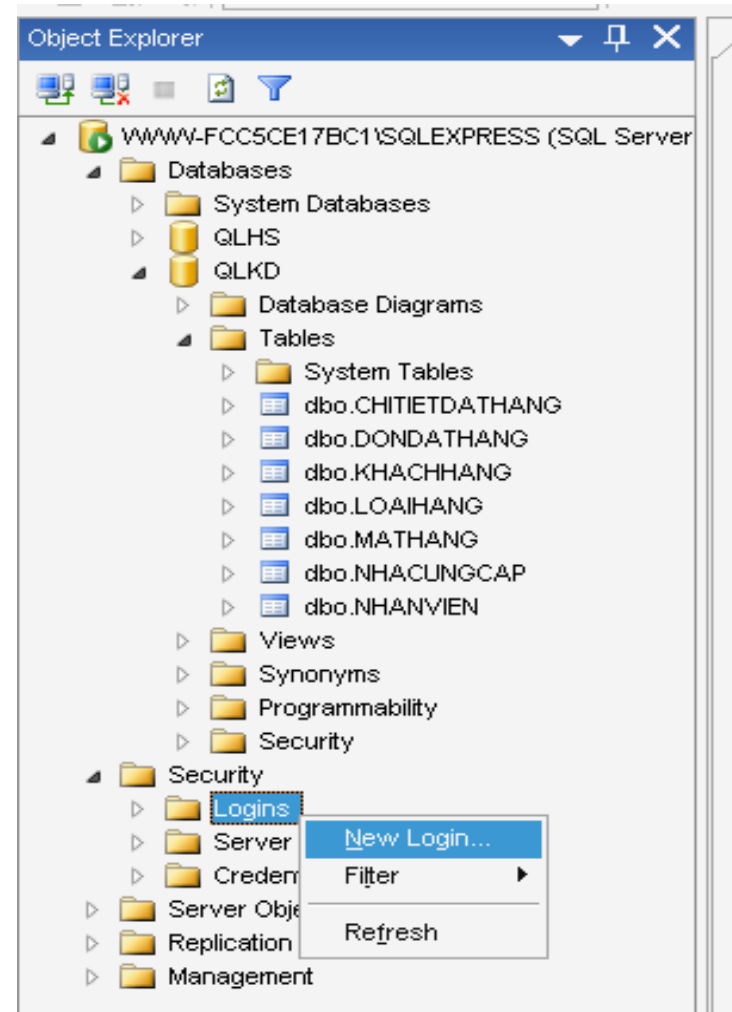
- Giả sử có một tài khoản trong ngân hàng, tên tài khoản là Khang, cần chuyển một số tiền trị giá 1000000đ từ TK ATM sang tiền tiết kiệm. Ngân hàng phải thực hiện 2 nhiệm vụ
 - Giảm 1000000đ trong TK ATM
 - Tăng 1000000đ trong bảng tiền tiết kiệm
- Trong khoảng thời gian xảy ra 2 nghiệp vụ trên, bất kỳ một ảnh hưởng phần cứng, sự cố mạng, hay lỗi do ràng buộc dữ liệu đều có thể dẫn đến sai sót dữ liệu. Nếu sai sót xảy ra, có thể phát biểu thứ nhất không thực hiện nhưng phát biểu thứ 2 vẫn diễn ra, cũng có thể phát biểu thứ 2 gặp sự cố trong khi phát biểu thứ nhất thực hiện tốt. Điều gì sẽ xảy ra với TK của Khang.
- Xuất phát từ 2 phát biểu rời rạc trên, chúng ta cần có một thể thống nhất cho 2 nghiệp vụ này: một là cả 2 cùng xảy ra thành công, hoặc cả 2 không thực hiện gì cả. Đây chính là khái niệm giao tác.
- Giao tác được định nghĩa bắt đầu bằng **BEGIN TRAN** và kết thúc bởi hành động **COMMIT TRAN** hoặc **ROLLBACK TRAN**

Bài 9: Bảo mật và phân quyền người dùng

1. Cách thực hiện

Bước 1: Tạo Login name

Mục Security – chuột phải tại Logins – Chọn New Login



Bài 9: Bảo mật và phân quyền người dùng

1. Cách thực hiện:

Bước 1: Tạo Login name

Login - New

Select a page

- General
- Server Roles
- User Mapping
- Securables
- Status

Script Help

Login name: TuyetQN Search...

☐ Windows authentication

☒ SQL Server authentication

Password:

Confirm password:

☐ Enforce password policy

☐ Enforce password expiration

☐ User must change password at next login

☐ Mapped to certificate

Certificate name:

☐ Mapped to asymmetric key

Key name:

Default database: QLKD

Default language: <default>

Connection

Server: WWW-FCC5CE17BC1\SQLEXP

Connection: WWW-FCC5CE17BC1\TuyetQN

[View connection properties](#)

Progress

Ready

OK Cancel

Gõ
tên

Gõ mật
khẩu

Chọn
Data

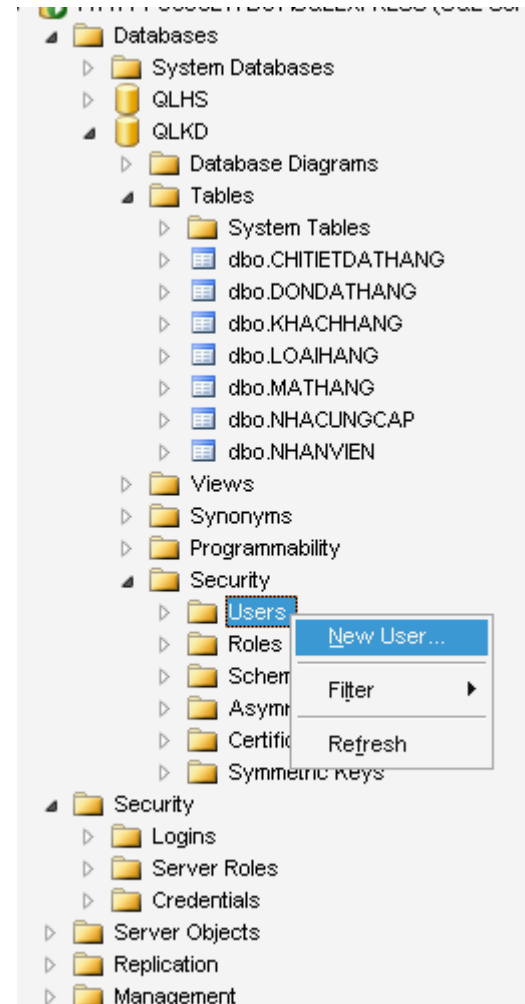
Nhấn OK

Bài 9: Bảo mật và phân quyền người dùng

1. Cách thực hiện:

Bước 2: Tạo Users name trùng tên
với Login name

Mục Security – nhấn chuột phải –
chọn Users – Chọn New User

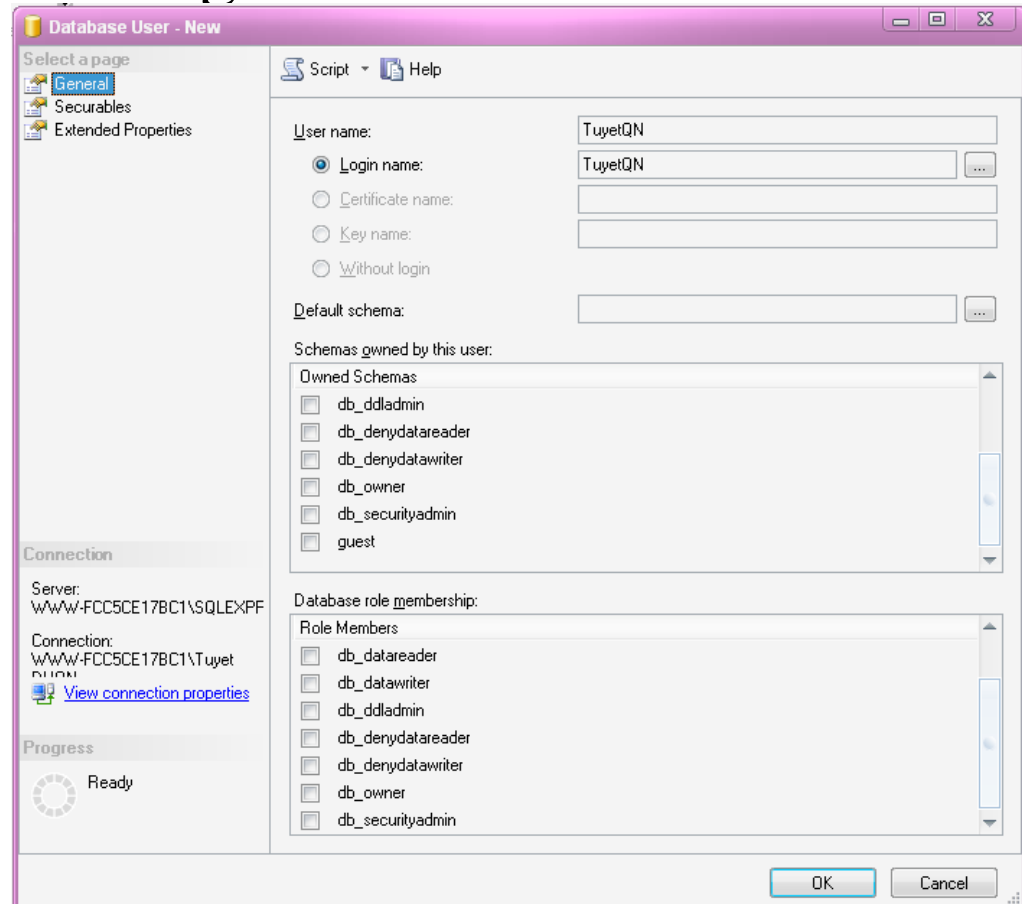


Bài 9: Bảo mật và phân quyền người dùng

1. Cách thực hiện:

Bước 2: Tạo Users name trùng tên

với Login name



Bài 9: Bảo mật và phân quyền người dùng

2. Các mức bảo mật (Levels of Security) Có 4 mức độ sau:

- **Mức hệ điều hành**

Để kết nối với hệ phục vụ, một người dùng cần xem xét tỉ mỉ vì kiểu thủ tục đăng nhập hệ điều hành nhằm truy cập hệ thống hợp lệ.

- **SQL Server**

Để kết nối với SQL Server, người dùng cần phải có đăng nhập người dùng SQL hợp lệ.

- **Database**

Để truy xuất CSDL trong SQL Server, người dùng phải được cấp quyền trên CSDL đó.

- **Đối tượng (bảng, view, store procedure)**

Muốn truy xuất một đối tượng trong một CSDL, người dùng phải được cấp quyền trên đối tượng đó.

Bài 9: Bảo mật và phân quyền người dùng

3. Các kiểu bảo mật

- a) *Standard Security* (chế độ bảo mật ngầm định) người dùng muốn khởi nhập vào SQL Server phải cung cấp tên người dùng và mật khẩu để SQL Server đối chiếu với bảng hệ thống.
- b) *Integrated Security* Việc quản trị người dùng SQL Server tích hợp trực tiếp với hệ điều hành Windows NT. Người sử dụng dùng tài khoản Windows NT hợp lệ có thể khởi nhập vào SQL Server mà không cần cung cấp tên người dùng và mật khẩu một lần nữa.
- c) *Mixed Security*
Mixed Security phối hợp cả Integrated và Standard Security.

Bài 9: Bảo mật và phân quyền người dùng

4. Phân cấp tính bảo mật (Security Hierachy)

a. Quản trị viên hệ thống (System Administrator - SA)

SA là người có toàn quyền truy cập SQL Server. Mọi câu lệnh SQL đều có thể thực thi bởi SA. SA cũng có thể cấp quyền cho các user khác.

b. Sở hữu chủ CSDL (Database Owner - DBO)

DBO là người dùng đã tạo CSDL hoặc được gán quyền sở hữu. DBO có toàn quyền truy cập đến mọi đối tượng trong CSDL của DBO và được phép gán quyền đối tượng (object permission) cho những người dùng khác.

Bài 9: Bảo mật và phân quyền người dùng

4. Phân cấp tính bảo mật (Security Hierachy)

c. *Sở hữu chủ đối tượng CSDL (Database Object Owner - DBOO)* là người tạo đối tượng CSDL. SQL Server thừa nhận rằng nếu bạn có quyền cần thiết để tạo đối tượng thì đương nhiên bạn có mọi quyền với đối tượng đó (select, update, insert, delete, reference và execute).

d. *Người dùng khác* cần phải được cấp các quyền đối tượng (select, update, insert, delete, reference và execute) để hoạt động trong CSDL. SA cũng có thể cấp quyền cho những người dùng khác nên họ có thể tạo và xóa đối tượng trong CSDL.

Bài 9: Bảo mật và phân quyền người dùng

5. Các quyền trên CSDL bao gồm:

- Public: quyền chung cho tất cả các người dùng CSDL, đây là quyền tối thiểu để truy cập CSDL.
 - Db_owner: quyền sở hữu CSDL
 - Db_accessadmin: quyền quản trị truy cập CSDL
 - Db_securityadmin: quyền bảo mật CSDL
 - Db_backupoperator: quyền sao lưu dữ liệu
 - Db_datareader: quyền đọc dữ liệu trong CSDL
 - Db_datawriter: quyền ghi dữ liệu lên CSDL
 - Db_denydatareader: không được quyền đọc dữ liệu trong CSDL
 - Db_denydatawriter: không được quyền ghi dữ liệu lên CSDL
-

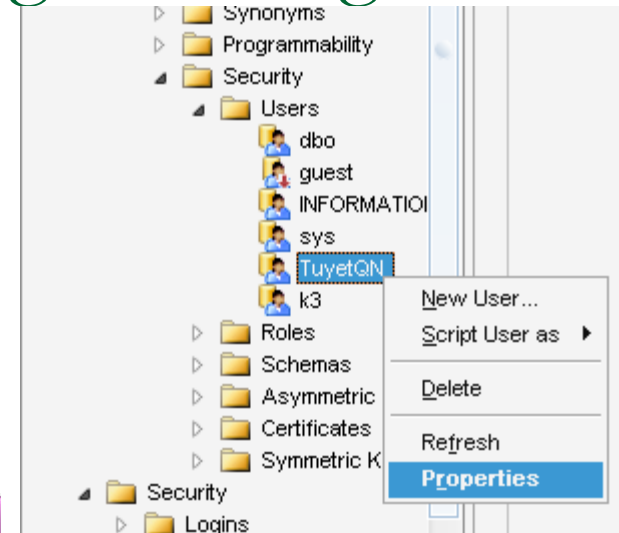
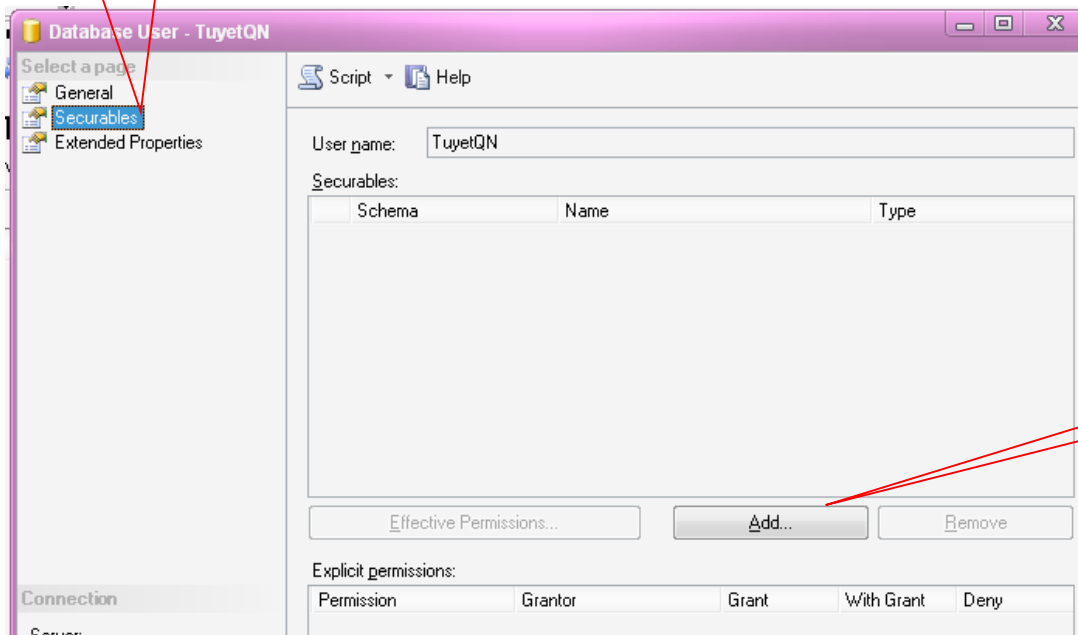
Bài 9: Bảo mật và phân quyền người dùng

Bước 3. Phân quyền người dùng

B1: Nhấn chuột phải tại tên User

- Chọn Properties
- Chọn Securables - Add

Chọn

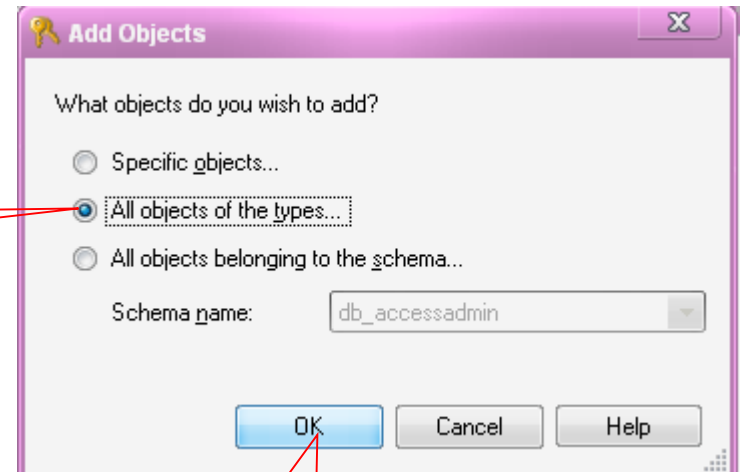


Chọn add

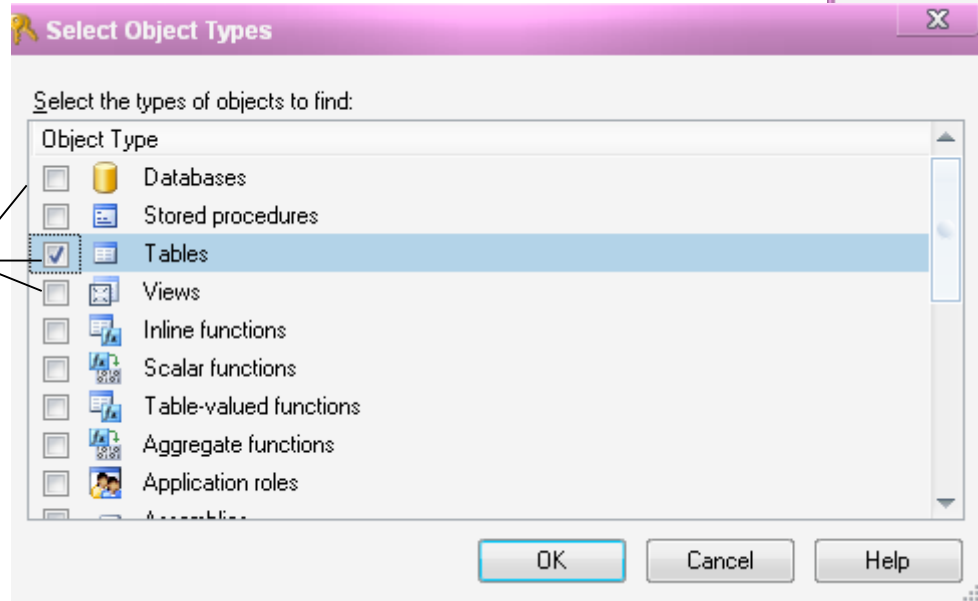
Bài 9: Bảo mật và phân quyền người dùng

6. Phân quyền người dùng B2:

Chọn



Chọn



Chọn

Bài 9: Bảo mật và phân quyền người dùng

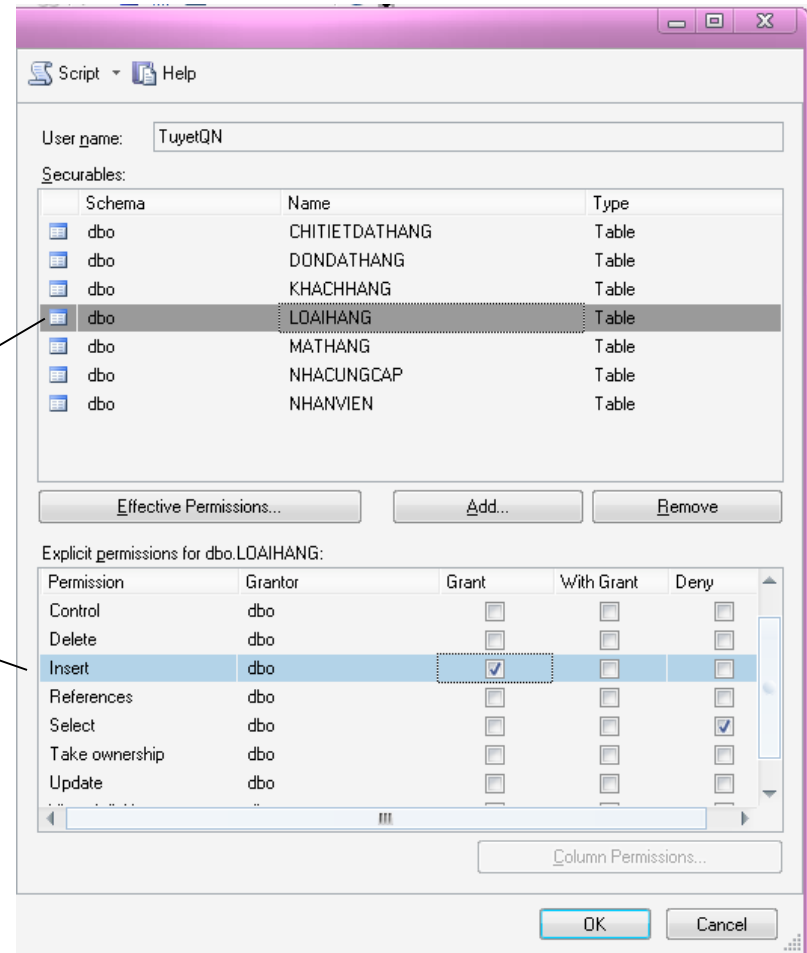
6. Phân quyền người dùng

B2:

Grant: cho quyền

Deny: từ chối quyền

Chọn

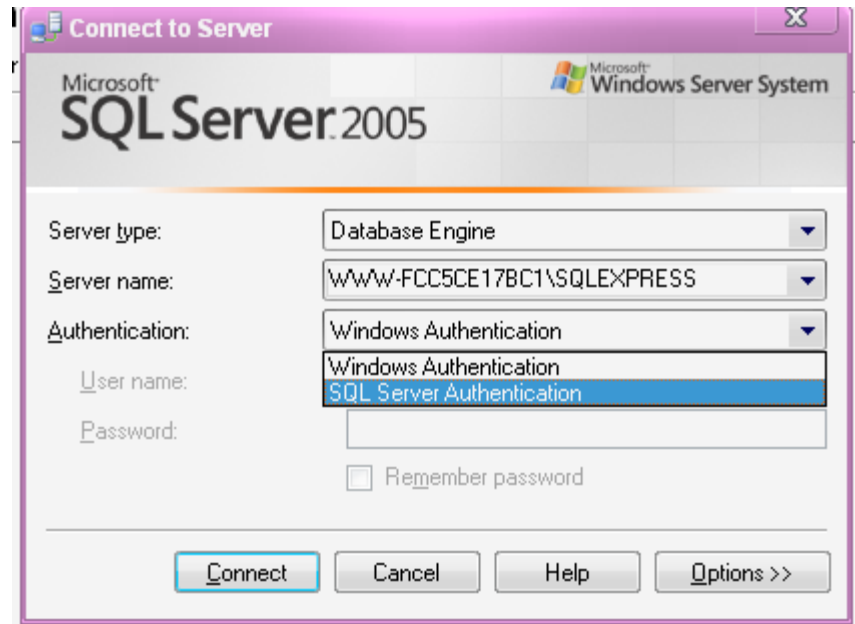


Bài 9: Bảo mật và phân quyền người dùng

6. Phân quyền người dùng

B3: Kết nối user

- File – Disconnect object..
- File – Connect object...
- Mục Authentication chọn SQL Server Authentication
- Gõ user name:....
- Gõ Password:....
- Chọn Connect



Bài 9: Bảo mật và phân quyền người dùng

- USE Northwind

GRANT SELECT

ON Customers

TO PUBLIC

- USE Northwind

DENY SELECT

ON Customers

TO PUBLIC

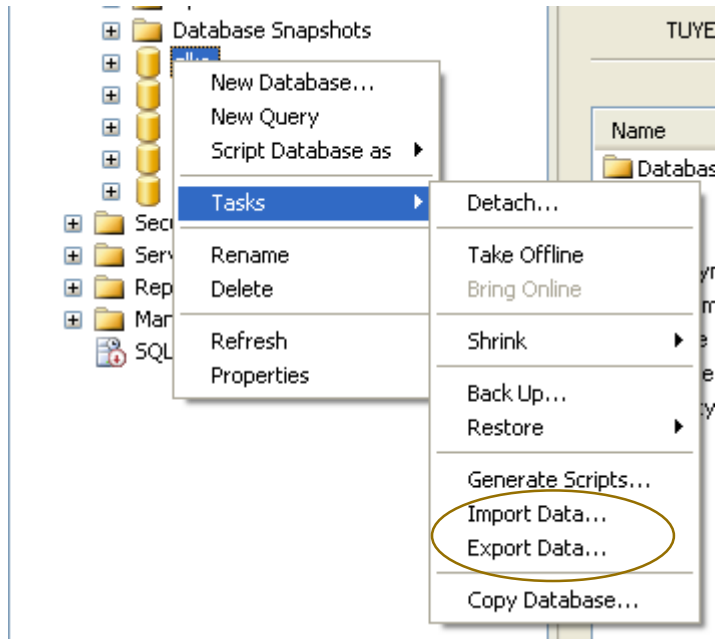
- USE Northwind

REVOKE SELECT

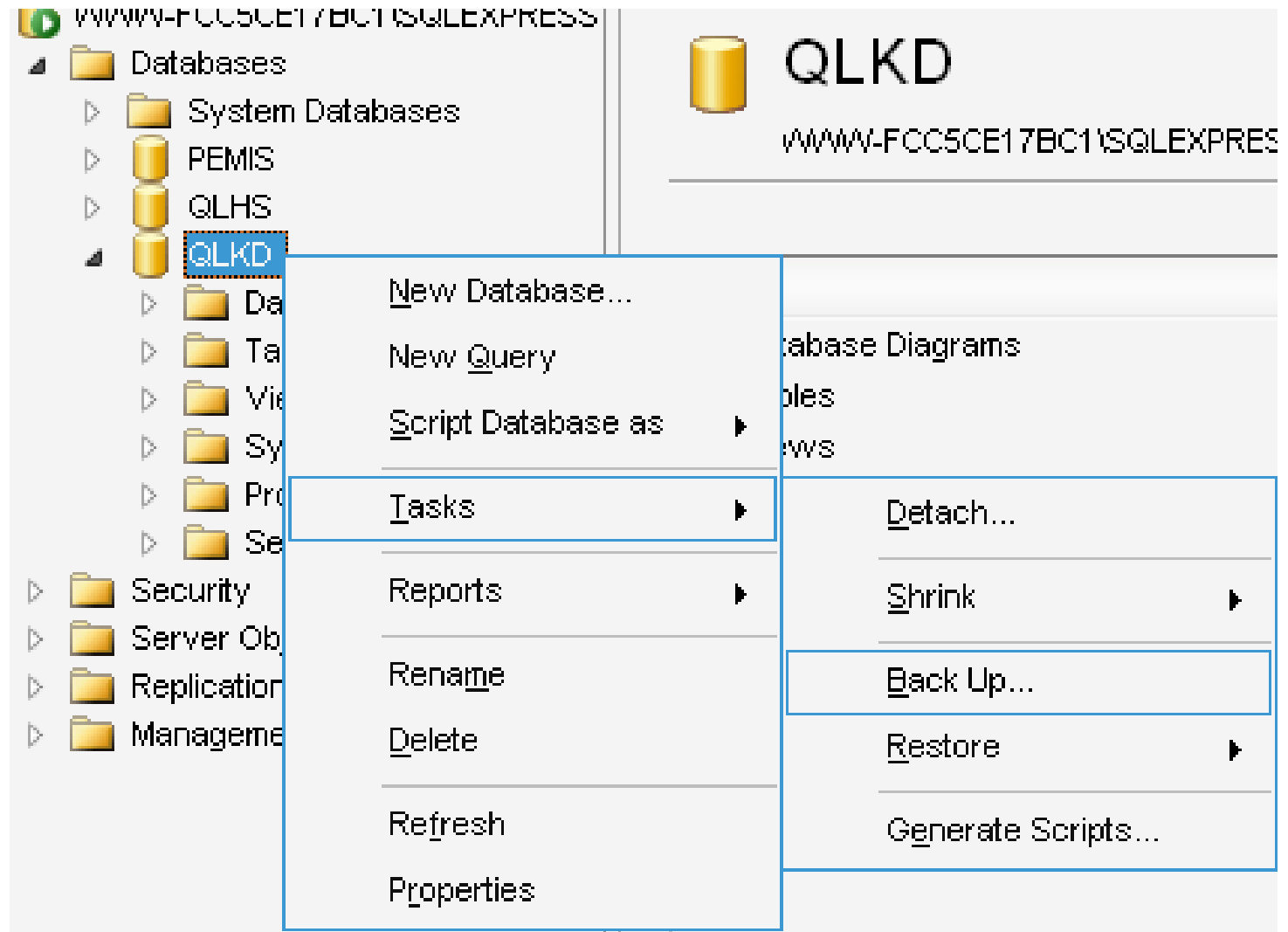
ON Customers

TO PUBLIC

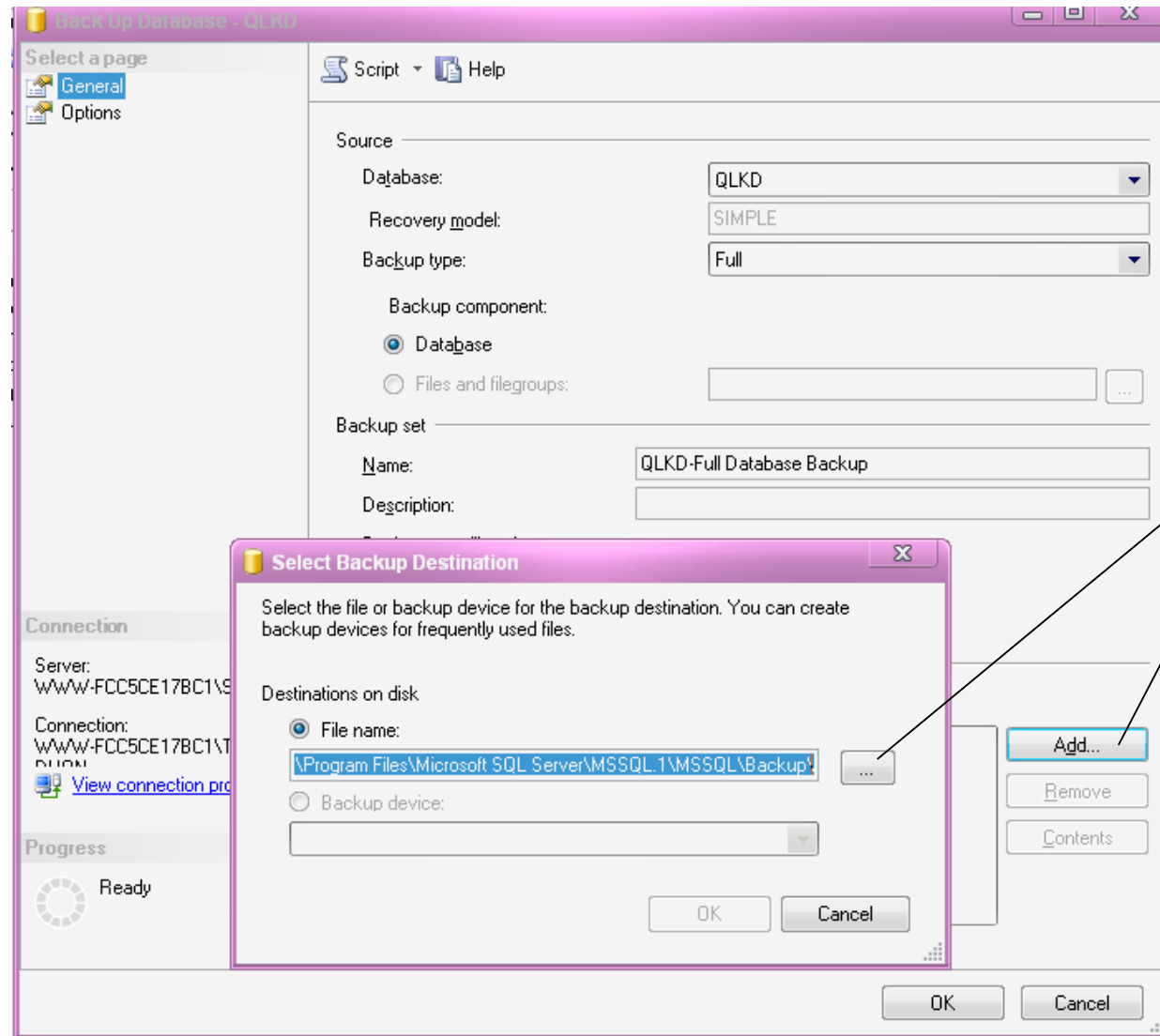
Bài 10: Nhập xuất CSDL



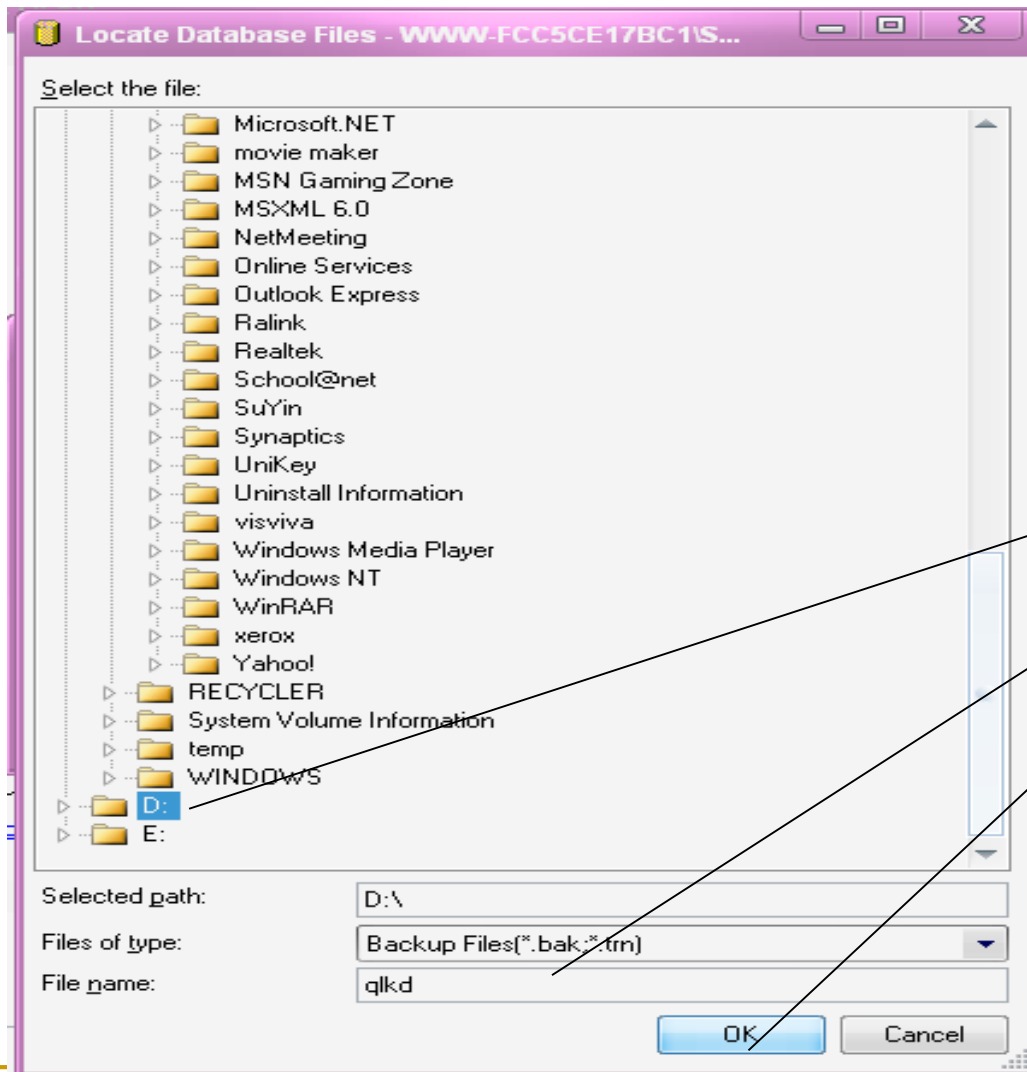
Bài 11: Sao lưu CSDL



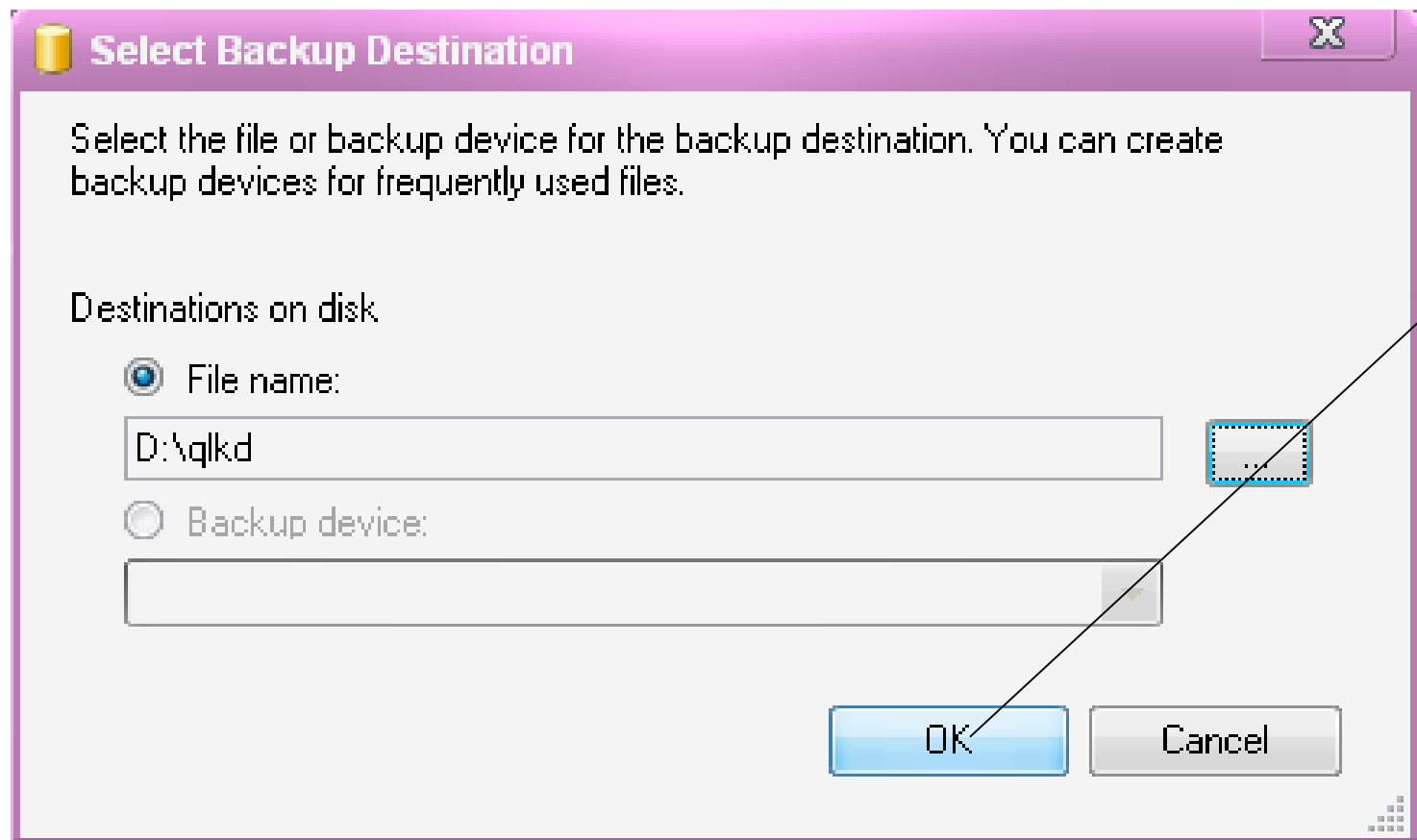
Bài 11: Sao lưu CSDL



Bài 11: Sao lưu CSDL



Bài 11: Sao lưu CSDL



Nhấp chọn