

CHƯƠNG 2

CÁC ĐỐI TƯỢNG TRONG SQL SERVER

Giảng viên: Lương Thị Hồng Lan

Nội dung

2.1 Cơ sở dữ liệu trong SQL Server

2.2. View

2.3. Chỉ mục

2.4. Con trỏ

1. Các kiểu dữ liệu trong SQL Server

Giới thiệu chung

- Kiểu dữ liệu (Data type) quy định về cấu trúc, miền giá trị của dữ liệu có thể nhập vào và tập các phép toán/toán tử có thể tác động lên miền giá trị đó
- Cần xác định kiểu dữ liệu thích hợp cho từng thuộc tính dữ liệu để đảm bảo tối ưu bộ nhớ trong quá trình sử dụng
 - Ví dụ:
 - *-Thuộc tính « Ngày Sinh »: kiểu dữ liệu ngày tháng*
 - *Thuộc tính « Họ Tên »: kiểu dữ liệu dạng chuỗi ký tự*
 - *Thuộc tính « Lương »: kiểu dữ liệu số*

Danh sách các kiểu DL trong SQL Server

Tên kiểu	Mô tả
INT	Như kiểu Integer
TINYTINT	Số nguyên có giá trị từ 0 đến 255.
SMALLINT	Số nguyên có giá trị từ -2^{15} đến $2^{15} - 1$
BIGINT	Số nguyên có giá trị từ -2^{63} đến $2^{63}-1$
NUMERIC (p, s)	Kiểu số với độ chính xác cố định.
DECIMAL (p, s)	Tương tự kiểu Numeric
FLOAT	Số thực từ $-1.79E+308$ đến $1.79E+308$
REAL	Số thực từ $-3.40E + 38$ đến $3.40E + 38$

Danh sách các kiểu DL trong SQL Server

Tên kiểu	Mô tả
MONEY	Kiểu tiền tệ
BIT	Kiểu bit (có giá trị 0 hoặc 1)
DATETIME	Kiểu ngày giờ (8 byte)
SMALLDATETIME	Kiểu ngày giờ (4 byte)
BINARY	Dữ liệu nhị phân với độ dài cố định (tối đa 8000 bytes)
VARBINARY	Dữ liệu nhị phân với độ dài chính xác (tối đa 8000 bytes)

Danh sách các kiểu DL trong SQL Server

- Tiền tệ: Money, SmallMoney
- Ngày giờ: DateTime (8 bytes - từ 01/01/1753 đến 31/12/9999), SmallDateTime (4 bytes – từ 01/01/1900 đến 6/6/2079)
- Chuỗi nhị phân: Binary, VarBinary, Image

Kiểu DL trong SQL

- **Ví dụ 1.1:** Định nghĩa bảng với kiểu dữ liệu được qui định cho các cột trong bảng NHANVIEN

```
CREATE TABLE  NHANVIEN
(
  MANV  NVARCHAR(10)  NOT NULL,
  HOTEN NVARCHAR(30) NOT NULL,
  GIOITINH BIT,
  NGAYSINH SMALLDATETIME,
  NOISINH NCHAR(50),
  HSLUONG DECIMAL(4,2),
  MADV    INT
)
```


2. Cơ sở dữ liệu trong SQL Server

Giới thiệu

- SQL server quản lý trực tiếp các CSDL, mỗi CSDL SQL Server sẽ quản lý các cấu trúc vật lý của nó
- Mỗi Server quản lý một danh sách các CSDL, tên các CSDL là duy nhất, không trùng nhau.
- Đặc điểm của việc quản trị CSDL:
 - *Để khai thác CSDL, client phải kết nối đến Server quản trị CSDL đó*
 - *Chỉ khai thác các CSDL có trong server*
 - *Không có các phương thức mở CSDL trực tiếp từ tệp tin*
 - *Client chỉ thực hiện khai thác theo quy định đã định sẵn trong CSDL*

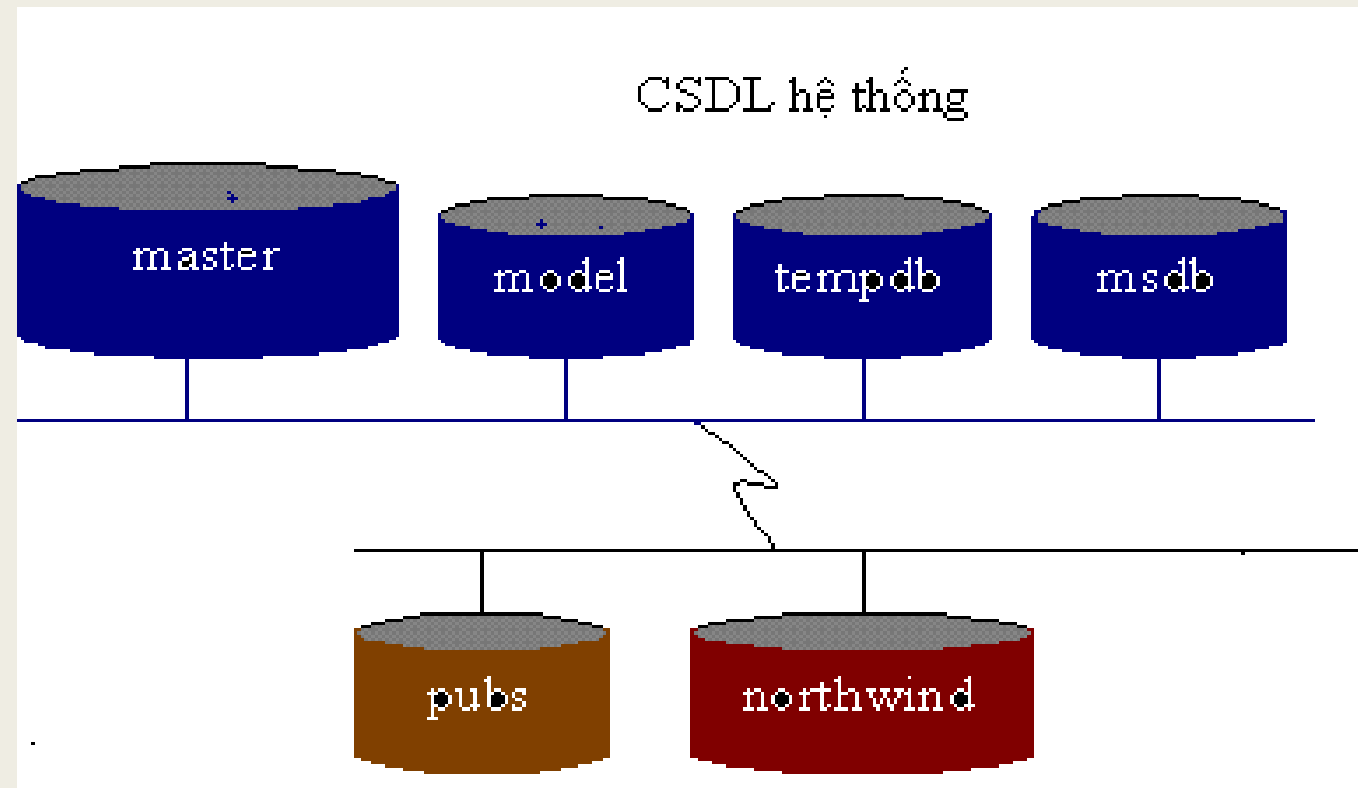
CƠ SỞ DỮ LIỆU TRONG SQL SERVER

- SQL SERVER có 3 kiểu cơ sở dữ liệu

Cơ sở dữ liệu hệ thống

Cơ sở dữ liệu mẫu

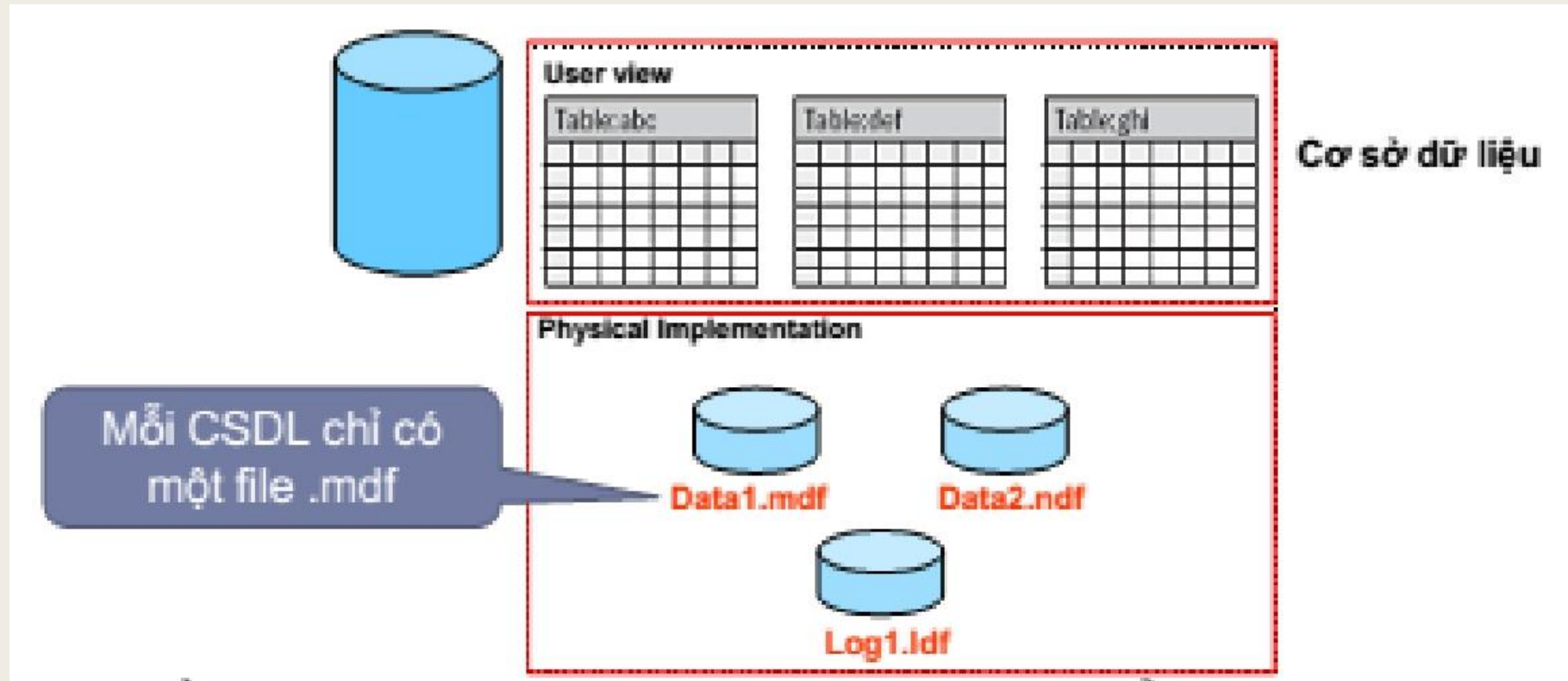
Cơ sở dữ liệu do người dùng định nghĩa



CSDL hệ thống trong SQL Server

Cơ sở dữ liệu	Mô tả
Master	Lưu tất cả những thông tin hệ thống: thông tin về các database khác trong hệ thống như vị trí của các data files, các login account và các thiết đặt cấu hình hệ thống của SQL Server.
Tempdb	Lưu tất cả những đối tượng (table, stored procedure..) được tạm thời tạo ra trong quá trình làm việc bởi user hay do bản thân SQL Server engine.
Model	CSDL mẫu để tạo ra các CSDL người dùng.
Msdb	được sử dụng cho SQL Server Agent để lập lịch các công việc và các cảnh báo (schedule alerts and jobs)
Resource	là một CSDL chỉ đọc chứa các object hệ thống mà được sử dụng trong SQL Server.

Cấu trúc CSDL trong SQL Server



- Về mặt logic, DL trong CSDL được tổ chức trong các đối tượng của CSDL
- Về mặt vật lí, CSDL được lưu trữ trên 2 hay nhiều tập tin

Cấu trúc vật lý của CSDL trong SQL Server

■ Mỗi database trong SQL Server gồm:

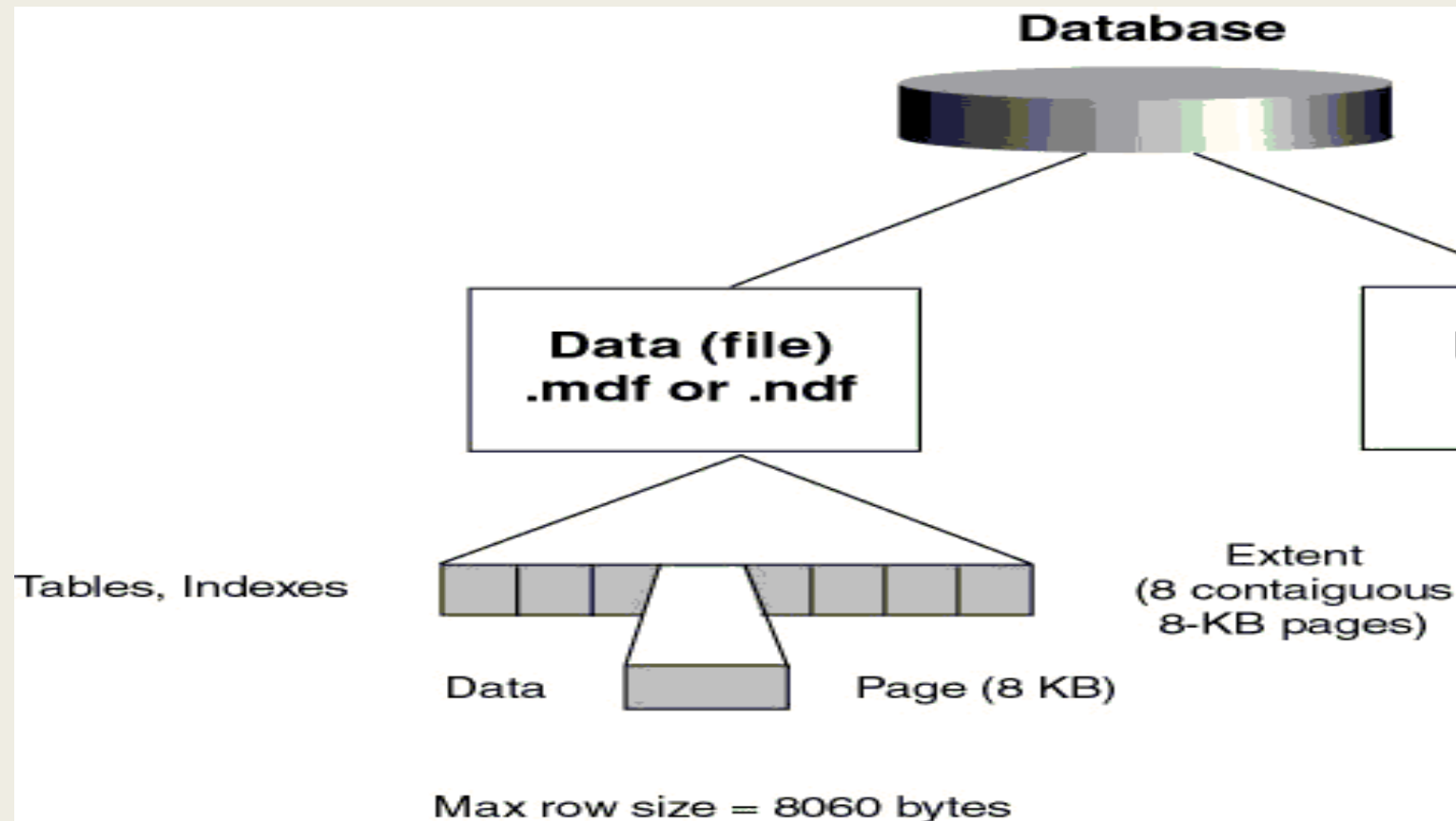
- Ít nhất một data file chính (primary)
- Có thể có thêm một hay nhiều data file phụ (secondary)
- ✓ – Một transaction log file

Cấu trúc vật lý của CSDL trong SQL Server

- Primary data file (*.mdf):
 - *file chính chứa data và những system tables*
- Secondary data file (*.ndf)
 - *file phụ thường chỉ được sử dụng khi CSDL được phân chia để chứa trên nhiều đĩa.*
- Transaction log file (.ldf)
 - *file ghi lại tất cả những thay đổi diễn ra trong một CSDL, chứa đầy đủ những thông tin để quay lui, backup hay phục hồi dữ liệu khi cần*

Cấu trúc vật lý của CSDL trong SQL Server

- Dữ liệu trong SQL SERVER được chứa trong các page. Mỗi page có dung lượng 8KB, 8 page liên tục tạo thành 1 extent

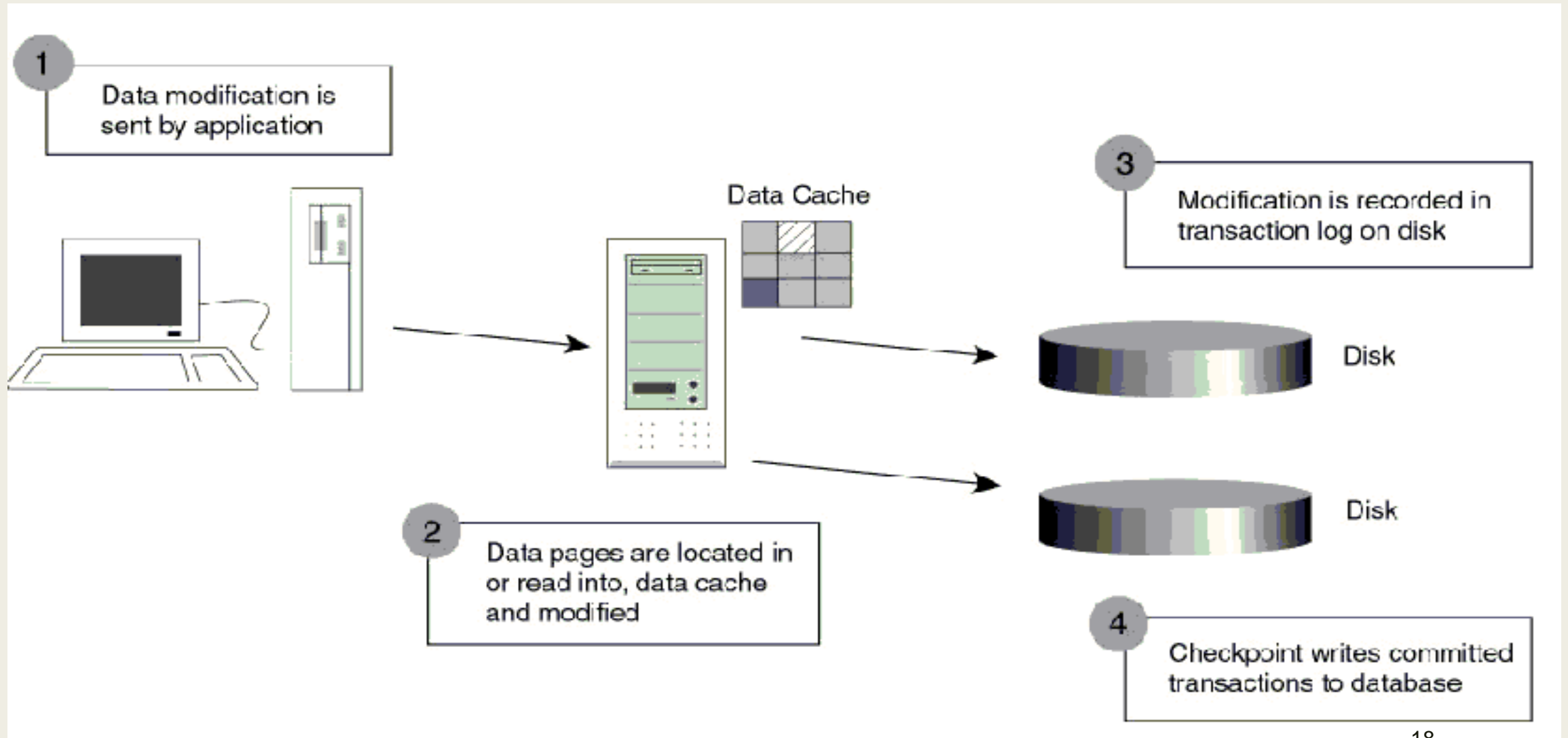


Cấu trúc vật lý của CSDL trong SQL Server

- Có hai loại phân đoạn (extent):
 - *Mixed Extent*: dùng để chứa data của nhiều table trong cùng một extent
 - *Uniform Extent*: dùng để chứa data của một bảng
- Đầu tiên, SQL Server dành các page trong Mixed Extent để chứa dữ liệu cho các bảng. Khi data tăng trưởng đến ngưỡng nào đó thì SQL Server dùng hẳn Uniform Extent để chứa data cho 1 bảng đó.

Quy trình thao tác dữ liệu

- Transaction Log: dùng để ghi lại các thay đổi diễn ra trong database



Cấu trúc logic của CSDL SQL SERVER

- Được tổ chức thành những objects: tables, views, stored procedures, indexes, constraints.... thường có bắt đầu bằng chữ *sys* hay *sp*
- Một số Sytem objects:

System Stored Procedure	Ứng dụng
<i>Sp_help</i> [<i>object</i>]	Cung cấp thông tin về một database object (table, view...) hay một data type.
<i>Sp_helpdb</i> [<i>database</i>]	Cung cấp thông tin về một database cụ thể nào đó.
<i>Sp_monitor</i>	Cho biết độ bận rộn của SQL Server
<i>Sp_spaceused</i> [<i>object</i> , <i>'updateusage'</i>]	Cung cấp thông tin về các khoảng trống đã được sử dụng cho một object nào đó
<i>Sp_who</i> [<i>login</i>]	Cho biết thông tin về một SQL Server user

3. Ngôn ngữ định nghĩa dữ liệu

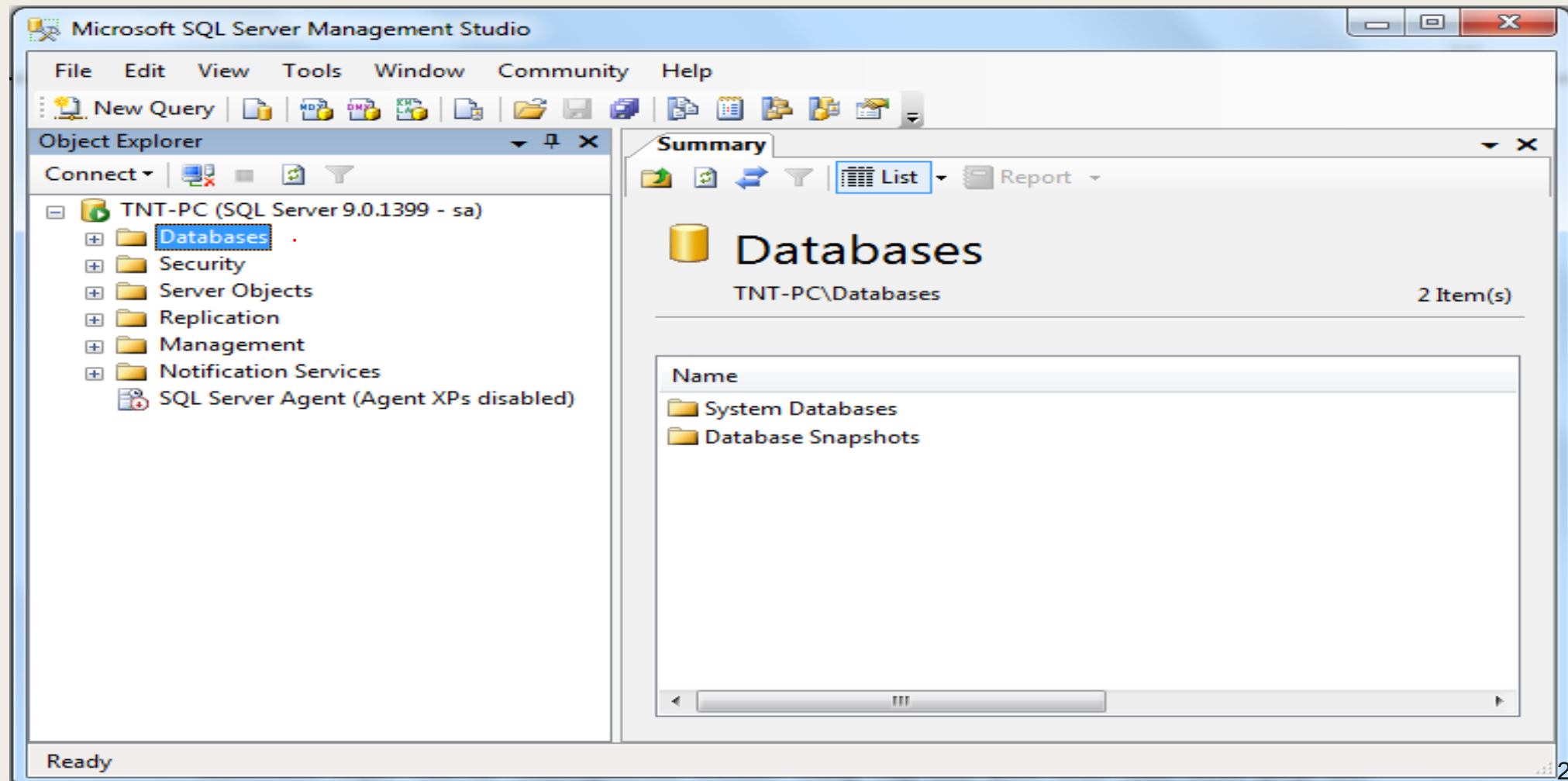
Tạo CSDL

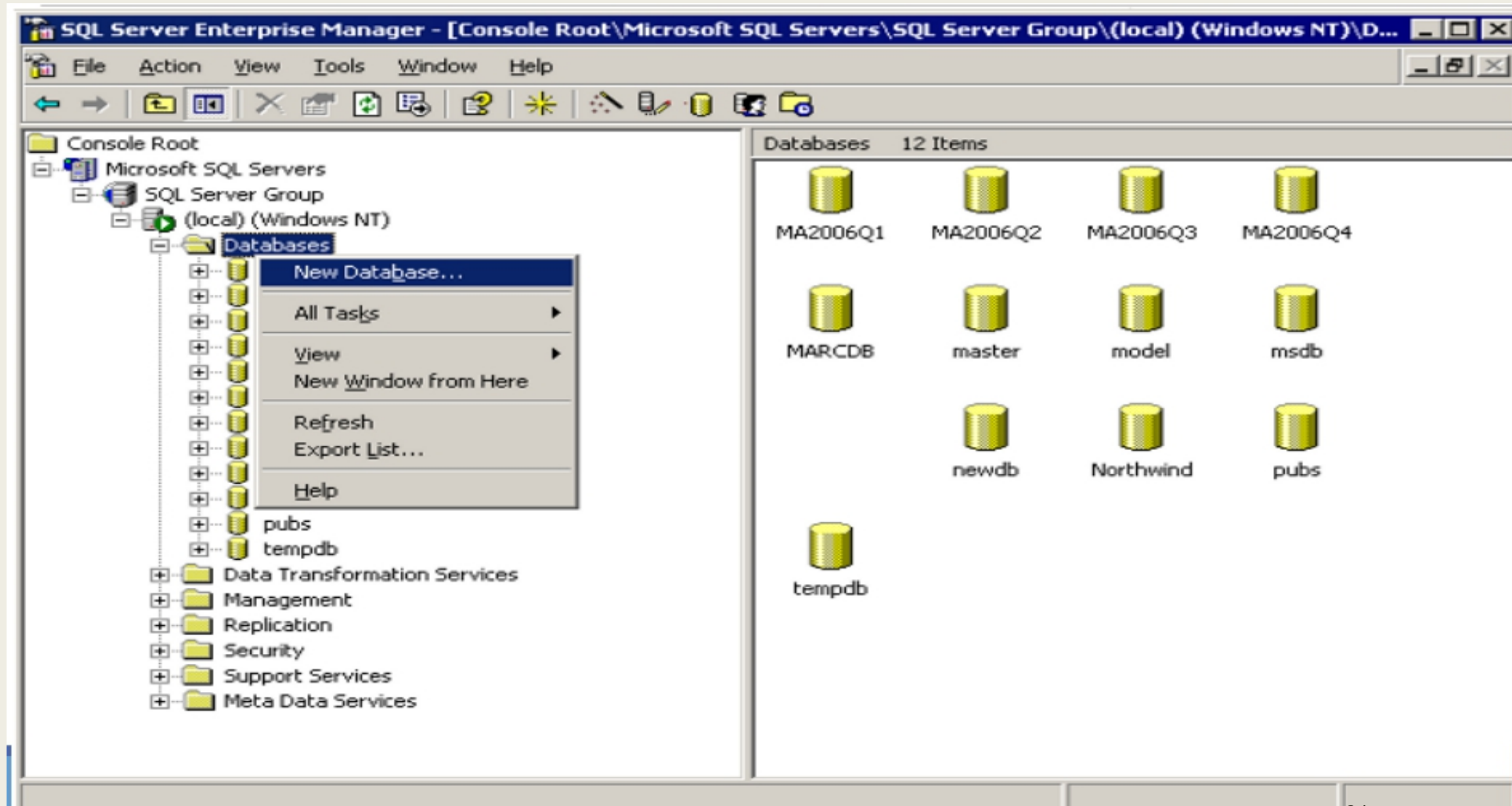
- Có thể tạo CSDL bằng:
 - Câu lệnh *CREATE DATABASE*
 - *SQL Server Management Studio*
- Mục đích sử dụng CSDL cho bài toán:
 - *Tên CSDL*
 - *Người sở hữu hoặc người tạo CSDL*
 - *Kích thước*
 - *Các tập tin và các nhóm tập tin được sử dụng để lưu trữ CSDL*

Tạo CSDL bằng SQL Server Management Studio

- Mở SQL Server Management Studio
- Kết nối bằng tài khoản người quản trị CSDL
- Kích chuột phải vào « Databases » và chọn « NewDatabase »
- Nhập tên CSDL
- Kích đúp chuột vào CSDL vừa tạo để mở hộp thoại chi tiết CSDL
- Ở thẻ General, ta có các thông tin chung về CSDL
- Kích chuột vào các thẻ và hộp thích hợp, thực hiện các thay đổi và nhấn OK

- ## ■ Cửa sổ Microsoft SQL Server Management Studio sau khi đăng nhập thành công





New Database

Select a page

General

Options

Filegroups

Script

Help

Database name:

GFI MailEssentials

Owner:

<default>

...

☐ Use full-text indexing

Database files:

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth
GFI MailEsse...	Data	PRIMARY	2	By 1 MB, unrestricted growth
GFI MailEsse...	Log	Not Applicable	1	By 10 percent, unrestricted growth

Connection

Server:

ISA

Connection:

sa

View connection properties

Progress

Ready

Add

Remove

OK

25Cancel

Tạo CSDL bằng T-SQL

```
CREATE DATABASE Tên_CSDL
```

```
On Primary
```

```
(
```

```
Name=logical_file_name ,
```

```
FileName= 'os_file_name' đ / đ an
```

```
Size=size [ KB | MB | GB | TB ] ,
```

```
MaxSize=max_size [ KB | MB | GB | TB ] | UNLIMITED],
```

```
FileGrowth=growth_increment
```

-- Tên file logic

-- Tên file vật lý

-- Kích thước tập tin ban đầu

-- Kích thước tối đa

-- Kích thước tăng trưởng

```
)
```

```
Log On
```

```
(
```

```
Name= logical_file_name,
```

```
FileName= 'os_file_name',
```

```
Size=size,
```

```
MaxSize=max_size,
```

```
FileGrowth= growth_increment
```

```
);
```

Tạo CSDL bằng T-SQL (Tiếp)

- Tên CSDL: tên của CSDL
- On Primary: mô tả primary file của CSDL
 - *Name: tên primary file*
 - *File name: đường dẫn của primary file*
 - *Size: kích thước của primary file*
 - *MaxSize: kích thước lớn nhất của primary file*
 - *Filegrowth: chỉ định độ tăng nào được sử dụng đối với việc tự động phát triển của primary file*
- Log on: mô tả nhật kí của CSDL

- Ví dụ 1: Tạo CSDL TEST dành ra 20MB lúc đầu cho phần dữ liệu và 5MB cho phần nhật ký giao tác . Các tập tin có thể phát triển lên đến 100 MB với phần dữ liệu, 15 MB với phần nhật ký giao tác. Tốc độ tăng trưởng của file dữ liệu là 5MB, của file nhật ký giao tác là 10%. Các file được lưu trữ trong thư mục D:\ BT SQL\

CREATE DATABASE TEST

On Primary

(Name=TestData,

Filename= 'D:\BT SQL\TestDat.mdf',

Size=20 MB,

MaxSize=100MB)

Log On

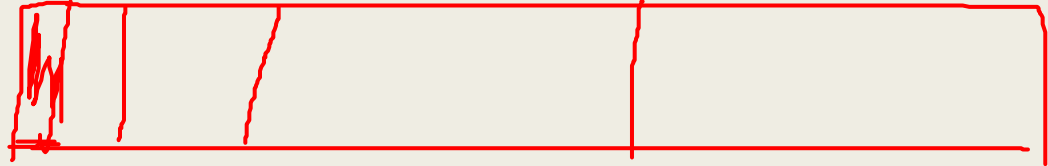
(Name=TestLog,

FileName= 'D:\BT SQL\TestLog.ldf',

Size=5MB,

MaxSize=15MB

)



- Chú ý: để xem lại thông tin về CSDL, sử dụng thủ tục

sp_helpdb

- Ví dụ:

sp_helpdb TEST

Hiệu chỉnh thuộc tính của tập tin

► Cú pháp:

```
ALTER DATABASE database_name
  MODIFY FILE (
    NAME = logical_file_name ,
    FILENAME = 'os_file_name' ,
    SIZE = size ,
    MAXSIZE = max_size ,
    FILEGROWTH = size
  )
```

Hiệu chỉnh thuộc tính của tập tin

- Ví dụ 1: Tăng kích thước ban đầu của CSDL TEST lên thành 25 MB.
- Ví dụ 2: hủy thuộc tính tự tăng trưởng của CSDL TEST

Hiệu chỉnh thuộc tính của tập tin

- Ví dụ 1: Tăng kích thước ban đầu của CSDL TEST lên thành 25 MB.

- Cú pháp:.

```
ALTER DATABASE Database name
```

```
MODIFY FILE (NAME = logical_file_name, SIZE = size)
```

- Thực hiện

```
ALTER DATABASE TEST
```

```
MODIFY FILE (NAME = TestData, SIZE = 25 MB)
```

Hiệu chỉnh thuộc tính của tập tin

- Ví dụ 1: hủy thuộc tính tự tăng trưởng của CSDL TEST .

- Cú pháp:.

```
ALTER DATABASE database_name  
MODIFY FILE (NAME = ogical_file_name, FILEGROWTH = size
```

- Thực hiện

```
ALTER DATABASE TEST  
MODIFY FILE (NAME = TestData, FILEGROWTH = 0)
```

Hiệu chỉnh thuộc tính của tập tin

- Cú pháp giảm kích thước tập tin của CSDL

```
USE database_name  
DBCC SHRINKFILE (NAME = logical_file_name , size)
```

- Giảm kích thước tập tin dữ liệu của CSDL QLSV xuống còn 20MB

```
USE QLSV  
DBCC SHRINKFILE (NAME = QLSV_data , 20)
```

Bổ sung thêm tập tin dữ liệu và tập tin nhật ký cho CSDL

■ Cú pháp

```
ALTER DATABASE database_name  
ADD FILE  
(  
    NAME = logical_file_name ,  
    FILENAME = 'os_file_name' ,  
    SIZE = size ,  
    MAXSIZE = max_size ,  
    FILEGROWTH = max_size  
)
```

► Ví dụ:

```
ALTER DATABASE QLSV
ADD FILE (
    NAME = QLSV_data2,
    filename = D:\qlsv_data2.ndf,
    size = 2,
    maxsize = 5,
    filegrowth = 2)
```

Gỡ bỏ một tập tin khỏi CSDL

- Cú pháp

ALTER DATABASE database_name

REMOVE FILE logical_name

- Chú ý: Tập tin phải trống trước khi gỡ bỏ

=> sử dụng **DBCC SHRINKFILE** với tùy chọn **EMPTYFILE** để làm trống tập tin

Gỡ bỏ một tập tin khỏi CSDL

- Gỡ bỏ tập tin QLSV_data khỏi CSDL QLSV

```
USE QLSV GO
DBCC SHRINKFILE (
    QLSV_data, EMPTYFILE
)
GO
ALTER DATABASE QLSV
    REMOVE FILE QLSV_data
```

Đổi tên CSDL

- Cú pháp 1:

```
ALTER DATABASE database_name  
    MODIFY NAME = new_database_name
```

- Ví dụ:

```
ALTER DATABASE baitap  
    MODIFY NAME = baitap1
```


Đổi tên CSDL (tiếp)

- Cú pháp 2: Thực thi thủ tục lưu trữ hệ thống `sp_renamedb` để đổi tên CSDL

`sp_renamedb` 'old_name', 'new_name'

- Ví dụ: `sp_renamedb` 'baitap', 'baitap1'
- Chú ý: khi muốn đổi tên CSDL, phải tắt hết các ứng dụng, các cửa sổ lệnh, view.. hiện thời đang truy cập vào CSDL

Tạo CSDL- Dùng SQL Server Managenent Studio

- ▶ Thực hiện theo các bước sau:
 - ▶ Mở SQL Server Management Studio
 - ▶ Right-click lên trên "database" và chọn "New Database"
 - ▶ Nhập tên CSDL
 - ▶ Kích phải chuột vào tên CSDL vừa tạo và chọn Properties hoặc (kích đúp chuột).
 - ▶ Ở thẻ General ta có các thông tin chung của CSDL (tên, kích thước...)
 - ▶ Để thay đổi thuộc tính các tập tin CSDL, kích chuột vào thẻ và hộp thích hợp, thực hiện các thay đổi và nhấn OK.

Tên file dữ liệu

Đường dẫn vật lý của file dữ liệu

Tùy chọn file tự động tăng trưởng được bật

SQL Server Enterprise Manager - Help

Server: QLsinhvien

Database: QLsinhvien

☒ Full-text indexing

Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth	Path
QLsinhvien	File ...	PRIMARY	2	By 1 MB, unrestricted growth	d:\SETUP\Microsoft SQL Server\MSS
QLsinhvien_log	Log	Not Applicable	1	By 10 percent, unrestricted growth	d:\SETUP\Microsoft SQL Server\MSS

Buttons: Add, Remove

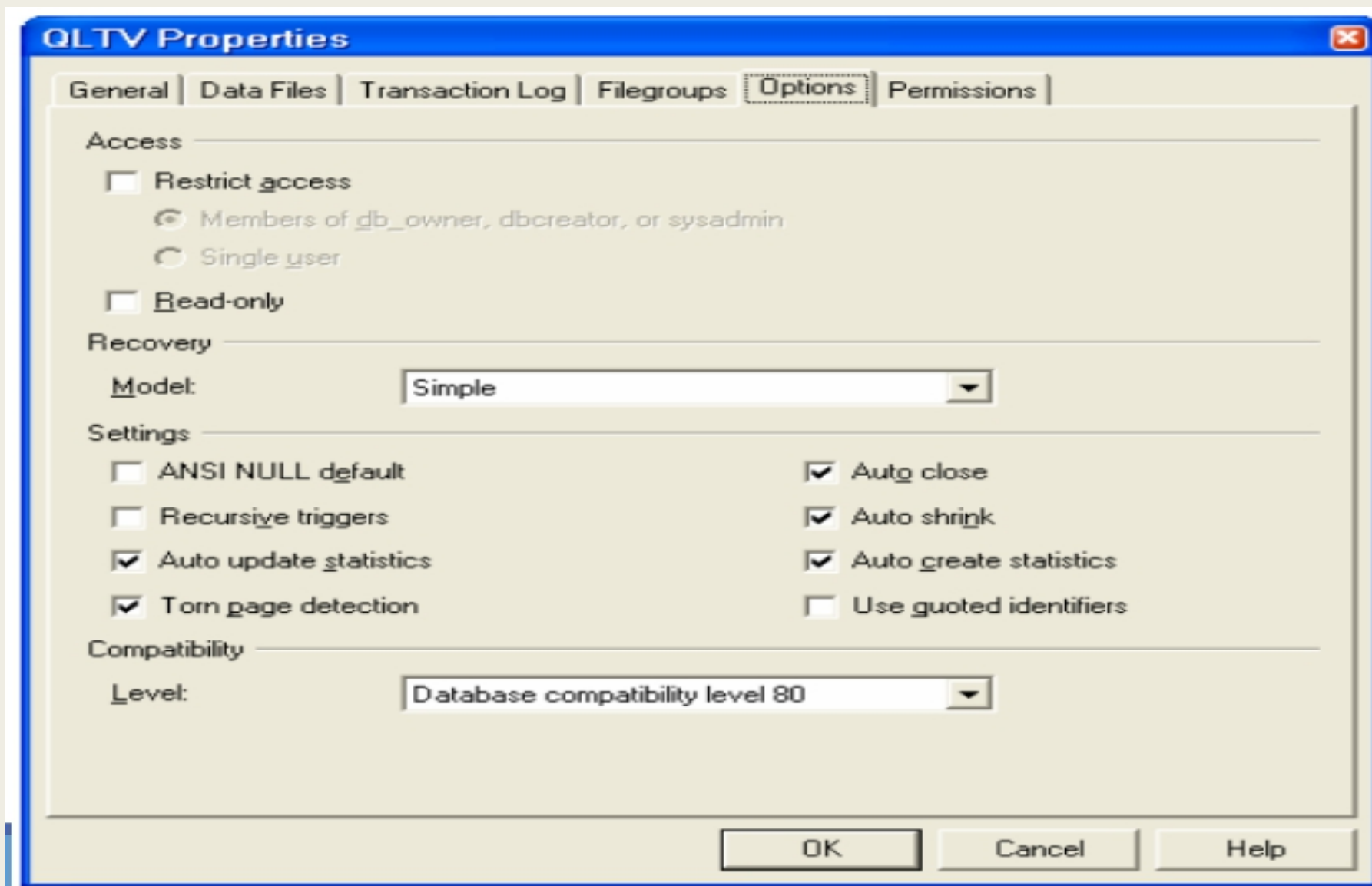
Xoá CSDL

- Xoá bằng SQL Server Management Studio
 - *Trong Database, kích chuột phải vào CSDL muốn xoá rồi chọn « Delete »*
 - Sau đó chọn « Yes »
- Xoá bằng câu lệnh T-SQL
 - *Cú pháp: Drop DATABASE Tên_CSDL*
 - *Ví dụ: Drop DATABASE QL SV*

Sửa tham số CSDL

- Sửa bằng SQL Server Management Studio
 - *Trong Database, kích chuột phải vào CSDL muốn sửa rồi chọn « Properties »*
 - *Sửa các tham số cần thiết sau đó chọn « OK »*
- Sửa bằng câu lệnh T-SQL
 - *Sử dụng câu lệnh ALTER DATABASE TênCSDL*

Sửa tham số CSDL



2.2 View (Khung nhìn)

Khái niệm khung nhìn (View)

- Là đối tượng thuộc CSDL
- Là một bảng ảo có cấu trúc như một bảng: gồm dòng và cột.
- Có nội dung được xác định từ một truy vấn nhằm truy xuất và hiển thị dữ liệu từ các bảng trong CSDL.
- Không lưu trữ dữ liệu mà chỉ giúp quan sát dữ liệu được truy vấn từ các bảng thông qua câu lệnh truy vấn SELECT
- Người dùng có thể áp dụng ngôn ngữ thao tác dữ liệu trên các View giống như trên các Table

Khái niệm khung nhìn (View)

- Mục đích sử dụng khung nhìn :
 - Để tập trung trên dữ liệu được xác định
 - Để đơn giản hóa thao tác dữ liệu
 - Để tùy biến dữ liệu
 - Để xuất dữ liệu
 - Để bảo mật dữ liệu

Ví dụ View

- CSDL QLSV có 3 bảng:

Bảng SinhVien

MaSV	HoTen	GioiTinh	NgaySinh	DiaChi
0	Đào Thị Thúy	Nữ	1994-04-27	127 Hoàng Hoa Thám, Hà Nội
4	Nguyễn Văn A	Nam	1995-09-15	175 Tây Sơn
6	Nguyễn Văn B	Nam	1994-09-14	120 Phố Huế

Bảng MonHoc

MaMon	TenMon	MoTa
1	Hệ Quản trị CSDL	NULL
2	Thuật toán ứng dụng	NULL

Bảng KETQUA

MaSV	MaMon	Diem
0	1	6
0	2	8
4	1	2
4	2	8

Cần xem điểm thi của các sinh viên ?

Ví dụ khung nhìn (tiếp)

- **Ví dụ:** Tạo khung nhìn KetQuaThi lấy dữ liệu từ 3 bảng để quan sát dữ liệu dễ dàng hơn
 - Chỉ cần quan sát dữ liệu trong view KetQuaThi thay vì quan sát dữ liệu từ 3 bảng

MaSV	HoTen	MaMon	TenMon	Diem
0	Đào Thị Thúy	1	Hệ Quản trị CSDL	6
0	Đào Thị Thúy	2	Thuật toán ứng dụng	8
4	Nguyễn Văn A	1	Hệ Quản trị CSDL	2
4	Nguyễn Văn A	2	Thuật toán ứng dụng	8
6	Nguyễn Văn B	NULL	NULL	NULL

Ưu điểm

- Bảo mật dữ liệu: người dùng được cấp phát quyền trên các khung nhìn với phần dữ liệu mà người dùng được phép → Hạn chế việc người dùng có thể truy cập trực tiếp dữ liệu
- Đơn giản hoá các thao tác truy vấn dữ liệu: người dùng chỉ cần thực hiện truy vấn dữ liệu đơn giản từ khung nhìn thay vì phải đưa ra những truy vấn trên nhiều bảng
- Tập trung và đơn giản hoá dữ liệu: Cung cấp cho người dùng cấu trúc đơn giản, dễ hiểu hơn về dữ liệu, tập trung hơn trên những dữ liệu cần thiết.
- Độc lập dữ liệu: cho phép người sử dụng cái nhìn về dữ liệu độc lập với cấu trúc với các bảng trong CSDL (Bảng nếu thay đổi 1 phần về cấu trúc cũng không làm ảnh hưởng đến khung nhìn)

Nhược điểm

- Thông qua khung nhìn có thể thực hiện được thao tác bổ sung và cập nhật dữ liệu cho bảng cơ sở nhưng chỉ hạn chế với những khung nhìn đơn giản.
- Nếu khung nhìn được định nghĩa bởi một truy vấn phức tạp thì thời gian thực hiện truy vấn trên khung nhìn sẽ lớn

Tạo khung nhìn

- Cú pháp:
- **CREATE VIEW** tên_khung_nhìn [danh sách tên cột]
AS
 Câu_lệnh_**SELECT**

Ví dụ khung nhìn

- Tạo khung nhìn KetQuaThi lấy dữ liệu về MaSV, HoTen, MaMon, TenMon, Diem từ 3 bảng SinhVien, MonHoc, KETQUA

MaSV	HoTen	MaMon	TenMon	Diem
0	Đào Thị Thúy	1	Hệ Quản trị CSDL	6
0	Đào Thị Thúy	2	Thuật toán ứng dụng	8
4	Nguyễn Văn A	1	Hệ Quản trị CSDL	2
4	Nguyễn Văn A	2	Thuật toán ứng dụng	8
6	Nguyễn Văn B	NULL	NULL	NULL

Ví dụ khung nhìn (Tiếp)

■ **CREATE VIEW** KetQuaThi

AS

SELECT SinhVien.MaSV, HoTen, MonHoc.MaMon, TenMon,
KETQUA. Diem

FROM SinhVien

left join KETQUA **on** SinhVien.MaSV = KETQUA.MaSV

left join KETQUA **on** KETQUA.MaMon = MonHoc.MaMon

Các cột của khung nhìn là MaSV, HoTen, MaMon, TenMon, Diem

Ví dụ khung nhìn (Tiếp)

- Tạo khung nhìn ViewSinhVien lấy dữ liệu về MaSV, HoTen, Tuoi từ bang SinhVien
- CREATE VIEW ViewSinhVien (MaSV, HoTen, Tuoi)

AS

```
SELECT MaSV, HoTen, DATEDIFF(yyyy, NgaySinh, getdate())  
FROM SinhVien
```

Nếu một thuộc tính trong View được xây dựng từ một biểu thức (VD: Tuoi) thì bắt buộc phải đặt tên cho thuộc tính đó

Nguyên tắc khi tạo khung nhìn

- Tên khung nhìn và tên cột của khung nhìn phải tuân theo quy tắc định danh và phải duy nhất đối với mỗi người sử dụng
- Không thể quy định ràng buộc, quy tắc, chỉ mục cho khung nhìn
- Phải đặt tên cho các cột của khung nhìn khi:
 - *Có ít nhất một cột được sinh ra bởi biểu thức, hàm hay hằng*
 - *Tồn tại hai hay nhiều cột trong kết quả của câu lệnh SELECT có cùng tiêu đề cột*
 - *Muốn thay đổi tên cột trong khung nhìn khác với tên cột của bảng cơ sở*

Sử dụng khung nhìn

- Khung nhìn sau khi tạo có thể được sử dụng để truy vấn như với một bảng thông thường.
- Ví dụ:

```
select * from KetQuaThi
```

Sử dụng khung nhìn (tiếp)

- Ví dụ: sử dụng khung nhìn KetQuaThi để hiển thị thông tin điểm thi môn Hệ Quản trị CSDL bao gồm MaSV, HoTen, Diem

```
select MaSV, HoTen, Diem
from KetQuaThi
where TenMon = N'Hệ Quản trị CSDL'
```

- Để hiển thị thông tin trên với truy vấn trong bảng:

```
select SinhVien.MaSV, HoTen, Diem
from SinhVien, KETQUA, MonHoc
where SinhVien.MaSV = KETQUA.MaSV
and KETQUA.MaMon = MonHoc.MaMon
and MonHoc.TenMon = N'Hệ Quản trị CSDL'
```

Phân loại khung nhìn

- ❑ Khung nhìn chỉ đọc (**Read only view**): View này chỉ dùng để xem, truy vấn dữ liệu với câu lệnh **SELECT**
- ❑ Khung nhìn có thể cập nhật (**Updatable view**):
 - Dùng để xem, truy vấn dữ liệu (**SELECT**)
 - Có thể tiến hành thực hiện các thao tác cập nhật (**UPDATE**), bổ sung (**INSERT**) và xóa (**DELETE**) dữ liệu trên các bảng cơ sở thông qua View

Cập nhật, bổ sung, xóa dữ liệu

■ Để có thể thực hiện thao tác bổ sung, cập nhật và xóa dữ liệu, câu lệnh `SELECT` khi tạo view phải thỏa mãn:

- Các thành phần trong danh sách chọn của câu lệnh `SELECT` phải là các cột trong các bảng cơ sở
- Không chứa từ khóa `DISTINCT`, `TOP`
- Không chứa mệnh đề `GROUP BY` và `HAVING`
- Không chứa toán tử `UNION`
- Không chứa các hàm tập hợp (aggregate function: `sum`, `avg`, `count`, `min`, `max`)
- Không chứa các biểu thức tính toán

Thêm dữ liệu qua khung nhìn

- Có khung nhìn ViewSinhVien(MaSV, HoTen, Tuoi) đã tạo, thực hiện thêm mới một sinh viên thông qua khung nhìn

```
insert into ViewSinhVien(HoTen) values (N'Nguyễn Thị Tuyên')
```

HoTen	NgaySinh	DiaChi	GioiTinh	MaSV	Email
Đào Thị Thúy	1994-04-27	127 Hoàng Hoa Thám, Hà Nội	Nữ	0	NULL
Nguyễn Văn A	1995-09-15	175 Tây Sơn	Nam	4	NULL
Nguyễn Văn B	1994-09-14	120 Phố Huế	Nam	6	NULL
Nguyễn Thị Tuyên	NULL	NULL	Nam	7	NULL

Bản ghi tương ứng được thêm vào bảng SinhVien

Cập nhật dữ liệu qua khung nhìn

- Có thể sửa dữ liệu thông qua khung nhìn, ví dụ:

```
update ViewSinhVien  
set HoTen = N'Nguyễn Thị Minh Tuyên'  
where HoTen = N'Nguyễn Thị Tuyên'
```

HoTen	NgaySinh	DiaChi	GioiTinh	MaSV	Email
Đào Thị Thúy	1994-04-27	127 Hoàng Hoa Thám, Hà Nội	Nữ	0	NULL
Nguyễn Văn A	1995-09-15	175 Tây Sơn	Nam	4	NULL
Nguyễn Văn B	1994-09-14	120 Phố Huế	Nam	6	NULL
Nguyễn Thị Minh Tuyên	NULL	NULL	Nam	7	NULL

Bản ghi tương ứng được cập nhật trong bảng SinhVien

Xóa dữ liệu qua khung nhìn

- Có thể xóa dữ liệu thông qua khung nhìn, ví dụ:

```
delete from ViewSinhVien  
where HoTen = N'Nguyễn Thị Minh Tuyên'
```

- Bản ghi tương ứng được xóa khỏi bảng SinhVien

HoTen	NgaySinh	DiaChi	GioiTinh	MaSV	Email
Đào Thị Thúy	1994-04-27	127 Hoàng Hoa Thám, Hà Nội	Nữ	0	NULL
Nguyễn Văn A	1995-09-15	175 Tây Sơn	Nam	4	NULL
Nguyễn Văn B	1994-09-14	120 Phố Huế	Nam	6	NULL

Xóa dữ liệu qua khung nhìn

- Có thể xóa dữ liệu thông qua khung nhìn, ví dụ:

```
delete from ViewSinhVien  
where HoTen = N'Nguyễn Thị Minh Tuyên'
```

- Bản ghi tương ứng được xóa khỏi bảng SinhVien

HoTen	NgaySinh	DiaChi	GioiTinh	MaSV	Email
Đào Thị Thúy	1994-04-27	127 Hoàng Hoa Thám, Hà Nội	Nữ	0	NULL
Nguyễn Văn A	1995-09-15	175 Tây Sơn	Nam	4	NULL
Nguyễn Văn B	1994-09-14	120 Phố Huế	Nam	6	NULL

Cập nhật, bổ sung, xóa dữ liệu ...

- Trong trường hợp khung nhìn được tạo từ phép nối (JOIN) trên nhiều bảng:
 - Có thể thực hiện được thao tác bổ sung (INSERT) hoặc cập nhật (UPDATE) dữ liệu nếu thao tác này chỉ tác động đến đúng một bảng
 - Không thể thực hiện câu lệnh DELETE trong trường hợp này

Cập nhật, bổ sung, xóa dữ liệu ...

- **Ví dụ:** khung nhìn KetQuaThi(MaSV, Hoten, MaMon, TenMon, Diem) được tạo từ 3 bảng SinhVien, MonHoc và KETQUA

- Câu lệnh sau thêm một bản ghi vào bảng SinhVien

```
insert into KetQuaThi(HoTen) values (N'Mai Thu Trang')
```

- Câu lệnh sau thêm một bản ghi vào bảng MonHoc

```
insert into KetQuaThi(TenMon) values (N'Học Máy')
```

- Câu lệnh sau không thực hiện được vì tác động đến nhiều bảng cơ sở

```
insert into KetQuaThi(MaSV, MaMon, Diem) values (6,1,10)
```

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.

- Cú pháp:

ALTER VIEW tên_khung_nhìn [danh sách tên cột]

AS

Câu_lệnh_**SELECT**

Sửa đổi khung nhìn (tiếp)

- Ví dụ: khung nhìn ViewSinhVien

```
CREATE VIEW ViewSinhVien(MaSV, HoTen, Tuổi)
as
SELECT MaSV, HoTen, DATEDIFF(yyyy, NgaySinh, getDate())
FROM SinhVien
```

- Có thể được sửa lại như sau:

```
ALTER VIEW ViewSinhVien(MaSV, HoTen, Tuổi, GioiTinh)
as
SELECT MaSV, HoTen, DATEDIFF(yyyy, NgaySinh, getDate()), GioiTinh
From SinhVien
```

Xóa khung nhìn

- Cú pháp:

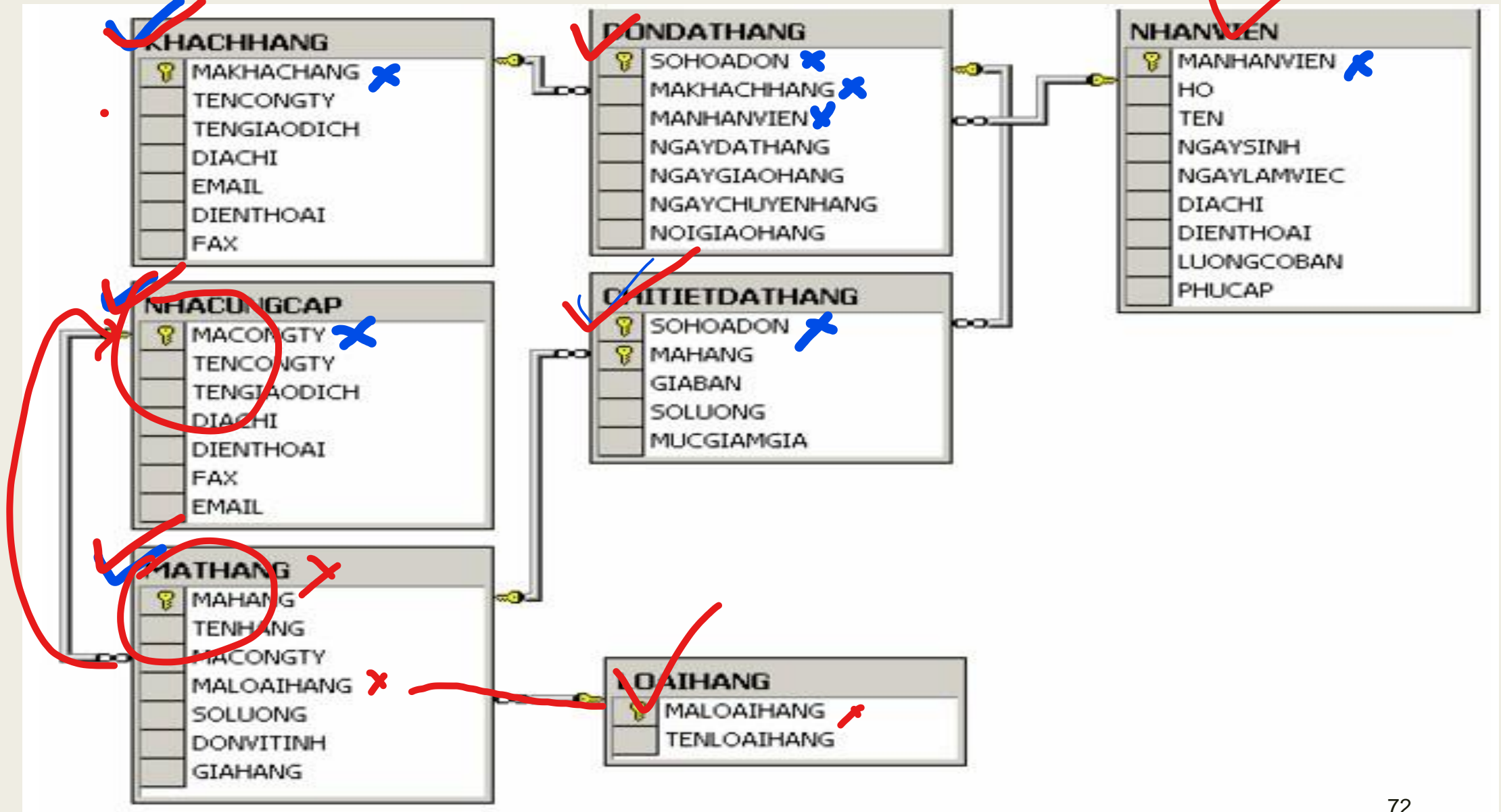
`DROP VIEW` tên_khung_nhìn

- Ví dụ:

`DROP VIEW` ViewSinhVien

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

Bài tập: Tạo một số view từ CSDL sau



Bài tập

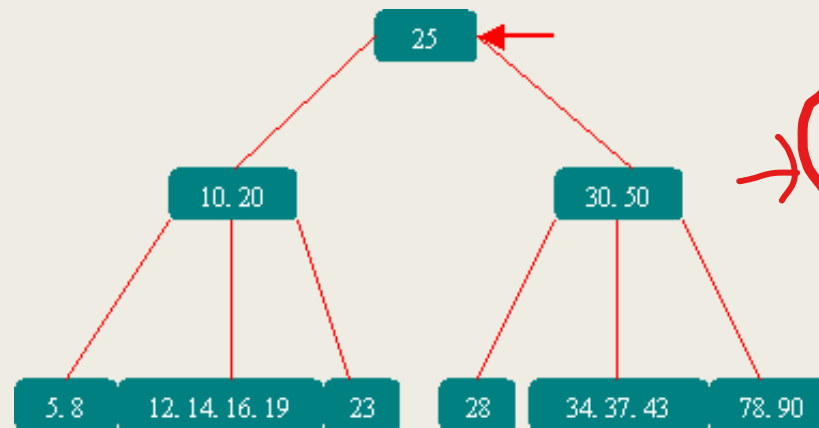
- Bài 1: tạo view_MatHang(MaHang, TenHang, MaCongTy, TenCongTyCungCap, MaLoaiHang, TenLoaiHang, SoLuong, DonViTinh, GiaHang) ✓
- Bài 2: Tạo view View_DonDatHang (SoHoaDon, MaKhachHang, TenCongTyKhachHang, HoVaTenNhanVien, NgayDatHang, NgayGiaoHang, NgayChuyenHang, NoiGiaoHang, MaHang, TenHang, SoLuong, GiaBan, MucGiamGia) ✓
- Thử xem có thể cập nhật, thêm, sửa, xóa dữ liệu qua các view đã tạo không.

2.3 Chỉ mục (index)

Khái niệm

→ select/where

- Là một trong những yếu tố quan trọng làm tăng tốc độ truy vấn dữ liệu
- Index cung cấp phương pháp truy xuất nhanh chóng tới các dòng trong bảng, tương tự như mục lục của cuốn sách
- Index được tạo ra trên các cột của table hay của view
- Cấu trúc của Index: Index được tạo thành từ các index node (index page) và chúng được tổ chức trong một cấu trúc có tên gọi là B-tree



→ Hơn chỗ
↑ delete/insert

Quá trình tìm kiếm theo index

- Khi index được tạo, giá trị của cột được đánh chỉ số sẽ được lưu trữ trong B-tree gồm các index page
- Bảng dữ liệu được lưu trữ trong các index page theo thứ tự của giá trị trong cột được chỉ số.
- Khi có yêu cầu truy vấn theo cột được index, thay vì quét toàn bộ bảng dữ liệu thì sẽ quét cây B-tree để tìm xem dòng dữ liệu nằm trên index page nào.

Quá trình tìm kiếm theo index (tiếp)

- Index được sử dụng để tìm ra giá trị duy nhất
- Index có thể được tạo ra bởi 1 hoặc nhiều trường. Đối với những trường được khai báo là **UNIQUE**, SQL Server tự động tạo index cho trường đó.
- Nên sử dụng **index trên cột** khi:
 - Cột được sử dụng thường xuyên cho việc tìm kiếm
 - Cột được dùng để sắp xếp dữ liệu
- Không nên sử dụng index khi:
 - Bảng chứa ít dòng, cột không sử dụng thường xuyên
 - Cột chỉ chứa vài giá trị khác nhau, có chứa các tập giá trị giống nhau

Các kiểu index

■ Clustered Index:

- Tiến hành lưu trữ mức vật lý của dữ liệu trong bảng rồi tiến hành sắp xếp chúng (thường áp dụng cho bảng có lượng bản ghi lớn).
- DL được lưu trữ theo cách sắp xếp trên khóa clustered và mỗi nút lá của clustered index chứa một bản ghi

■ Đặc điểm:

- Mỗi bảng chỉ có duy nhất một clustered index 1
- Khóa chính là 1 clustered index
- Clustered index chỉ được tạo ra trên cột hoặc tập cột có chứa giá trị/tập giá trị duy nhất.

Các kiểu index

■ Non-clustered Index

- Xác định thứ tự lưu trữ logic của dữ liệu trong bảng
- Non-clustered Index được định Nghĩa trên bảng trong đó DL có thể có cấu trúc phân cụm hoặc dạng vun đống (heap)
- Mỗi hàng index trong Non-clustered Index sẽ chứa một giá trị khóa Non-clustered Index và một bộ định vị hàng
- Non-clustered Index thường được áp dụng cho bảng chứa một lượng bản ghi nhỏ, Một bảng có thể có nhiều non-clustered index, lớn nhất là 249.
- DL được lưu trữ theo cách sắp xếp trên khóa non-clustered và mỗi nút lá của non-clustered index chứa một bản ghi
- Mặc định, lệnh `CREATE INDEX` tạo ra non_clustered index

Lệnh tạo index

CREATE [UNIQUE] [CLUSTERED] [NON CLUSTERED]



INDEX idex_name

ON table_name (column_name, column_name)



Lệnh tạo index

- Ví dụ:

```
CREATE CLUSTERED INDEX ind_NHANVIEN  
ON NHANVIEN(MaNhanVien)
```

- Hiển thị index trong bảng:

```
EXEC Sp_helpindex NHANVIEN
```

- Xóa index: DROP INDEX tên_bảng.tên_index

```
DROP INDEX NHANVIEN.ind_NHANVIEN
```

Unique index

- Unique index hay Unique nonclustered index dùng để tạo chỉ mục trên cột hay tập các cột chứa những giá trị hoặc tập giá trị duy nhất.
- Lưu ý: khi tạo Unique index trên cột hay tập cột thì không thể chèn (INSERT) vào bảng này bản ghi có giá trị hay tập giá trị giống với bất kì giá trị hay tập giá trị đã được chèn trước đó.

VD: CREATE UNIQUE NONCLUSTERED INDEX uninonclus_student ON Student(email);

Hoặc: CREATE UNIQUE INDEX uninonclus_student ON Student(email);

2.4 Con trỏ (Cursor)

Ko kinf

Khái niệm

- Các thao tác như SELECT, UPDATE, DELETE đều thao tác lên nhiều dòng dữ liệu thỏa mãn điều kiện WHERE mà không thể thao tác lên từng dòng dữ liệu cụ thể
- CURSOR là một đối tượng của CSDL được dùng để thao tác với từng dòng dữ liệu
- Đặc điểm: do phải lặp qua từng dòng dữ liệu nên đây là cách xử lý chậm nhất.
- Có 3 loại cursors: Transact- SQL Cursors, API Cursors và Client Cursors.

Duyệt trong tung bang

Các bước với con trỏ

n record

- Khai báo con trỏ: DECLARE
- Mở con trỏ: OPEN
- Duyệt dữ liệu trong con trỏ: FETCH
- Đóng con trỏ: CLOSE
- Giải phóng bộ nhớ: DEALLOCATE

*van con con tro, vi minh declare
khi nao can. open ra luon!*

Khai báo con trỏ

DECLARE Tên_cursor **CURSOR**

[LOCAL | GLOBAL] **phạm vi**

[FORWARD_ONLY | **SCROLL**] **duyet**

[STATIC| DYNAMIC| KEYSET] **kieu**

[READ_ONLY| SCROLL_LOCK] **vua doc vua ghi**

FOR Câu_lệnh_SELECT

[FOR UPDATE [OF Danh_sách_cột]]

**cho phép update voi
cot danh sach nao**

Khai báo con trỏ

- **Tên cursor** : tên của biến kiểu cursor.
LOCAL\GLOBAL: phạm vi hoạt động của biến cursor là cục bộ hay toàn cục
- **FORWARD_ONLY**: duyệt mẫu tin chỉ theo chiều từ trên xuống dưới.
- **SCROLL**: duyệt mẫu tin theo chiều tùy ý
- **STATIC**: dữ liệu trong con trỏ không thay đổi dù dữ liệu trong bảng nguồn thay đổi
- **DYNAMIC**: dữ liệu trong con trỏ thay đổi theo sự thay đổi của dữ liệu trong bảng nguồn

Khai báo con trỏ



KEYSET: giống DYNAMIC nhưng chỉ thay đổi những dòng bị cập nhật

READ_ONLY: chỉ đọc

■ **SCROLL_LOCK**: đọc/ghi

■ **SELECT**: không chứa các mệnh đề INTO, COMPUTE, COMPUTE BY

■ **Danh sách các cột cập nhật**: danh sách các cột sẽ thay đổi được.

Khai báo con trỏ

Ví dụ: Để định nghĩa một biến cursor chứa toàn bộ các dòng dữ liệu bên trong bảng NHANVIEN, duyệt được theo cả hai chiều:

```
declare cur_Nhanvien cursor
```

```
scroll
```

```
for
```

```
select * from Nhanvien
```

B2 Mở con trỏ

Cú pháp:

OPEN tên_con_trỏ

Ví dụ:

OPEN Cur_Nhanvien

B3 Duyệt các bản ghi trong con trỏ → rows

■ Cú pháp

FETCH [NEXT | PRIOR | FIRST | LAST | ABSOLUTE { n | $@nvar$ } |
RELATIVE { n | $@nvar$ }]

FROM [GLOBAL] *cursor_name* duyet where ?

[INTO $@variable_name$ [,... n]]

while
check: fetch status success = 0 ?
success -> end

■ Trong đó:

- NEXT: Chuyển sang mẫu tin kế tiếp
- PRIOR: Chuyển về mẫu tin trước đó

Duyệt các bản ghi trong con trỏ

➡ **FIRST** : duyệt bản ghi đầu tiên

➡ **LAST**: Chuyển đến bản ghi cuối cùng

➡ **ABSOLUTE {*n* | @*nvar*}** : duong/am: duyệt tu dong dau tien toi vi tri thu *n*

- Nếu *n* or @*nvar* > 0, tìm đến dòng thứ *n* tính từ dòng đầu tiên đếm xuống trong tập mẫu tin.
- Nếu *n* or @*nvar* < 0, tìm đến dòng thứ *n* tính từ dòng cuối cùng đếm lên.
- Nếu *n* or @*nvar* = 0, chuyển đến vùng BOF và không có giá trị trả về. no return
- Hằng số *n* phải là số nguyên và biến @*nvar* phải thuộc kiểu smallint, tinyint, hoặc int.
- Không sử dụng phương thức ABSOLUTE cho kiểu DYNAMIC.

Duyệt các bản ghi trong con trỏ

■ **RELATIVE {n | @nvar}** : *xd con trỏ đang ở vị trí nào ? duyệt từ vị trí đó*

- Nếu n hoặc $@nvar > 0$, chuyển xuống n dòng tính từ dòng kề dưới dòng hiện hành.
- Nếu n or $@nvar < 0$, Chuyển lên n dòng trước dòng hiện hành. ↑
- Nếu n or $@nvar = 0$, trả về dòng hiện hành. *con trỏ hiện hành -> bản ghi thu bao nhiêu ?*

■ **cursor_name**: Tên cursor đang mở.

- Nếu tồn tại cursor cục bộ và cursor toàn cục có cùng tên thì tên cursor được sử dụng sẽ là cursor cục bộ nếu không có từ khóa GLOBAL.

■ **INTO @varname[,...n]** : Danh sách biến cục bộ nhận dữ liệu


- Số biến phải bằng số cột đã liệt kê trong câu lệnh Select khi tạo Cursor.
- Kiểu dữ liệu của mỗi biến phải tương thích với kiểu dữ liệu của cột hoặc được hỗ trợ chuyển kiểu ngầm định theo kiểu của cột.

Đóng và giải phóng con trỏ

- Đóng con trỏ:

CLOSE **tên_con_trỏ**

- Giải phóng con trỏ:



DEALLOCATE **tên_con_trỏ**

Ví dụ

	MaNV	HoTen	Phong	Bacluong	NgaySinh	NguoiquanLy
1	KS001	nguyen toan	NS	NULL	1978-03-23	NULL
2	KS002	Nguyễn Quỳnh Châu	KHCN	NULL	1987-03-23	Ks001
3	KS003	Nguyễn Tuấn Dũng	KHCN	NULL	1982-12-20	KS001
4	KS004	Hoàng Thanh Thu	NS	NULL	1986-06-23	NULL
5	KS005	Hoàng Thanh Minh	KT	NULL	1986-06-23	NULL
6	KS006	Hoàng Văn Thắng	KT	NULL	1986-06-23	KS005
7	KS007	Hoàng Văn Thái	TC	NULL	1987-06-23	NULL
8	KS008	Thái Minh Quân	TC	NULL	1982-06-27	KS007
9	KS010	Nguyen Tuan	KHCN	NULL	1982-12-21	KS001
10	KS011	Nguyen Tuan	KHCN	NULL	1982-12-21	KS001
11	KS013	Nguyen Tuan	KHCN	NULL	1982-12-21	KS001
12	KS014	Nguyen Tuan	KHCN	NULL	1982-12-21	KS001
13	KS12	Nguyen Tuan	KHCN	NULL	1982-12-21	KS001

Ví dụ

```
SQLQuery1.sql - L:\NOVO-PC\ORACLE (32)
declare cur_Nhanvien cursor
scroll
for
select * from Nhanvien
open cur_Nhanvien
fetch first from cur_Nhanvien
```

100 %

Results Messages

	MaNV	Ho Ten	Phong	Bacluong	Ngay Sinh	NguoiQuanLy
1	KS001	nguyen toan	NS	NULL	1978-03-23	NULL

- Câu lệnh

FETCH NEXT FROM Cur_Nhanvien



trở đến bản ghi tiếp theo trong con trỏ:

	MaNV	HoTen	Phong	Bacluong	NgaySinh	NguoiquanLy
1	KS002	Nguyễn Quỳnh Châu	KHCN	NULL	1987-03-23	Ks001

■ Câu lệnh

FETCH ABSOLUTE 3 FROM Cur_Nhanvien

trở đến bản ghi thứ 3 trong con trỏ:

 Results  Messages						
	MaNV	HoTen	Phong	Bacluong	NgaySinh	NguoiQuanLy
1	KS003	Nguyễn Tuấn Dũng	KHCN	NULL	1982-12-20	KS001



- Câu lệnh

FETCH ABSOLUTE -6 FROM Cur_Nhanvien


- Câu lệnh

FETCH RELATIVE 3 FROM CUR_NHANVIEN

trở đến bản ghi thứ 3 nằm phía sau với bản ghi hiện tại trong con trỏ:

<div><div> Results</div><div> Messages</div></div>						
	MaNV	HoTen	Phong	Bacluong	NgaySinh	NguaiQuanLy
1	KS006	Hoàng Văn Thắng	KT	NULL	1986-06-23	KS005

- Ví dụ: Sử dụng con trỏ kết hợp với các biến khác
- Khai báo con trỏ như sau:



```
declare cur_Nhanvien cursor  
scroll  
for  
select MaNV, Hoten from Nhanvien
```

- Sử dụng hai biến @a và @b để chứa dữ liệu trong con trỏ:

open cur_Nhanvien

declare @a char(10), @b nchar(40)

while @@fetch_status=0 ✓

begin

{ fetch next from cur_Nhanvien into @a, @b }

{ print N'Mã nhân viên:' + @a + N' Họ tên nhân viên:' + @b }

end