



**PHÂN HIỆU TRƯỜNG ĐẠI HỌC THỦY LỢI**  
**THUYLOI UNIVERSITY - SOUTHERN CAMPUS**

## **Chương 2: CÁC ĐỐI TƯỢNG TRONG SQL SERVER**

Email: [vienthanhnha@tlu.edu.vn](mailto:vienthanhnha@tlu.edu.vn)

Ths. Viên Thanh Nhã

# Cơ sở dữ liệu - Database

## 2.1 Cơ sở dữ liệu – Database

### 2.1.1 Các Database hệ thống

⊕ master

⊕ model

⊕ msdb

⊕ tempdb

Khi cài đặt SQL Server có 4 Database hệ thống được cài đặt, đó là:

- master: Ghi nhận thông tin cấp hệ thống, thông tin khởi tạo SQL Server và thiết lập cấu hình SQL Server. Database này cũng ghi nhận tất cả các tài khoản đăng nhập, sự tồn tại của các Database khác, vị trí tập tin chính cho tất cả Database người dùng.
- tempdb: Giữ các bảng tạm, các stored procedure tạm,.v.v... Được dùng cho các nhu cầu lưu trữ tạm của SQL Server.
- model: là khuôn mẫu cho tất cả các CSDL khác được tạo trên hệ thống kể cả tempdb. Database model phải được tồn tại trên hệ thống, bởi vì nó được dùng để tạo lại tempdb mỗi khi SQL server được khởi động.
- msdb: Giữ các bảng mà SQL Server Agent dùng để lập thời gian biểu thực thi các công việc, các cảnh báo và các operator.

# Cơ sở dữ liệu - Database

## 2.1.2 Tạo Database

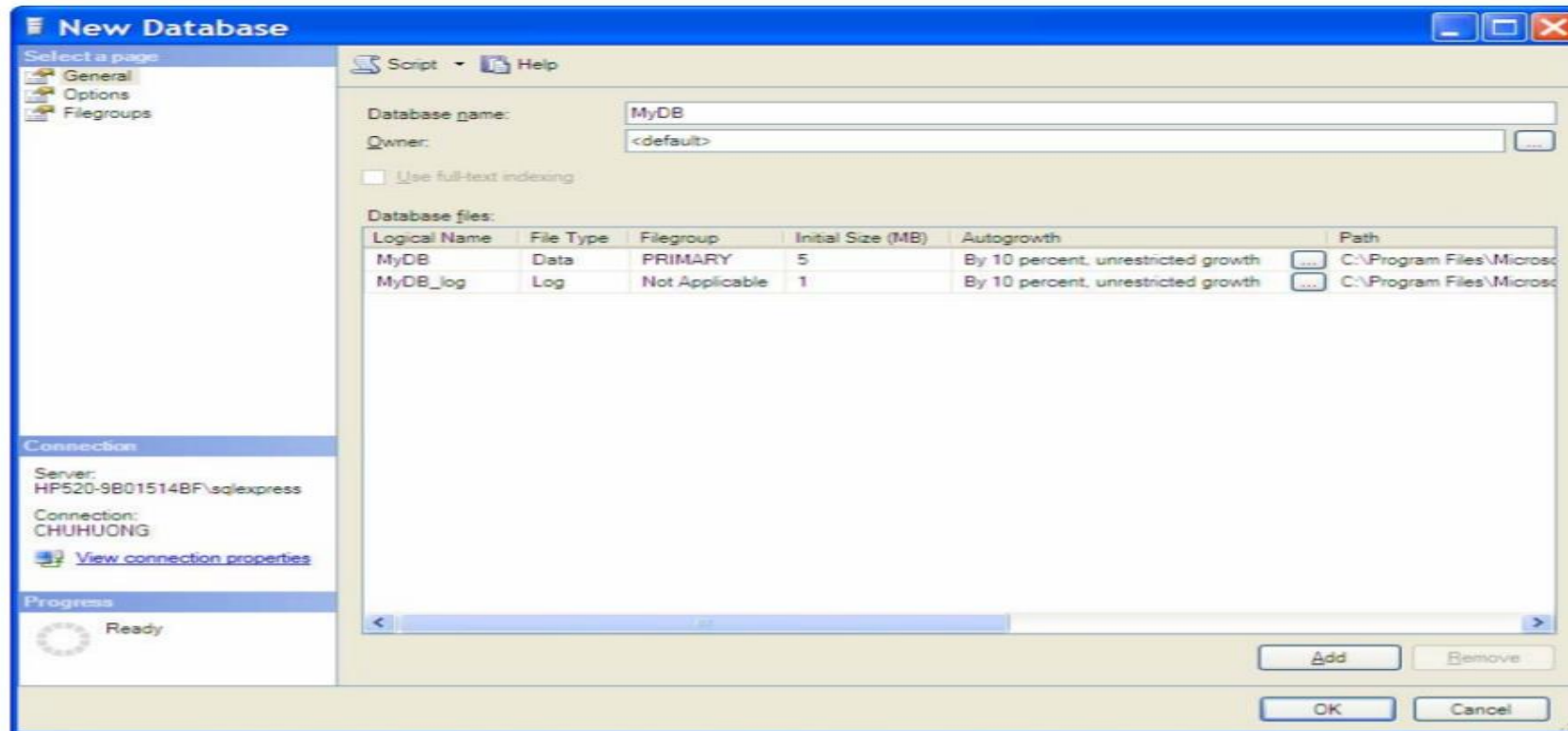
Trong SQL Server 2012 , ta có 2 cách khác nhau để tạo:

- + SQL Server Management Studio
- + Dùng T-SQL

# Cơ sở dữ liệu - Database

\* Sử dụng SQL Server Management Studio của SQL Server 2012: Như đã giới thiệu ở trên, SQL Server Management Studio là công cụ tích hợp SQL Server Enterprise Manager của phiên bản 2000 nên ta có thể sử dụng để tạo database tương tự như phiên bản 2000. Cách tạo như sau:

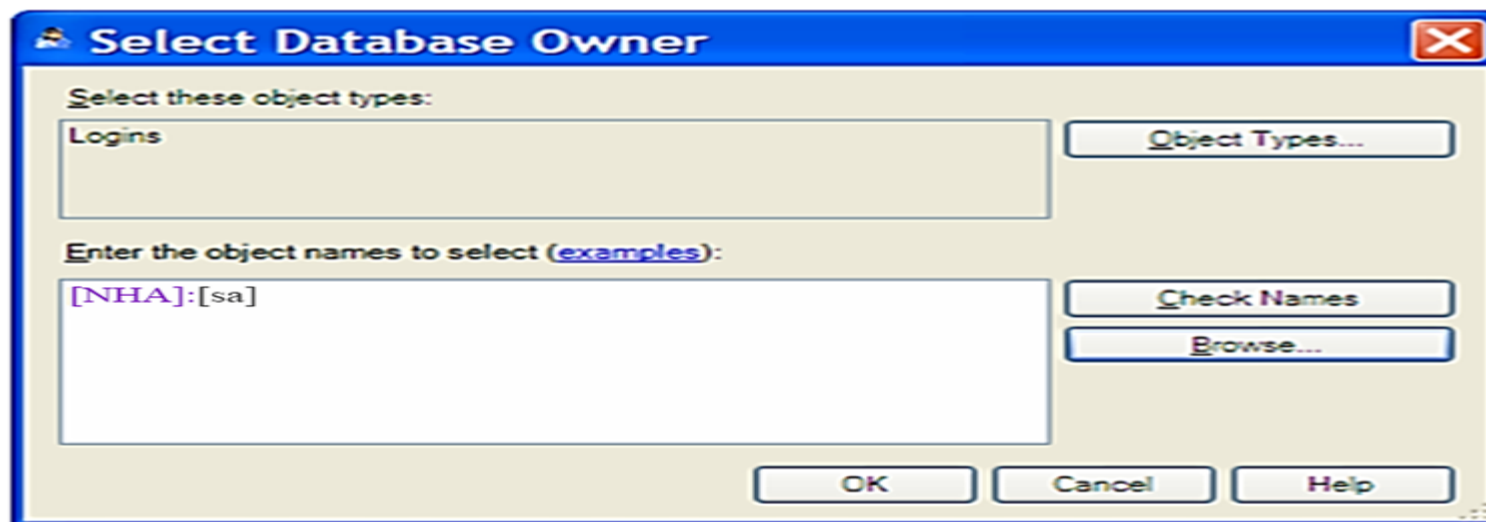
1. Mở rộng các đối tượng trên thể hiện của SQL trong cửa sổ Object Explorer muốn tạo Database.
2. Right click lên mục Database\ chọn New Database xuất hiện cửa sổ New Database sau:



# Cơ sở dữ liệu - Database

Trong cửa sổ New Database, tại tab General ta thực hiện điền các thông số cho các mục như sau:

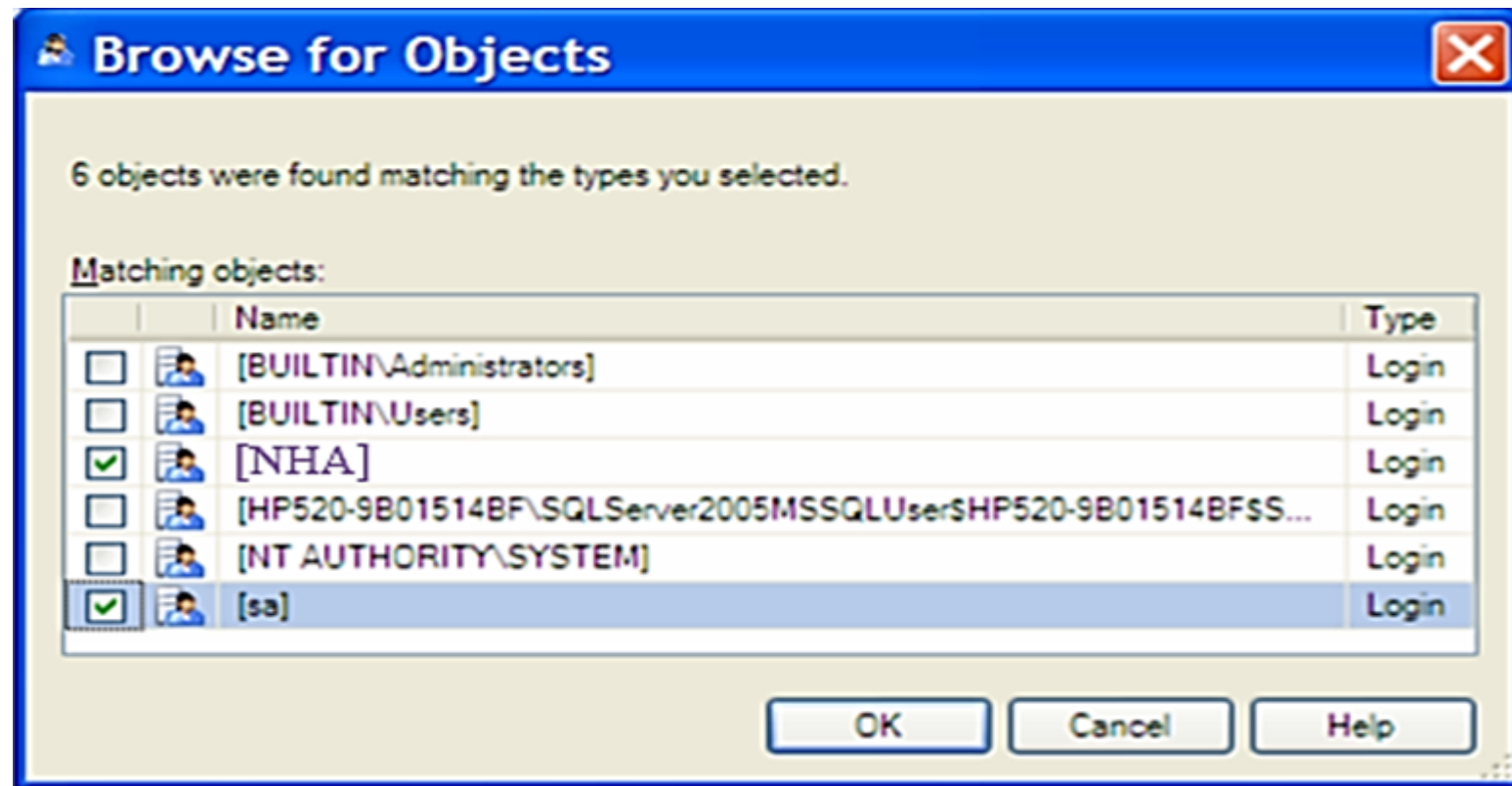
- + Database name: Điền tên Database muốn tạo
- + Owner: Chỉ định tên các Logins sở hữu Database đang tạo. Để chọn các logins ta click vào nút ... xuất hiện cửa sổ 'Select Database Owner'



Cửa sổ Select Database Owner.

# Cơ sở dữ liệu - Database

Trong mục 'Enter the Object names to select' ta nhập tên các logins hoặc chọn nút Browse để liệt kê và chọn các logins trong cửa sổ 'Browse for Objects'



Cửa sổ Browse for Objects.

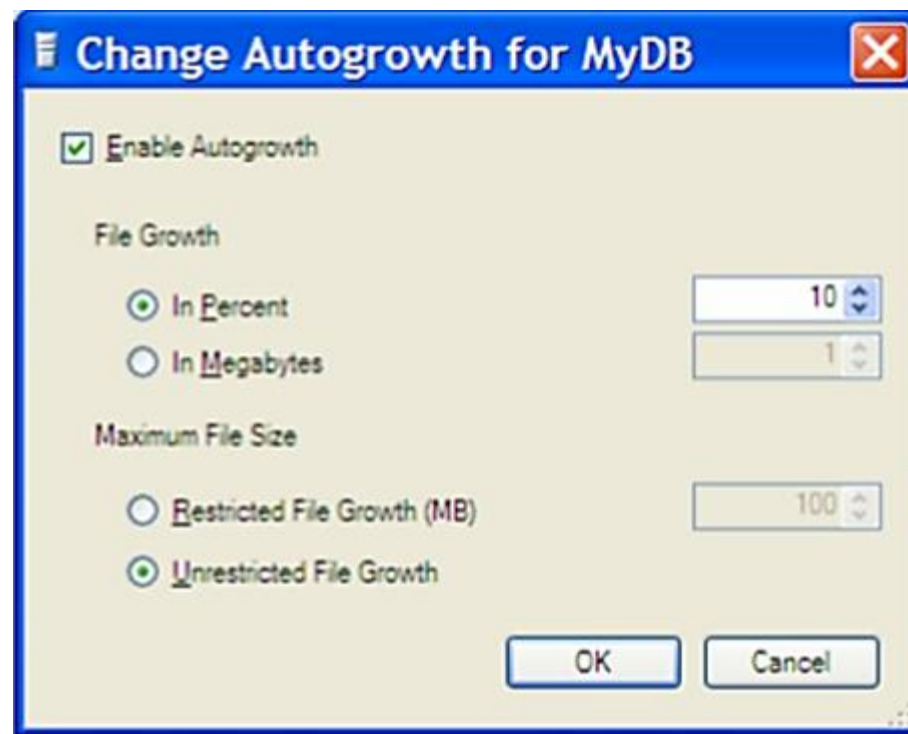
# Cơ sở dữ liệu - Database

+ Database files: Thực hiện chọn các thuộc tính cho các files database, bao gồm:

- ☐ Logical name: Tên logic của file database.
- ☐ File Type: Kiểu file (Data, Log)
- ☐ Filegroup: Chỉ định nhóm file.
- ☐ Initial Size (MB): Kích thước khởi tạo dung lượng các file, tính theo MB.
- ☐ Autogrowth: Chỉ định các tham số tự động gia tăng dung lượng cho các file. Click vào nút ... xuất hiện cửa sổ 'Change Autogrowth for MyDB' cho phép khai báo các tham số Autogrowth:
  - Tùy chọn Enable Autogrowth cho phép tự động gia tăng;

# Cơ sở dữ liệu - Database

- File Growth chỉ định độ tự động gia tăng theo phần trăm (In Percent) hoặc theo dung lượng chỉ định (In Megabytes);
- Maximum File Size định nghĩa độ gia tăng của file đến dung lượng lớn nhất.



Cửa sổ Change Autogrowth for MyDB



# Cơ sở dữ liệu - Database

❑ Path: Chỉ định đường dẫn vật lý lưu trữ các files database. Click nút ... để thay đổi đường dẫn mặc định.

**\*Sử dụng T-SQL Ta có thể sử dụng T-SQL hay script để tạo Database:**

Với cách này ta cần hiểu rõ về các cú pháp câu lệnh trong SQL Server để tạo Database.

```
CREATE DATABASE database_name
    [ ON
        [ PRIMARY ] [ <filespec> [ ,...n ]
        [ , <filegroup> [ ,...n ] ]
    [ LOG ON { <filespec> [ ,...n ] } ]
    ]
[;]
```

# Cơ sở dữ liệu - Database

***Để attach một database:***

```
CREATE DATABASE database_name
    ON <filespec> [ ,...n ]
    FOR { ATTACH | ATTACH_REBUILD_LOG }
[;]
```

**<filespec> ::=**

```
{
(
    NAME = logical_file_name ,
    FILENAME = 'os_file_name'
        [ , SIZE = size [ KB | MB | GB | TB ] ]
        [ , MAXSIZE = { max_size [ KB | MB | GB | TB ]
| UNLIMITED } ]
        [ , FILEGROWTH = growth_increment [KB | MB |
GB | TB | % ] ]
) [ ,...n ]
}
```

# Cơ sở dữ liệu - Database

```
<filegroup> ::=  
{  
  FILEGROUP filegroup_name [ DEFAULT ]  
    <filespec> [ ,...n ]  
}
```

Các tham số trong đó: *database\_name*: Là tên của CSDL. Nếu tên của file dữ liệu (data file) không được chỉ định thì SQL Server sử dụng *database\_name* là tên cho *logical\_file\_name* và *os\_file\_name*.

*ON*

Chỉ định định nghĩa các file trên đĩa được sử dụng để lưu trữ các phần dữ liệu của database, data files.

*PRIMARY*

Chỉ định này liên quan đến danh sách định nghĩa primary file <filespec>. File đầu tiên được chỉ định trong <filespec> của nhóm filegroup primary filegroup trở thành file primary.

# Cơ sở dữ liệu - Database

Một database chỉ có thể duy nhất một file primary.

Nếu từ khóa PRIMARY không được chỉ định thì file đầu tiên trong danh sách các file của câu lệnh CREATE DATABASE sẽ trở thành file primary.

## LOG ON

Chỉ định định nghĩa file log lưu trữ trên đĩa, log files.

## FOR ATTACH

Chỉ định database được tạo bằng việc attach tập các file hệ thống đã tồn tại.

FOR ATTACH có các yêu cầu sau:

- Các files data (MDF và NDF) phải đã tồn tại.
- Nếu nhiều files log tồn tại, thì tất cả phải sẵn có.

# Cơ sở dữ liệu - Database

## *FOR ATTACH\_REBUILD\_LOG*

Chỉ định database được tạo bằng việc attach tập các file hệ thống đã tồn tại. Nếu một hoặc nhiều files log giao dịch bị lỗi thì file log sẽ được xây dựng lại.

*<filespec>*

Điều khiển các thuộc tính của file.

- NAME logical\_file\_name: Chỉ định tên logical cho file. NAME được yêu cầu khi FILENAME được chỉ định.
- FILENAME os\_file\_name: Chỉ định tên, đường dẫn file hệ điều hành (file vật lý).
- SIZE size: Chỉ định kích thước file.

# Cơ sở dữ liệu - Database

- MAXSIZE max\_size: Chỉ định kích thước lớn nhất mà file có thể phát triển đến. Từ khóa UNLIMITED chỉ định file được phát triển cho đến khi đĩa bị đầy.
- FILEGROWTH growth\_increment: Chỉ định độ t ự đồng gia tăng của file.

*<filegroup>*

Điều khiển các thuộc tính của filegroup.

- FILEGROUP filegroup\_name: Chỉ định tên logical của filegroup.
- DEFAULT: Chỉ định tên filegroup là filegroup mặc định trên database.

# Cơ sở dữ liệu - Database

## Ví dụ:

Đưa ra một cách tạo Database MyDB với tập tin dữ liệu chính là MyDB\_Data.mdf, dung lượng khởi tạo là 1MB và tối đa là 10MB và độ gia tăng kích thước là 10%. Tập tin bản ghi giao dịch là MyDB\_Log.ldf với dung lượng ban đầu là 2MB và kích thước tối đa không giới hạn, độ gia tăng dung lượng là 10MB.

Để tạo script (dùng T- SQL) ta mở cửa sổ query để soạn thảo:

- Đối với SQL Server 2012: Trong cửa sổ SQL Server Management Studio, trên thanh Standard chọn nút New Query để tạo cửa sổ truy vấn kết nối đến thể hiện SQL đang được kết nối hoặc nút Database Engine Query để xuất hiện cửa sổ kết nối ta thực hiện kết nối đến thể hiện SQL mà muốn thực hiện trên đó.

# Cơ sở dữ liệu - Database

Xuất hiện cửa sổ truy vấn ta thực hiện nhập đoạn mã lệnh sau:

```
Use master
go
create database MyDBT
On
(
Name= MyDBT_Data,

FileName='E:\Temp\MyDBT_Data.mdf',
Size=10MB,
MaxSize=100MB,
FileGrowth=10%
)
Log On
(
Name= MyDBT_log,
FileName='E:\Temp\MyDBT_Log.ldf',
Size=2MB,
MaxSize=UNLIMITED,
FileGrowth=10%
)
```



# Cơ sở dữ liệu - Database

- Click nút Execute hoặc F5 để chạy.

## c) Xóa Database

### ❖ Sử dụng SQL Server Management Studio của SQL Server 2012:

Để xóa một database ta đăng nhập vào SQL Server Management Studio với một login có quyền xóa database đó và thực hiện các bước sau:

1. Mở rộng các đối tượng trên thể hiện của SQL trong cửa sổ Object Explorer muốn xóa Database.
2. Mở rộng mục Database và Right click lên database muốn xóa và chọn Delete xuất hiện cửa sổ 'Delete Object' chọn OK để xóa.

# Cơ sở dữ liệu - Database

## ❖ Dùng T-SQL

Để xóa Database ta sử dụng cú pháp sau:

```
DROP DATABASE database_name [ ,...n ]
```

Ví dụ: nhập đoạn T-SQL sau để xóa Database MyDB.

```
Use master
```

*Go*

```
Drop Database MyDB
```

*Go*

# View

## 2.2 View

### 2.2.1 Khái niệm View

View là một bảng ảo được định nghĩa bởi một truy vấn với phát biểu SELECT. View được hình thành dữ liệu từ một hoặc nhiều bảng thật. Đối với người sử dụng thì view giống như một bảng thật.

### 2.2.2 Tạo view

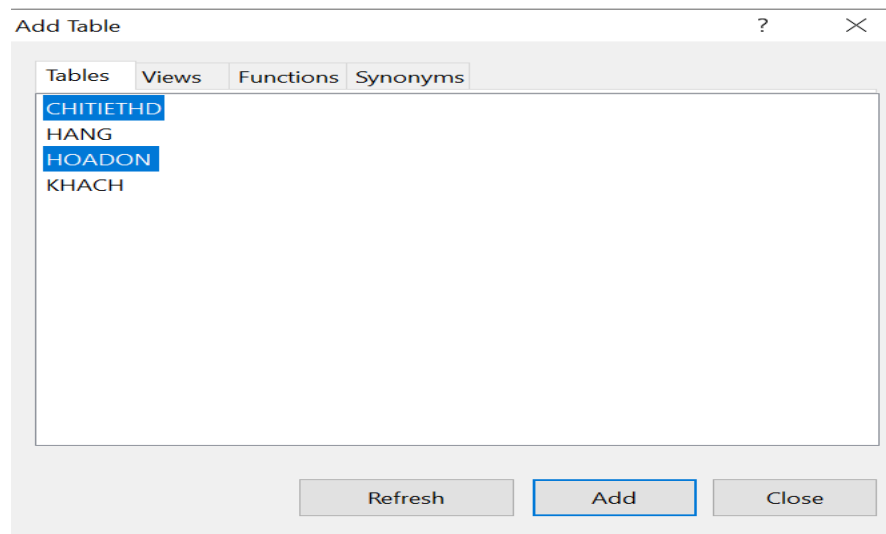
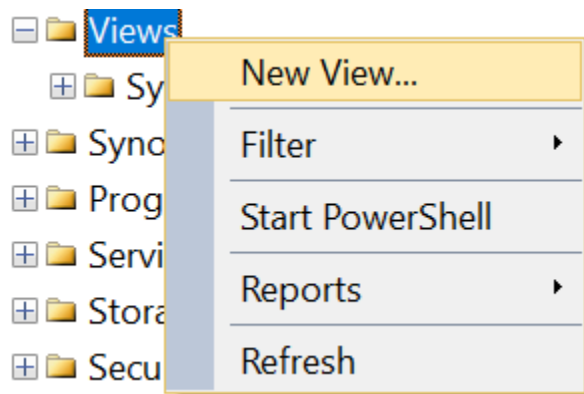
Trong SQL Server 2012, ta có 2 cách khác nhau để tạo:

- + SQL Server Management Studio
- + Dùng T-SQL

# View

## ❖ SQL server Management Studio

- SQL server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu muốn tạo view, chẳng hạn CSDL QLHH và chọn mục Views.
- Right Click lên danh mục Views, chọn New View. Xuất hiện cửa sổ thiết kế view đối với SQL Server 2012, gồm 4 vùng sau:



# View

DESKTOP-SJ5FR88\SERVERNHA.QLHH - dbo.View\_2\* - Microsoft SQL Server Management Studio

File Edit View Project Debug Query Designer Tools Window Help

Object Explorer

- FileTables
  - dbo.CHITIETHD
  - dbo.HANG
  - dbo.HOADON
  - dbo.KHACH
- Views
- Synonyms
- Programmability
- Service Broker
- Storage
- Security
- ReportServer\$SERVERNHA
- ReportServer\$SERVERNHATempDB
- QLHH
  - Database Diagrams
  - Tables
  - System Tables
  - FileTables
  - Views
  - System Views
  - Synonyms
  - Programmability
  - Service Broker
  - Storage
  - Security
- Security
- Server Objects
- Replication

DESKTOP-SJ5FR88\...LHH - dbo.View\_2\* x DESKTOP-SJ5FR88\...LHH - dbo.View\_1 SQLQuery2.sql - D...NHA.QLHH (sa (56))

Diagram pane

Grid pane

SQL pane

Results pane

Properties

[View] dbo.View\_2

(Identity)

(Name)	View_2
Database Name	QLHH
Description	
Schema	dbo
Server Name	desktop-sj5fr88\servernh

View Designer

Bind To Schema	No
Deterministic	Yes
Distinct Values	No
GROUP BY Extensio	<None>
Output All Columr	No
SQL Comment	
Top Specification	No
Update Specificati	No

Column Alias Table Outp... Sort Type Sort Order Filter Or... Or...

SoHD		CHITIETHD	☑					
MaH		CHITIETHD	☑					
SLBan		CHITIETHD	☑					
NgayHD		HOADON	☑					

SELECT dbo.CHITIETHD.SoHD, dbo.CHITIETHD.MaH, dbo.CHITIETHD.SLBan, dbo.HOADON.NgayHD  
FROM dbo.CHITIETHD INNER JOIN  
dbo.HOADON ON dbo.CHITIETHD.SoHD = dbo.HOADON.SoHD

SoHD	MaH	SLBan	NgayHD
00001	key	3	2020-03-...
00001	mo	2	2020-03-...
00001	web	1	2020-03-...

1 of 6 Cell is Read Only.

# View

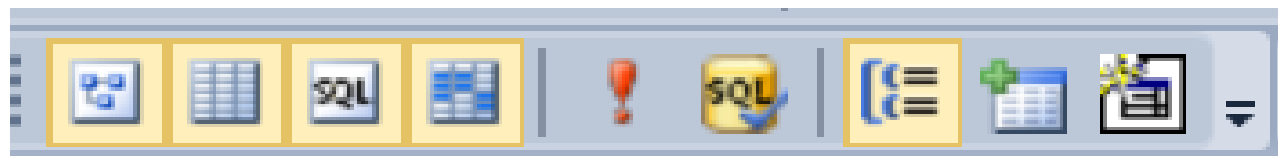
- + Diagram pane: Hiển thị dữ liệu nguồn có thể là các bảng, các view khác, functions để tạo view. Các cột dữ liệu của view được chọn từ vùng này.
- + Grid pane (Criteria pane): Hiển thị các cột của view đã được chọn từ vùng diagram.
- + SQL pane: Hiển thị phát biểu SQL dùng để định nghĩa view.
- + Results pane: Hiển thị kết quả nhận được từ view.

Ta có thể ẩn hoặc hiện các cửa sổ này bằng cách click vào các nút tương ứng trên thanh công cụ của cửa sổ thiết kế view.

# View

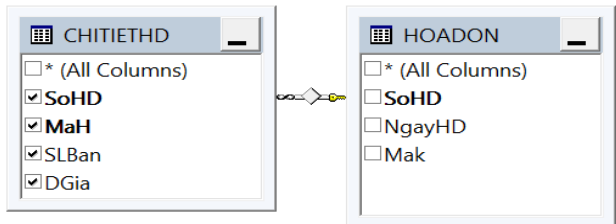
## ➤ Đối với SQL Server 2012:

- + Show Diagram pane: Ẩn hoặc hiện cửa sổ Diagram pane.
- + Show Criteria pane: Ẩn hoặc hiện cửa sổ Criteria pane.
- + Show SQL pane: Ẩn hoặc hiện cửa sổ SQL pane.
- + Show Results pane: Ẩn hoặc hiện cửa sổ kết quả.
- + Execute SQL: Thực hiện truy vấn.
- + Verify SQL Syntax: Kiểm tra cú pháp câu lệnh SQL.
- + Add Group By: Thêm mệnh đề group by trong câu lệnh SQL.
- + Add Table: Thêm các bảng, view, hàm làm nguồn dữ liệu cho view mới.



# View

- Click vào nút Add Table, xuất hiện hộp thoại Add Table chọn các bảng các view làm nguồn dữ liệu cho view mới. Trong ví dụ ta chọn bảng HOADON và CHITIETHOADON.
- Sau khi chọn nguồn dữ liệu ta thực hiện chọn các trường làm dữ liệu trên view như hình (SQL Server 2012)
- Click vào nút Save, xuất hiện hộp thoại Save lưu với tên Danhsach.



Column	Alias	Table	Output	Sort Type	Sort Order	Filter	Or...	Or...
SoHD		CHITIETHD	<input checked="" type="checkbox"/>					
MaH		CHITIETHD	<input checked="" type="checkbox"/>					
SLBan		CHITIETHD	<input checked="" type="checkbox"/>					
DGia		CHITIETHD	<input checked="" type="checkbox"/>					
			<input type="checkbox"/>					
			<input type="checkbox"/>					

```
SELECT dbo.CHITIETHD.SoHD, dbo.CHITIETHD.MaH, dbo.CHITIETHD.SLBan, dbo.CHITIETHD.DGia
FROM   dbo.CHITIETHD INNER JOIN
       dbo.HOADON ON dbo.CHITIETHD.SoHD = dbo.HOADON.SoHD
```

	SoHD	MaH	SLBan	DGia
▶	00001	key	3	100000
	00001	mo	2	100000
	00001	web	1	100000





# View

Sử dụng vùng Grid Pane (Criteria) để hỗ trợ cho việc thiết kế View:

- + Cột Column: Chứa tên các trường trong bảng/view làm nguồn dữ liệu cho truy vấn này hoặc chứa một biểu thức định nghĩa một trường mới cho view.
- + Cột Alias: Chỉ định bí danh cho trường trong view mới.
- + Cột Table: Chỉ định nguồn dữ liệu.
- + Cột Output: Chỉ định hiển thị hay không hiển thị trường đó.
- + Cột Sort Type và Sort Order: Dùng để sắp xếp dữ liệu.
- + Cột Criteria: Dùng để đặt điều kiện lọc cho các bản ghi.
- + Các cột Or: Dùng kết hợp với cột Criteria để tạo các điều kiện lọc dữ liệu phức tạp.

# View

Tính tổng trong truy vấn: Để tính tổng trong truy vấn dùng vùng Grid Pane ta có thể tiến hành phân nhóm các bản ghi và thực hiện tính toán trên từng phân nhóm đó. Để tính tổng trong truy vấn ta chọn nút Use Group By xuất hiện cột Group By trong vùng Grid Pane. Ta thực hiện tiến hành phân các nhóm trường như sau:

- + Trường làm điều kiện, tiêu chuẩn tham gia phân nhóm và tính tổng: Chọn Where trong cột Group By và đặt biểu thức điều kiện trong cột Criteria.
- + Trường phân nhóm: Chọn Group by trong cột Group By.
- + Trường tính toán: Chọn một hàm có sẵn (sum, count, avg, max, min, .v.v...) trong cột Group By hoặc xây dựng một biểu thức tính toán trong cột Column.
- + Định tiêu chuẩn hiển thị kết quả: Đặt điều kiện ở cột Criteria tại các trường phân nhóm và trường tính toán.
- + Chọn thứ tự hiển thị: Dùng cột Sort Type và Sort Order tại các trường phân nhóm và các trường tính toán.

# View

## ❖T-SQL:

```
CREATE VIEW [schema_name.]view_name[(column[,...n])]ew thuộc lược đồ đó.  
[ WITH <view_attribute> [ ,...n ] ]  
AS select_statement [ ; ]  
[ WITH CHECK OPTION ]
```

```
<view_attribute> ::=  
{  
    [ ENCRYPTION ]  
    [ SCHEMABINDING ]  
    [ VIEW_METADATA ] }
```

Trong đó:

schema\_name

Là tên của lược đồ mà view thuộc lược đồ đó.

view\_name

Là tên của view.

# View

## *column*

Là tên được sử dụng cho một cột trong view. Tên một cột chỉ yêu cầu khi cột đó được sinh ra từ một biểu thức đại số, một hàm, một hằng; khi hai hoặc nhiều cột khác nhau có cùng tên, điển hình là trong liên kết; hoặc khi một cột trong view được chỉ định tên khác với tên từ nguồn dữ liệu. Các tên cột có thể được gán trong câu lệnh SELECT.

Nếu cột không được chỉ định trong view thì các cột trong view có tên cùng tên với các cột câu lệnh SELECT.

## *select\_statement*

Là các khối câu lệnh SELECT định nghĩa view. Các khối câu lệnh SELECT được phân cách nhau bởi mệnh đề UNION hoặc UNION ALL.

# View

Một Index được định nghĩa trên view đòi hỏi view phải được xây dựng từ một bảng đơn hoặc nhiều bảng liên kết nhau trong tổ hợp tùy chọn.

## *CHECK OPTION*

Bắt buộc tất các câu lệnh sửa đổi dữ liệu thực hiện trên view phải tuân theo các điều kiện trong khối câu lệnh `select_statement`.

## *ENCRYPTION*

Mã hóa bản text của câu lệnh `CREATE VIEW` trong `sys.syscomments`. Việc sử dụng `WITH ENCRYPTION` ngăn cản view bị công bố như là một phần bản sao SQL Server.

## *SCHEMABINDING*

Buộc view vào một lược đồ dưới một bảng hoặc nhiều bảng. Khi `SCHEMABINDING` được chỉ định, bảng hoặc các bảng cơ sở không được sửa trong những cách ảnh hưởng đến định nghĩa view.

# View

## *VIEW\_METADATA*

Chỉ định các thể hiện SQL Server sẽ trả về cho DB-Library, ODBC, và OLE DB APIs thông tin siêu dữ liệu về view, thay cho bảng và các bảng cơ sở khi trình duyệt siêu dữ liệu được yêu cầu cho một truy vấn tham chiếu đến view.

Ví dụ: Tạo view danh sách khách hàng trong bảng khách hàng

```
CREATE VIEW CUSTOMERS_VIEW AS  
SELECT mak, tenk, diachi  
FROM khach;
```

### Ví dụ:

Trong CSDL **QLHH** có hai bảng Orders và Order Details. Ta xây dựng view tính tổng giá trị cho từng hóa đơn.

```
CREATE VIEW TONGGIATRI_VIEW AS  
SELECT hd.sohd, sum(ct.Dgia*ct.SLBan) as tong  
FROM hoadon hd, chitiethd ct where hd.sohd=ct.sohd  
group by hd.sohd
```

# View

## c) Thay đổi view

### ❖ Dùng SQL Server Management Studio

- Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu muốn tạo view, chẳng hạn CSDL QLDiemSV và chọn mục Views.
- Right Click lên View, chọn Modify. Xuất hiện cửa sổ thiết kế view.
- Thực hiện các thay đổi trên view và ghi lại các thay đổi đó.

# View

❖ Dùng T-SQL: Ta dùng cú pháp câu lệnh sau

```
ALTER VIEW [schema_name.]view_name
[(column[,...n])]
[ WITH <view_attribute> [ ,...n ] ]
AS select_statement [ ; ]
[ WITH CHECK OPTION ]

<view_attribute> ::=
{ [ENCRYPTION] [SCHEMABINDING] [VIEW_METADATA]
}
```

**Ví dụ:** Sửa đổi view DSSV như sau:

```
ALTER VIEW DSSV
AS
    Select  MaSV, Hodem + ' ' + TenSV AS Hoten, Ngaysinh
    From  HosoSv
GO
```



# View

## d) Xóa view

- Trong cửa sổ SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu muốn xóa view, chẳng hạn CSDL QLDiemSV và chọn mục Views. Sau đó right click lên View muốn xóa, chọn Delete xuất hiện cửa sổ xác nhận xóa chọn OK.
- Dùng T-SQL dùng lệnh DROP VIEW theo cú pháp:

`DROP VIEW view_name`

**Ví dụ:** Xóa view DSSV như sau:

`DROP VIEW DSSV`

# Chỉ mục - Index

## 2.3 Chỉ mục – Index

### 2.3.1 Khái niệm

Chỉ mục là một trong những công cụ mạnh có sẵn đối với người thiết kế CSDL. Một chỉ mục là một cấu trúc phụ cho phép cải thiện hiệu suất thực thi các truy vấn bằng cách giảm thiểu các hoạt động nhập/xuất dữ liệu cần thiết để được dữ liệu yêu cầu. Tùy thuộc vào kiểu của nó, mà chỉ mục được lưu với dữ liệu hoặc tách biệt với dữ liệu.

- Chỉ mục khóa: Chỉ mục khóa chỉ rõ cột hoặc các cột được dùng để sinh ra chỉ mục. Nó cho phép tìm nhanh chóng dòng dữ liệu muốn tìm. Để truy cập dòng dữ liệu dùng chỉ mục, ta chỉ cần đưa ra giá trị khóa của chỉ mục hoặc đưa các giá trị vào mệnh đề WHERE trong khối câu lệnh SELECT.
- Chỉ mục duy nhất: Là chỉ mục chỉ chứa một dòng dữ liệu cho mỗi khóa chỉ mục. Một chỉ mục là duy nhất nếu bản thân dữ liệu là duy nhất, nếu không duy nhất ta có thể tạo chỉ mục kết hợp trên nhiều cột để đạt được chỉ mục duy nhất.

# Chỉ mục - Index

- Các kiểu chỉ mục: Các chỉ mục được lưu trữ dữ liệu dưới dạng cây nhị phân B-Tree. Có hai kiểu chỉ mục
  - Chỉ mục liên cung (clustered): Là chỉ mục lưu trữ các dòng dữ liệu thực sự của bảng trong nút lá, theo thứ tự đã được sắp xếp.
  - Chỉ mục phi liên cung (Nonclustered): Không chứa dữ liệu trong nút lá, mà nó chứa thông tin về vị trí của dòng dữ liệu: nếu không có chỉ mục liên cung trên bảng thì nó chứa số nhận dạng dòng (Row ID); nếu có chỉ mục liên cung thì trong nút lá này sẽ chứa giá trị khóa chỉ mục liên cung cho dữ liệu đó.

**Chú ý:** Indexes được tạo tự động khi các ràng buộc PRIMARY KEY và UNIQUE được định nghĩa trên các cột của bảng

# Chỉ mục - Index

## 2.3.2 Tạo chỉ mục

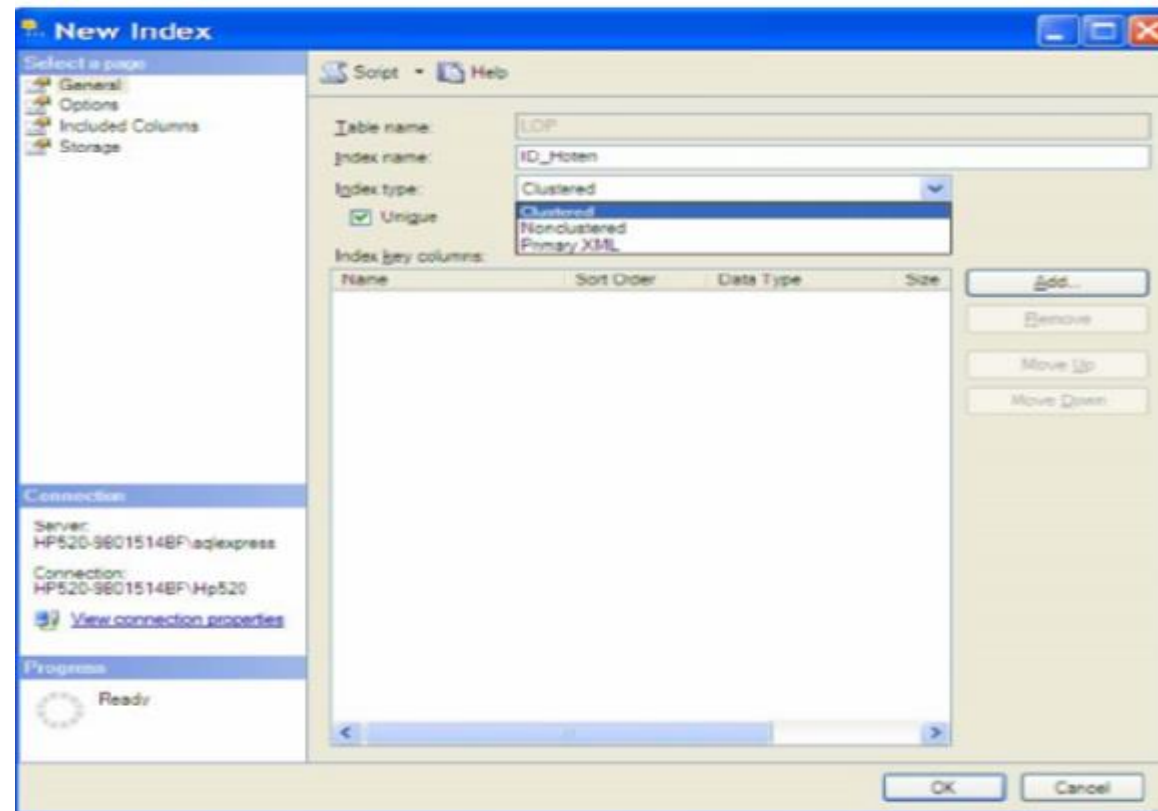
Trong SQL Server 2012, ta có 2 cách khác nhau để tạo:

- + SQL Server Management Studio
- + Dùng T-SQL

# Chỉ mục - Index

## ❖ Dùng SQL Server Management Studio

Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu chứa bảng hoặc view muốn tạo chỉ số index.



*Cửa sổ New Index*

## Chỉ mục - Index

- Mở rộng mục Table hoặc View, mở rộng bảng hoặc view muốn tạo Index. Right Click lên thư mục Indexes chọn New Index, xuất hiện cửa sổ New Index như hình trên. gồm các tham số:

+Table name (hoặc View name đối với view): Tên bảng (hoặc tên view) mà tệp chỉ số được xây dựng trên đó.

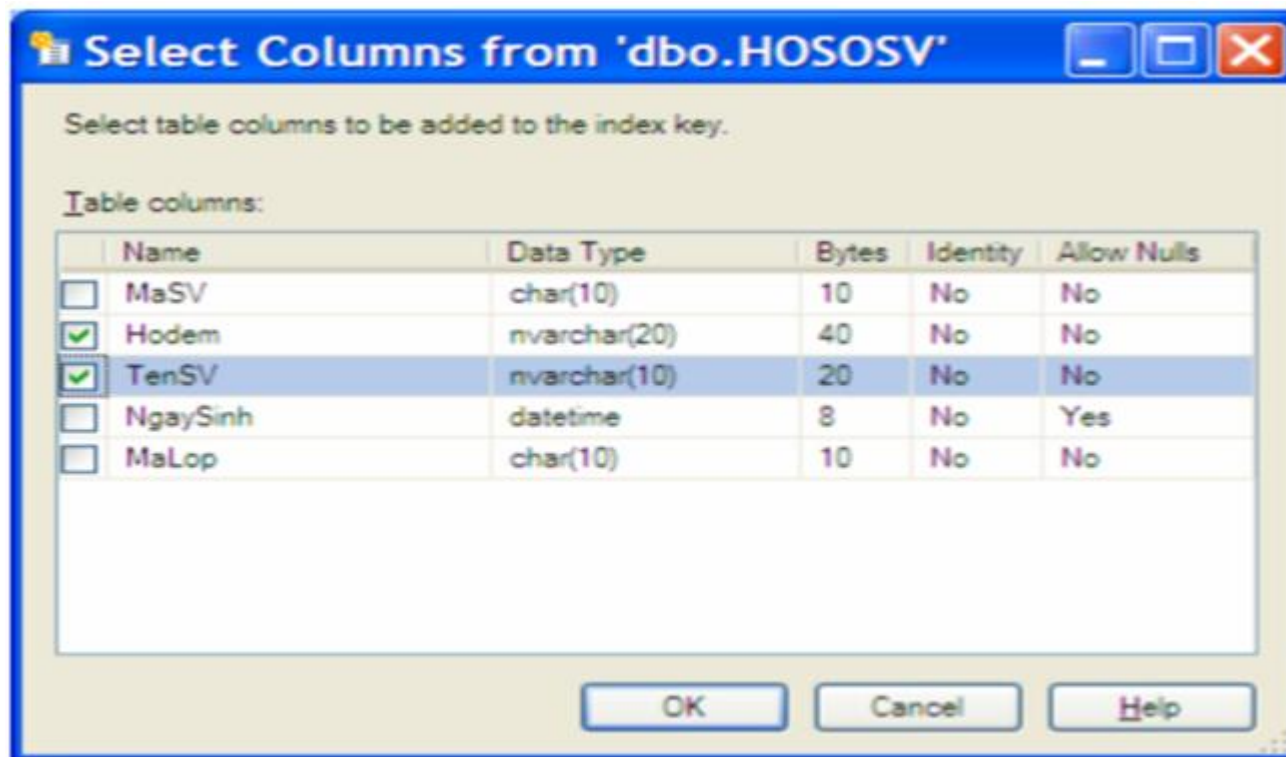
+Index name: Đặt tên tệp chỉ số muốn xây dựng.

+Index type: Kiểu của index (Clustered, Nonclustered)

+Unique: Tùy chọn cho phép tạo tệp chỉ số này có là tệp chỉ số duy nhất hay không?

# Chỉ mục - Index

Index key columns: Xác định các trường khóa của index bằng cách click nút Add xuất hiện cửa sổ Select Columns, ta chọn các cột làm khóa cho Index.



Cửa sổ Select Columns

# Chỉ mục - Index

## Chú ý:

Trong SQL Server 2012, chúng ta có thể mở rộng chức năng của các tệp chỉ số phi liên cung (nonclustered indexes) bằng cách thêm các cột không là khóa (nonkey columns) vào các nút lá của cây nonclustered index. Các cột khóa được lưu trữ tại tất cả các mức còn các cột không khóa chỉ lưu trữ tại mức lá của index. Bằng việc thêm các cột không khóa, ta có thể tạo các tệp chỉ số phi liên cung phủ nhiều truy vấn hơn bởi vì các cột không khóa có các lợi ích sau:

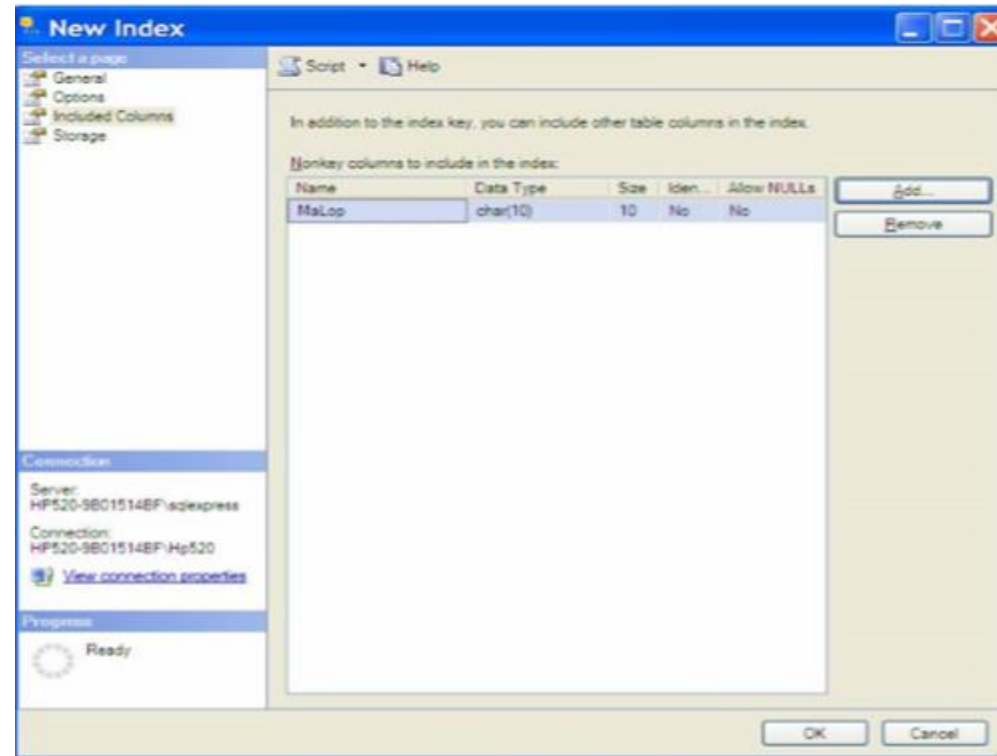
- + Chúng có thể là các cột có kiểu dữ liệu không được phép làm các cột khóa trong index.

- + Chúng không được Database Engine xét khi tính đến số các cột khóa của Index hay kích thước khóa của index. Một index được bao gồm tất cả các cột không khóa có thể cải thiện đáng kể sự thực thi truy vấn khi tất cả các cột trong được bao gồm trong index (cả các cột khóa và không khóa của index).



# Chỉ mục - Index

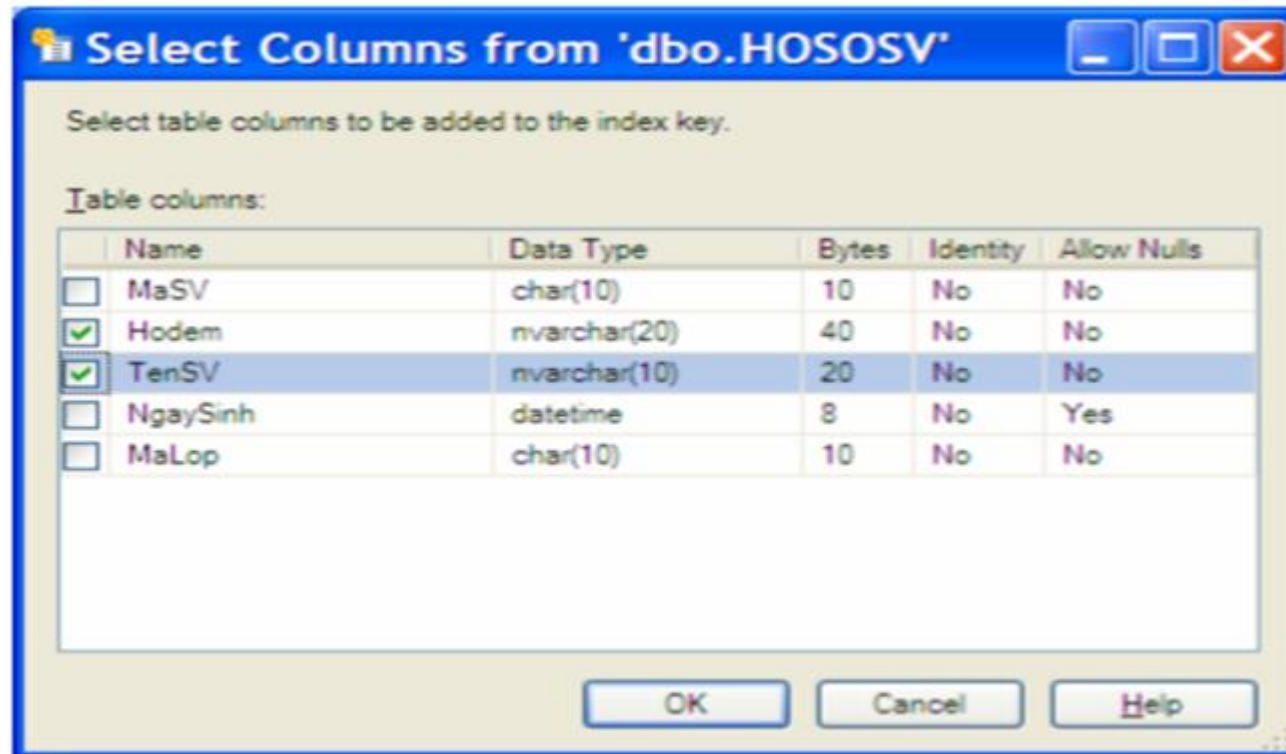
Để thêm các cột không khóa ta chọn trang Included Columns



Cửa sổ New Index

# Chỉ mục - Index

- Sau đó click nút Add để xuất hiện cửa sổ Select Columns.



*Cửa sổ Select Columns*

# Chỉ mục - Index

## ❖ Dùng T - SQL

### - Trên SQL Server 2012:

```
CREATE [ UNIQUE ] [ CLUSTERED | NONCLUSTERED ] INDEX
index_name
    ON <object> ( column [ ASC | DESC ] [ ,...n ] )
    [ INCLUDE ( column_name [ ,...n ] ) ]
    [ WITH ( <relational_index_option> [ ,...n ] ) ]
    [ ON { filegroup_name|default }
    ]
[ ; ]

<object> ::=
{
    [ database_name. [ schema_name ] . | schema_name. ]
    table_or_view_name
}

<relational_index_option> ::=
{
    PAD_INDEX = { ON | OFF }
| FILLFACTOR = fillfactor
| SORT_IN_TEMPDB = { ON | OFF }
| IGNORE_DUP_KEY = { ON | OFF }
| STATISTICS_NORECOMPUTE = { ON | OFF }
| DROP_EXISTING = { ON | OFF }
| ONLINE = { ON | OFF }
| ALLOW_ROW_LOCKS = { ON | OFF }
| ALLOW_PAGE_LOCKS = { ON | OFF }
| MAXDOP = max_degree_of_parallelism
}
```

# Chỉ mục - Index

Các tham số trong đó:

*UNIQUE*

Chỉ định tạo một unique index trên bảng hoặc trên view. Một clustered index trên view buộc phải là unique.

*CLUSTERED*

Chỉ định tạo chỉ mục liên cung.

*NONCLUSTERED*

Chỉ định tạo chỉ mục phi liên cung. Mặc định là chỉ mục NONCLUSTERED.

*index\_name*

Là tên của tệp chỉ số.

*column*

Là tên cột hoặc các cột mà index dựa trên đó.

*INCLUDE ( column [ ,... n ] )*

Chỉ định các cột không khóa được thêm vào mức lá của chỉ mục phi liên cung.

# Chỉ mục - Index

*ON filegroup\_name*

Tạo index trên filegroup chỉ định. Nếu không có chỉ định này thì index sử dụng cùng filegroup mà table hoặc view dựa trên.

*ON "default"*

Tạo index dựa trên filegroup mặc định.

## Ví dụ

Tạo Index trên bảng LOP của CSDL QLDiemSV

```
Use QLDiemSV
create Unique index indTenLopind On LOP (TenLop)
```

## Ví dụ

Xây dựng lại Index TenLop\_ind trên bảng LOP của CSDL QLDiemSV

# Chỉ mục - Index

```
Use QLDiemSV
create Unique index TenLop_ind
    On Lop (TenLop)
    With DROP_EXISTING
```

**Ví dụ:** Dùng từ khóa DBCC DBREINDEX

```
Use QLDiemSV
Go
DBCC DBREINDEX (HOSOSV)
```

# Chỉ mục - Index

## c) Loại bỏ chỉ mục

+Dùng SQL Server Management Studio: Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu chứa bảng hoặc view muốn xóa chỉ số. Mở rộng mục Indexes của bảng hoặc view đó, right click lên Index muốn xóa và chọn Delete.

## + Dùng lệnh T-SQL:

```
DROP INDEX <table.index> | <view.index>
```

**Chú ý:** Đối với SQL Server 2005 ta có thể sử dụng cú pháp sau:

```
DROP INDEX index_name ON <Table|View>
```

**Ví dụ 3.10.** Loại bỏ chỉ mục TenLop\_ind

```
Use QLDiemSV
Go
DROP INDEX Lop. indTenLop
Go
```

# Chỉ mục - Index

## d) Full Text Index

Full Text Indexes được sử dụng trong SQL Server để thực hiện việc tìm kiếm toàn văn (Full Text Searches). Ta đã sử dụng từ khóa LIKE trong các truy vấn để tìm kiếm. Full Text Indexing là một đặc điểm của SQL Server 2012 cho phép các câu truy vấn phức tạp được thực thi trên dữ liệu kiểu ký tự.

Sử dụng Full Text Indexing để thực hiện tìm kiếm theo từng từ, hai hoặc nhiều từ liên nhau, theo đoạn hay các phần của từ. Việc sử dụng Full Text Indexing để tìm kiếm rất hữu dụng khi dữ liệu được biểu diễn bởi một định dạng không cấu trúc. Khi câu truy vấn Full Text được thực hiện, bộ máy tìm kiếm trả về các giá trị khóa mà so khớp với điều kiện tìm kiếm. Đối tượng để cài đặt Full Text Searching là Full Text Index và Full Text Catalog.



## Chỉ mục - Index

+ Full Text Index theo dõi các từ có nghĩa trong bảng. Index này được sử dụng cho các tìm kiếm chứa các từ chỉ mục và các lựa chọn tìm kiếm cao cấp, chẳng hạn như các nhóm tìm kiếm. Một Full Text Index yêu cầu một cột chứa các giá trị khóa duy nhất. Các Full Text Indexes không tự động cập nhật khi có sự thay đổi dữ liệu trên bảng mà ta phải thực hiện Update thủ công.

+ Full Text Catalog: Tất cả các Full Text Indexes được lưu trữ trong các Full Text Indexing. Một Full Text Catalog là một thư mục có thể được view bởi window và Search Service. Theo mặc định tất cả các Full Text Indexes của một CSDL chứa trong một Full Text Catalog, tuy nhiên người quản trị hệ thống có thể chia catalog thành nhiều catalogs khi các indexes quá lớn.

# Chỉ mục - Index

## ❖ Tạo Full Text Index:

Chỉ có một Full Text Index được tạo trên một bảng, tuy nhiên ta có thể index này có thể được tạo trên sự kết hợp của nhiều cột. Tạo full text index bao gồm hai bước: Full Text Catalog và Full Text index.

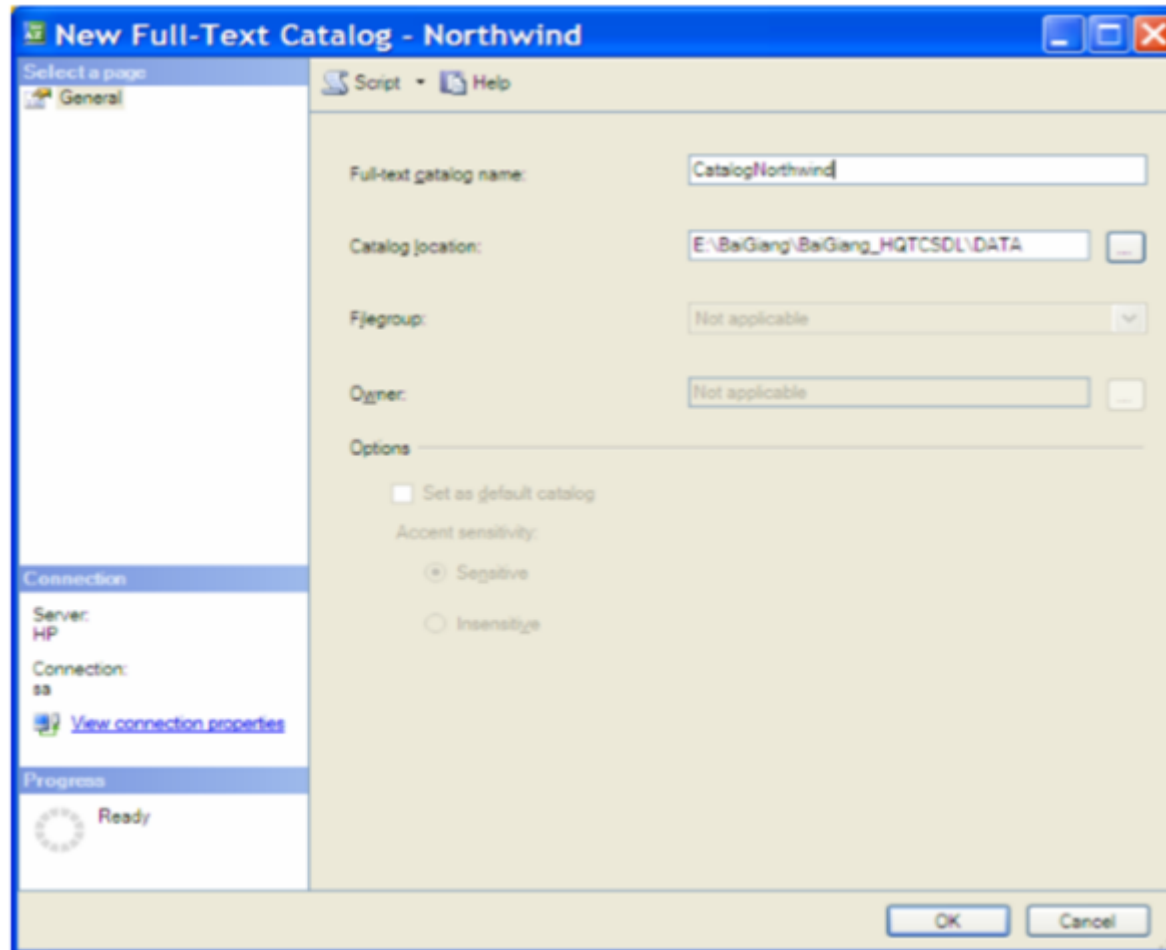
+ Sử dụng Management Studio:

o Tạo Full Text Catalog:

- Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu chứa bảng muốn tạo full text index và mở rộng mục Storage.
- Right click lên mục Full Text Catalogs/ chọn New Full Text Catalog xuất hiện cửa sổ New Full Text Catalog, thực hiện điền các tham số tên và vị trí lưu trữ catalog.

# Chỉ mục - Index

Full text catalog name: Tên của Full text catalog. Catalog location: Vị trí lưu file.

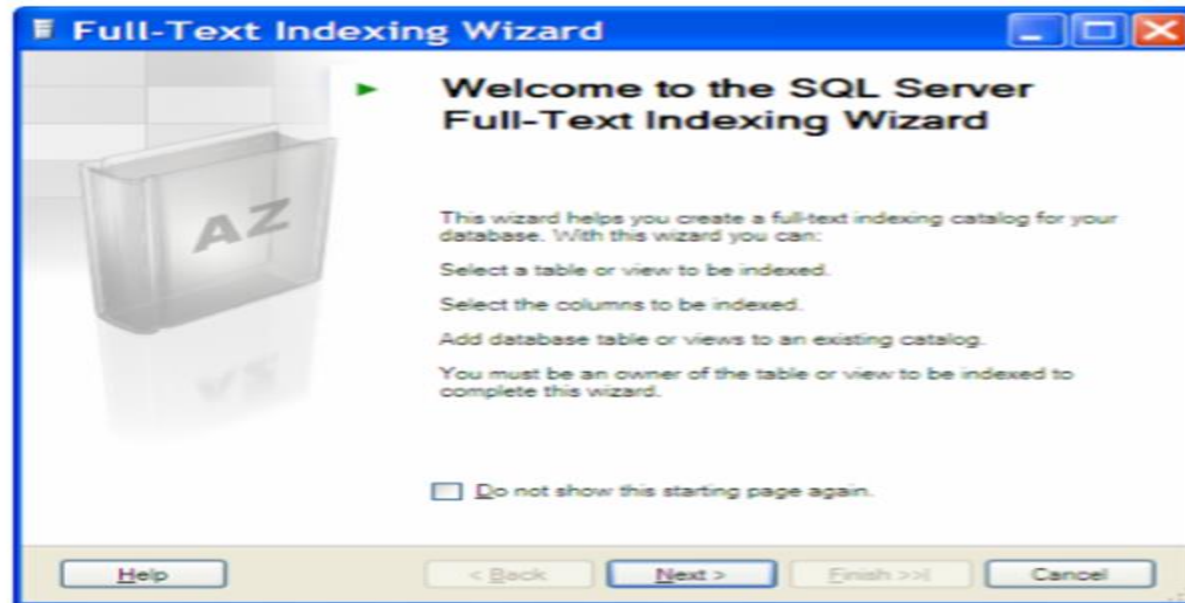


Cửa sổ New Full text catalog

# Chỉ mục - Index

o Tạo Full Text Index:

- Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu chứa bảng muốn tạo full text index.
- Right click trên bảng muốn tạo và chọn Full Text Index\ Define Full Text Index, xuất hiện cửa sổ Welcome to SQL Server Full Text Indexing Wizard.



Cửa sổ Welcome to SQL Server Full Text Indexing Wizard

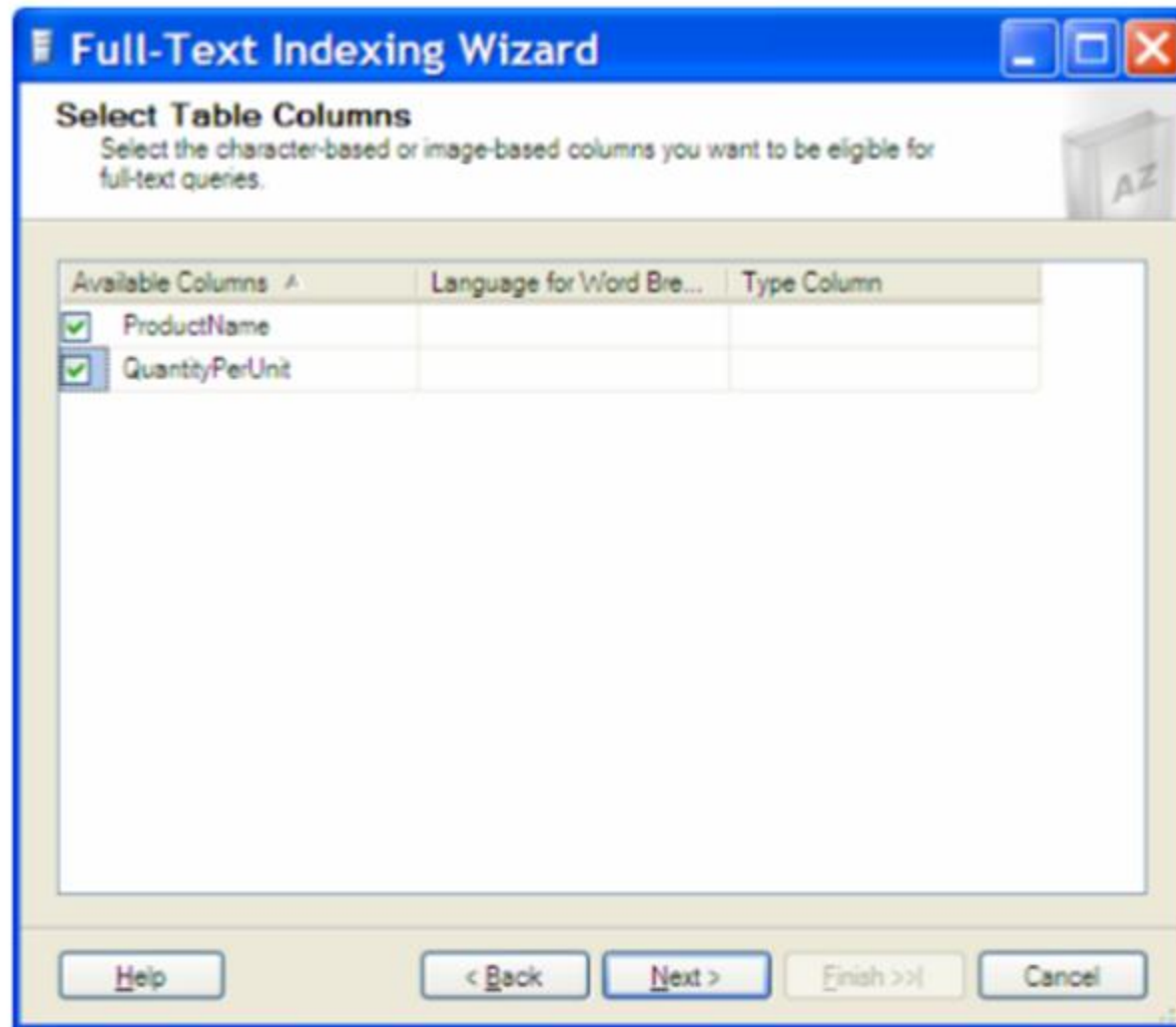
# Chỉ mục - Index

Tiếp theo thực hiện theo các chỉ dẫn của các cửa sổ wizard



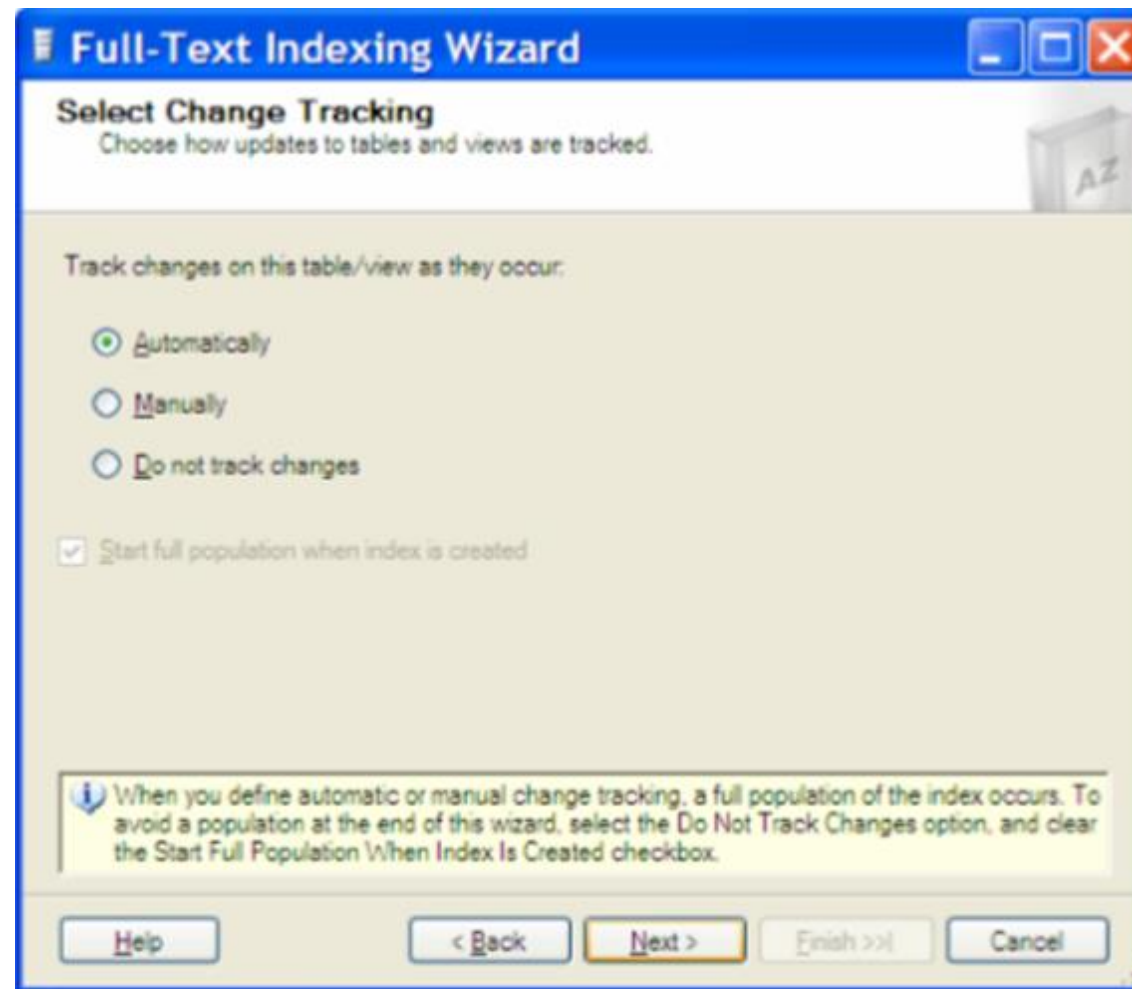
Cửa sổ Select an Index

# Chỉ mục - Index



Cửa sổ Select Table Columns

# Chỉ mục - Index



*Cửa sổ Select Change Tracking*

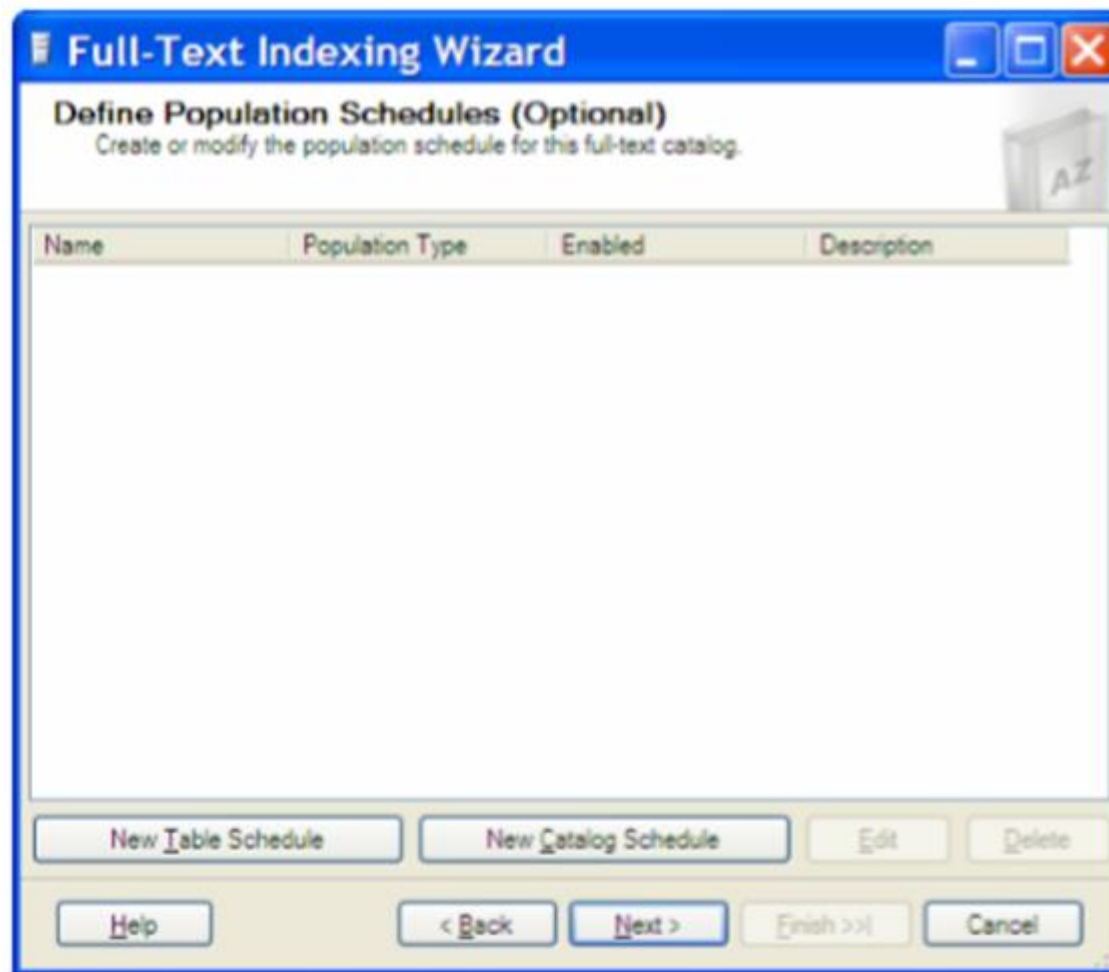
# Chỉ mục - Index



Cửa sổ Select a Catalog

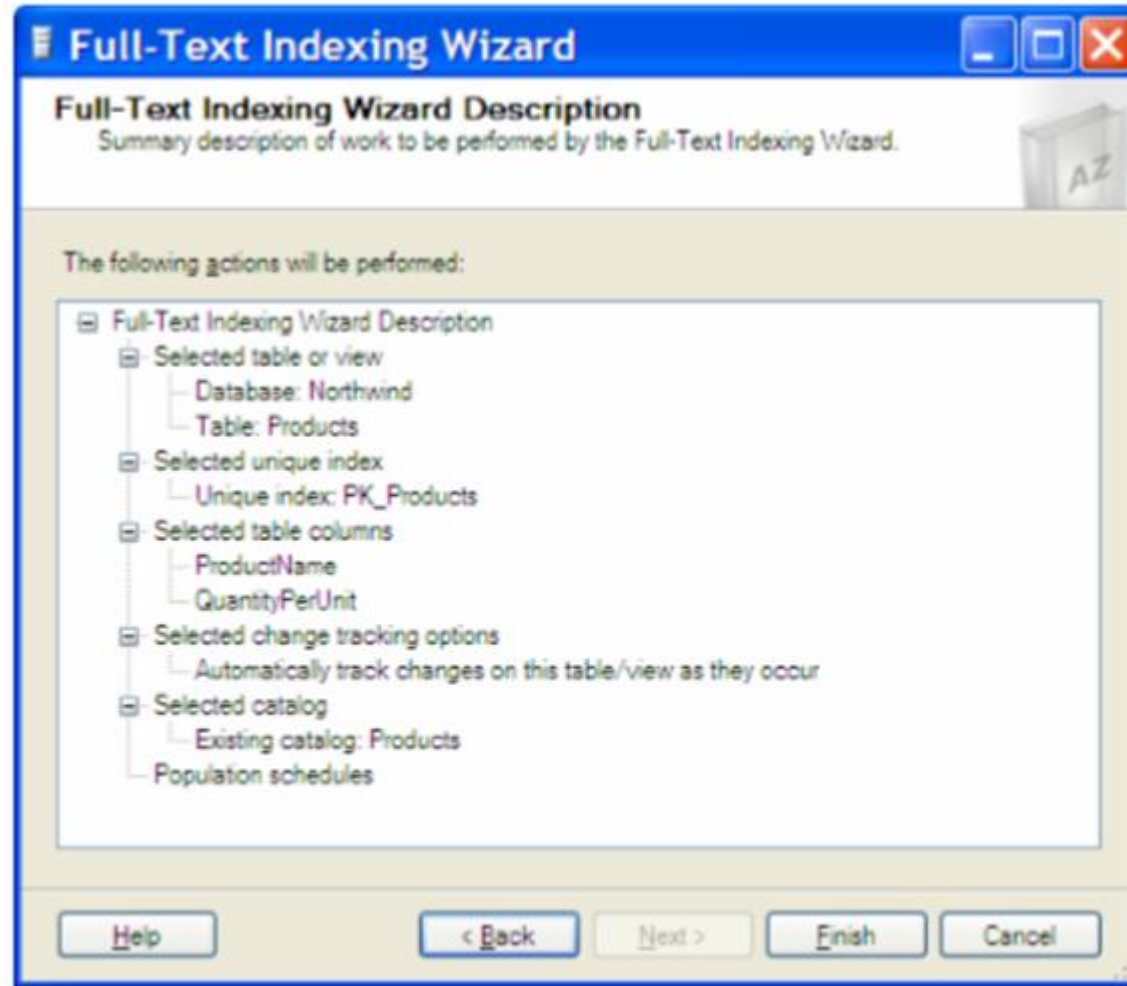


# Chỉ mục - Index



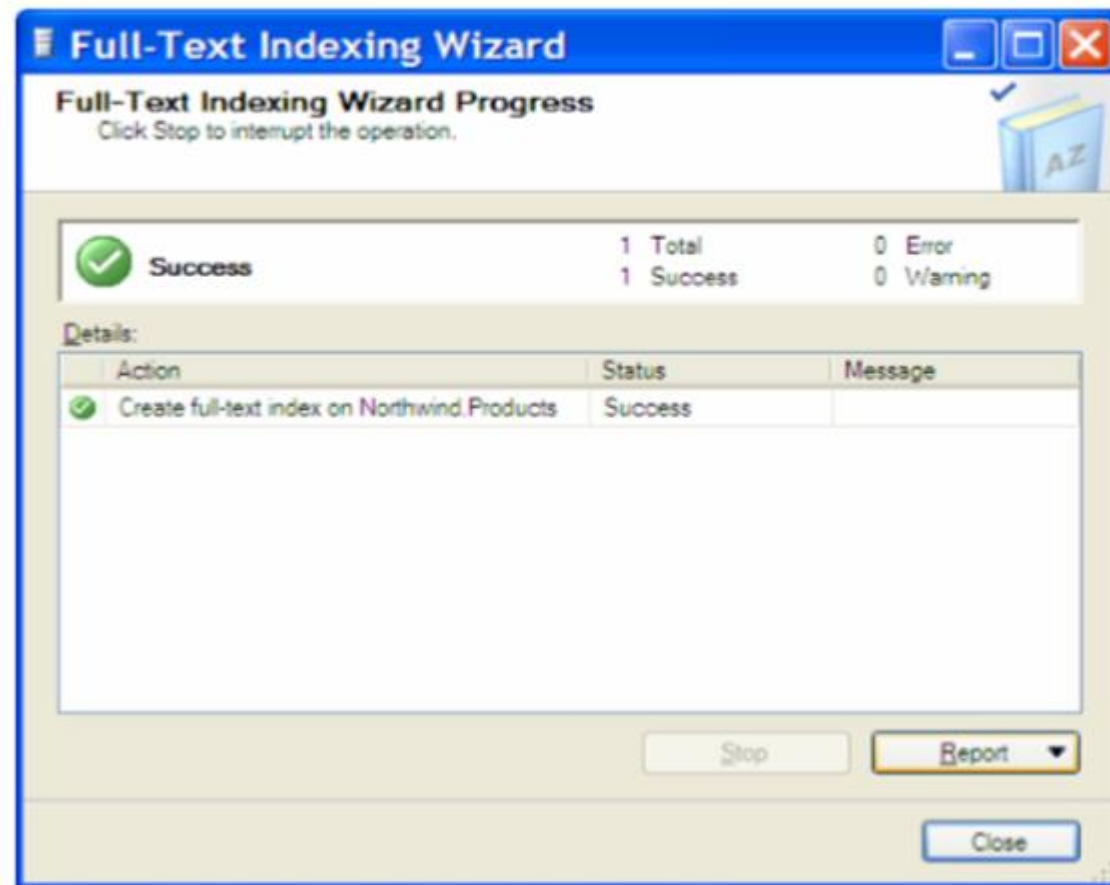
Cửa sổ Define Population Schedules

# Chỉ mục - Index



Cửa sổ Full Text Indexing Wizard Description

# Chỉ mục - Index



- Cửa sổ Full Text Indexing Wizard Progress

# Chỉ mục - Index

## + Sử dụng T SQL:

```
CREATE FULLTEXT CATALOG catalog_name
    [IN PATH 'rootpath']
    [AS DEFAULT]
CREATE FULLTEXT INDEX ON table_name
    [(column_name [,...n])]
    KEY INDEX index_name
    [ON fulltext_catalog_name]
]
```

Trong đó: *catalog\_name*: Là tên của catalog *rootpath*: Chỉ định thư mục gốc chứa catalog. Nếu không chỉ định thư mục này thì sẽ là thư mục mặc định khi cài đặt SQL Server. AS DEFAULT : Chỉ định catalog này là catalog mặc định. *table\_name*: Tên bảng tạo Full Text Index.

## Chỉ mục - Index

*column\_name*: Tên cột hoặc các cột được chứa trong full text index, chỉ có các cột kiểu char, varchar, nchar, nvarchar, text, ntext, image, xml, và varbinary có thể được chứa trong full-text search.

*index\_name*: Là tên của một key index dạng unique trên table\_name. KEY INDEX phải là unique, single-key, non-nullable column. Nên chọn key index unique nhỏ nhất cho full-text unique key, tốt nhất chọn một CLUSTERED index.

*fulltext\_catalog\_name*: Tên của full text catalog chứa full text index đang tạo.

# Chỉ mục - Index

Ví dụ: Tạo full text catalog và full text index

```
USE AdventureWorks;
GO
CREATE UNIQUE INDEX ui_ukJobCand ON
HumanResources.JobCandidate(JobCandidateID);
CREATE FULLTEXT CATALOG ft AS DEFAULT;

CREATE FULLTEXT INDEX ON
HumanResources.JobCandidate(Resume) KEY INDEX
ui_ukJobCand;
```

Ví dụ: Tạo full text catalog và full text index

```
CREATE FULLTEXT CATALOG [CatalogNorthwind]
IN PATH N'E:\BaiGiang\BaiGiang_HQTCSDL\DATA'
WITH ACCENT_SENSITIVITY = ON
AUTHORIZATION [dbo]

Create fulltext index on products (ProductName,
QuantityPerUnit) KEY INDEX PK_Products ON Products
```

\* **Sửa Full Text Index**: Dùng Management Studio hoặc dùng T-SQL

```
+ ALTER FULLTEXT CATALOG catalog_name
    { REBUILD | REORGANIZE | AS DEFAULT }
```

## Chỉ mục - Index

Trong đó: REBUILD: Chỉ định xây dựng lại toàn bộ catalog bằng cách xóa catalog cũ và tạo lại catalog mới. REORGANIZE: Chỉ định kết hợp các chỉ mục nhỏ thành trong tiến trình lập chỉ mục thành một chỉ mục lớn hơn. AS DEFAULT : Chỉ định catalog này là catalog mặc định.

```
+ ALTER FULLTEXT INDEX ON table_name
  {
    ENABLE
  | DISABLE
  | ADD (column_name)
  | DROP (column_name [, ...n] )
  }
```

# Chỉ mục - Index

Trong đó: ENABLE|DISABLE: Chỉ định SQL Server có hoặc không thu thập dữ liệu full text index cho table\_name, ENABLE kích hoạt còn DISABLE tắt . ADD|DROP: Thêm hoặc xóa các cột.

- Xóa Full Text Index: Dùng Management Studio hoặc dùng T-SQL

```
DROP FULLTEXT CATALOG catalog_name  
DROP FULLTEXT INDEX ON table_name
```

*Ví dụ* Xóa full text catalog và full text index

```
IF EXISTS (SELECT * FROM sysfulltextcatalogs ftc WHERE  
ftc.name = N'CatalogNorthwind')  
DROP FULLTEXT CATALOG [CatalogNorthwind]
```



# Chỉ mục - Index

- Tìm kiếm với Full Text Index: Trong mệnh đề WHERE của khối câu lệnh SELECT ta sử dụng các từ khóa CONTAINS hoặc FREETEXT để thực hiện tìm kiếm.
- Từ khóa CONTAINS: Sử dụng từ khóa này để thực hiện tìm kiếm cho + Một từ hoặc cụm từ.
  - + Tiền tố của một từ hoặc cụm từ.
  - + Một từ gần từ khác.
  - + Một từ là biến tố của từ khác (Ví dụ: drive có các biến tố drives, drove, driving, và driven).
  - + Một từ đồng nghĩa với các từ khác, sử dụng từ điển đồng nghĩa (Ví dụ từ metal (Tiếng Anh) đồng nghĩa với các từ aluminum và steel).

# Chỉ mục - Index

Cú pháp:

```
CONTAINS ( {column_name|(column_list)|* }  
          , '< contains_search_condition >')
```

```
< contains_search_condition > ::=  
    { < simple_term >  
      | < prefix_term >  
      | < generation_term >  
      | < proximity_term >  
      | < weighted_term >  
    }  
  
    | { ( < contains_search_condition > )  
      [ { < AND > | < AND NOT > | < OR > } ]  
      < contains_search_condition > [ ...n ]  
    }  
  
< simple_term > ::= word | " phrase "  
< prefix_term > ::= { "word * " | "phrase *" }  
< generation_term > ::=  
    FORMSOF ({INFLECTIONAL|THESAURUS},<simple_term>[  
    ,...n ] )  
< proximity_term > ::= {<simple_term>|<prefix_term>}  
    { { NEAR | ~ }{<simple_term >|<prefix_term>}}
```

# Chỉ mục - Index

```
[ ...n ]  
< weighted_term > ::= ISABOUT  
    ( { {  
    < simple_term >  
    | < prefix_term >  
    | < generation_term >  
    | < proximity_term >  
    }  
    [ WEIGHT ( weight_value ) ]  
    } [ ,...n ]  
    )  
< AND > ::=  
    { AND | & }  
< AND NOT > ::=  
    { AND NOT | & ! }  
< OR > ::=  
    { OR | | }
```

## Ví dụ: Tìm kiếm sử dụng CONTAINS

```
USE AdventureWorks;
GO
SELECT Name, ListPrice
FROM Production.Product
WHERE ListPrice = 80.99
    AND CONTAINS(Name, 'Mountain');
SELECT Name
FROM Production.Product
WHERE CONTAINS(Name, '"Mountain Frame"OR"Road" ');
SELECT Name
FROM Production.Product
WHERE CONTAINS(Name, ' "Chain*" ');
SELECT Description
FROM Production.ProductDescription
WHERE CONTAINS(Description, 'bike NEAR
performance');
SELECT Description
FROM Production.ProductDescription
WHERE CONTAINS(Description, ' FORMSOF (INFLECTIONAL,
ride) ');
SELECT Description
FROM Production.ProductDescription
WHERE CONTAINS(Description, 'ISABOUT (performance
weight (.8), comfortable weight (.4), smooth weight
(.2) ) ');
GO
```

# Chỉ mục - Index

```
USE AdventureWorks;  
GO  
DECLARE @SearchWord varchar(30)  
SET @SearchWord = 'Performance'  
SELECT Description FROM Production.ProductDescription  
WHERE CONTAINS (Description, @SearchWord);  
GO
```

Từ khóa FREETEXT: Thực hiện tìm kiếm đối với các kiểu dữ liệu ký tự, so khớp theo nghĩa không nhất thiết phải chính xác theo từ của chuỗi tìm kiếm. Khi FREETEXT được sử dụng, bộ máy tìm kiếm thực thi các hành động sau trên chuỗi tìm kiếm: gán cho mỗi từ một trọng số sau đó mới so khớp.

- + Chia chuỗi tìm kiếm thành từng từ.
- + Sinh các biến tố của các từ đó.
- + Nhận dạng danh sách mở rộng hoặc thay thế các mục đồng nghĩa trong chuỗi so khớp.

# Chỉ mục - Index

## Cú pháp:

```
FREETEXT({column_name | (column_list) | * },  
         'freetext_string' )
```

## Ví dụ: Tìm kiếm sử dụng FREETEXT

```
USE AdventureWorks;  
GO  
DECLARE @SearchWord varchar(30);  
SET @SearchWord = 'Install';  
SELECT Title FROM Production.Document WHERE  
FREETEXT(Title, @SearchWord);
```

# Con trỏ - Cursor

## 2.4 Con trỏ - Cursor

### 2.4.1 Khai báo con trỏ

#### 2.4.1.1 Đặt vấn đề:

Các lệnh trong SQL như: Select, update, delete,... đều thao tác lên nhiều dòng dữ liệu thỏa điều kiện where cùng lúc mà không thể thao tác lên từng dòng dữ liệu cụ thể

#### 2.4.1.2 Khái niệm:

Cursor là kiểu dữ liệu cơ bản dùng để duyệt qua từng dòng dữ liệu trả về từ câu truy vấn SELECT đó giúp ta có thể có những xử lý khác nhau cho từng dòng dữ liệu cụ thể

# Con trỏ - Cursor

## ❖ Đặc điểm:

Cho phép thao tác lên từng dòng dữ liệu trả về từ lệnh SELECT.

Do phải lặp qua từng dòng dữ liệu nên đây là cách xử lý chậm nhất

### 2.4.1.2 Khai báo con trỏ

#### ➤ Khai báo chuẩn

```
DECLARE cursor_name[ INSENSITIVE ][ SCROLL ] CURSOR
    FOR select_statement
    [ FOR { READ ONLY | UPDATE [ OF
        column_name [ ,...n ] ] } ]
[;]
```

- cursor\_name : tên của cursor, phải chứa 1 - 128 ký tự
- select\_statement : lựa chọn đến các cột trong bảng mà ta cần đọc. Nhớ lệnh SELECT trong cursor không chứa các mệnh đề COMPUTE, COMPUTE BY, FOR BROWSE, and INTO .



## Con trỏ - Cursor

Trong đó:

**INSENSITIVE:** Tạo một bản copy tạm thay thế bảng chính. Con trỏ này sẽ không được phép thay đổi và việc thay đổi sẽ không ảnh hưởng khi dùng lệnh **FETCH** lấy giá trị con trỏ này.

**READ\_ONLY:** Con trỏ không thể cập nhật.

**SCROLL:** Cursor được phép di chuyển tới lui, qua lại các dòng mẫu tin bên trong cursor.

**UPDATE[OF Column\_name[,...n,]]** để xác định khả năng cập nhật từng cột với con trỏ. Nếu **OF Column\_name** [,...n,]] được chỉ định. Thì chỉ các cột trong danh sách được sửa. Nếu có lệnh **UPDATE** mà không có danh sách cột thì mặc định tất cả cột có thể chỉnh sửa.

## Con trỏ - Cursor

### ▣ Ví dụ :

```
DECLARE vend_cursor CURSOR  
    FOR SELECT * FROM Purchasing.Vendor
```

### ▣ Ngoài ra, có thể khai báo riêng CURSOR rồi sau đó mới gán lệnh SELECT vào CURSOR sau

```
DECLARE @cursor_name CURSOR  
SET @cursor_name = CURSOR FOR select_statement
```

# Con trỏ - Cursor

## ➤ Khai báo mở rộng

```
DECLARE cursor_name CURSOR  
[LOCAL | GLOBAL]  
[FORWARD_ONLY | SCROLL]  
[STATIC | KEYSET | DYNAMIC ]  
[READ_ONLY | SCROLL_LOCK]  
FOR select_statement  
[FOR UPDATE [OF column name[,...n]]]  
[;]
```

# Con trỏ - Cursor

## ➤ Phạm vi

`LOCAL` | `GLOBAL` : phạm vi hoạt động của biến. local biến cục bộ; global biến toàn cục (tham chiếu đến bất kỳ thủ tục nào của kết nối tạo ra biến cursor đó.

Mặc định sẽ là `LOCAL`

`FORWARD_ONLY` : duyệt từ mẫu tin đầu đến cuối cùng , theo chiều đi tới.

`SCROLL`: cursor được phép di chuyển tới lui, qua lại các dòng mẫu tin bên trong cursor.



## Con trỏ - Cursor

**STATIC** (cursor tĩnh) : Nghĩa là khi có sự thay đổi bên dưới dữ liệu gốc (base table) thì các thay đổi đó không được cập nhật tự động trong dữ liệu của cursor.

**DYNAMIC** (cursor động) : Nghĩa là khi có thay đổi dữ liệu gốc(base table) thì các thay đổi đó tự động cập nhật trong dữ liệu kiểu cursor.

**KEYSET** gần như **DYNAMIC**. Nghĩa là những thay đổi trên cột không là khoá chính trong bảng gốc (base table) sẽ tự động cập nhật trong dữ liệu cursor. Tuy nhiên hiển thị trong những mẫu tin vừa thêm mới hoặc những mẫu tin vừa huỷ bỏ sẽ không hiển thị trong dữ liệu cursor có kiểu là keyset.

## Con trỏ - Cursor

`SCROLL_LOCK`: chỉ định SQL SERVER khoá các mẫu tin cần phải thay đổi giá trị hoặc bị huỷ bỏ bên trong bảng nhằm đảm bảo hành động cập nhật luôn thành công.

`READ_ONLY` : không thể cập nhật dữ liệu

`select_statement`: chỉ định danh sách các cột sẽ được phép thay đổi giá trị trong cursor. Mặc định là tất cả các cột trong mệnh đề select sẽ được phép thay đổi giá trị nếu dữ liệu cursor không phải là chỉ đọc.



# Con trỏ - Cursor

Ví dụ:

```
-- Declare our cursor
DECLARE CursorTest CURSOR
GLOBAL -- So we can manipulate it outside
the batch
SCROLL -- So we can scroll back and see the
changes
DYNAMIC
FOR SELECT SalesOrderID, CustomerID
FROM CursorTable;
```

# Con trỏ - Cursor

## 2.4.2 Sử dụng con trỏ

### 2.4.2.1 Mở con trỏ

```
OPEN { { [ GLOBAL ] cursor_name } |  
@cursor_variable_name }
```

- ▣ **GLOBAL**: Nếu cả biến toàn cục và cục bộ cùng một tên thì mặc định sẽ gọi biến cục bộ, thêm GLOBAL sẽ gọi theo tên theo biến toàn cục
- ▣ **cursor\_variable\_name** : tên của biến cursor mà tham chiếu đến một cursor
- ▣ Mặc định sẽ là **LOCAL**



# Con trỏ - Cursor

## 2.4.2.2 Cách sử dụng con trỏ

### ➤ Truy cập con trỏ

```
FETCH [NEXT | PRIOR | FIRST | LAST |  
ABSOLUTE {n | @nVar}  
      | RELATIVE {n | @nVar}]  
FROM {[GLOBAL] cursor_name} |  
@cursor_variable_name  
[ INTO @variable_name[,...n]]
```

## Con trỏ - Cursor

- ▣ **NEXT, PRIOR, FIRST, LAST**: chỉ định cách đọc dữ liệu.
- ▣ **ABSOLUTE {n | @nVar}** : Chỉ định số dòng n dữ liệu cần đọc, được đọc từ dòng đầu tiên..

n or @nVar	Mô tả
0	Nếu 0 thì không có giá trị trả về
< 0	Nếu giá trị dương thì được tính từ đỉnh của phần dữ liệu
> 0	Nếu n âm thì được tính từ phần đáy của dữ liệu

- ▣ **RELATIVE** : tương tự như **ABSOLUTE** nhưng bắt đầu từ dòng hiện tại

# Con trỏ - Cursor

**@@FETCH\_STATUS** : biến hệ thống để kiểm tra đọc dữ liệu thành công hay thất bại

Giá trị trả về	Mô tả
0	Câu lệnh FETCH thành công
-1	Câu lệnh FETCH thất bại hoặc dòng đã vượt quá kết quả gán
-2	Dòng truy cập bị xóa

## Ví dụ

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

# Con trỏ - Cursor

## 2.4.2.2 Đóng con trỏ và giải phóng bộ nhớ

### 2.4.2.2.1 Đóng con trỏ

```
CLOSE { { [ GLOBAL ] cursor_name } |  
        @cursor_variable_name }
```

- ▣ Lưu ý : Lệnh `CLOSE` chỉ là thực hiện hành động giải phóng các dòng dữ liệu tham chiếu bên trong biến cursor.

#### ◉ Ví dụ :

```
CLOSE Employee_Cursor
```

# Con trỏ - Cursor

## 2.4.2.2.2 Giải phóng bộ nhớ

```
DEALLOCATE{ { [ GLOBAL ] cursor_name } |  
            @cursor_variable_name }
```

- ▣ Lưu ý : Lệnh `DEALLOCATE` để giải phóng thật sự biến cursor ra khi bộ nhớ. Sau khi thực hiện lệnh này , nếu có lệnh nào tham chiếu đến tên cursor đều sẽ gây ra lỗi.

### ⦿ Ví dụ

```
DEALLOCATE Employee_Cursor
```

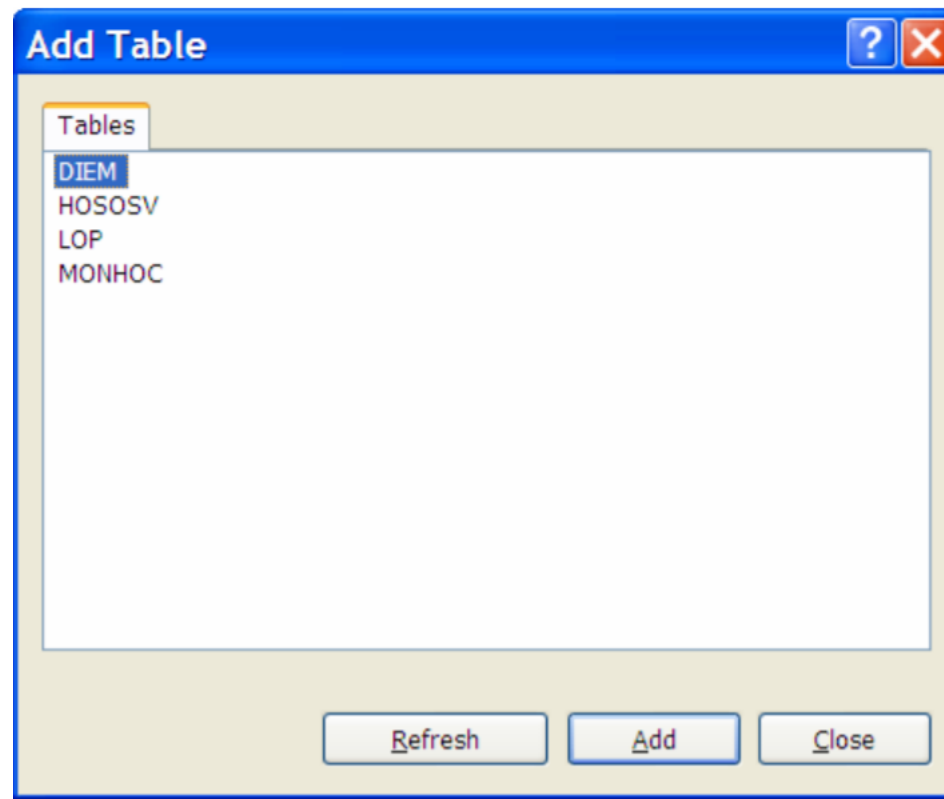
# Lược đồ - Diagrams

## 2.5 Lược đồ - Diagrams

### 2.5.1 Tạo lược đồ:

\*Dùng SQL Server Management Studio:

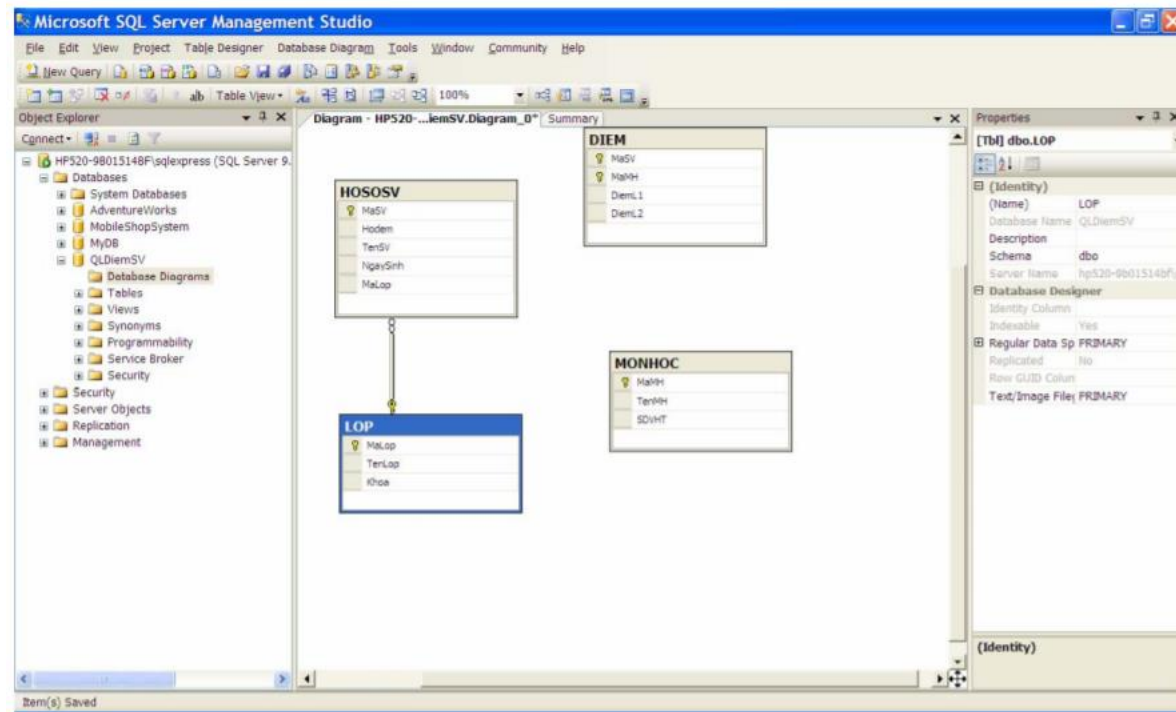
Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu muốn tạo Database Diagrams. Right click và chọn New Database Diagram xuất hiện cửa sổ Add Table hình





# Lược đồ - Diagrams

- Chọn các bảng xây dựng lược đồ thông qua nút Add. Xuất hiện cửa sổ thiết kế Diagram



# Lược đồ - Diagrams

- Ta thực hiện thiết kế mối quan hệ giữa các bảng trong cơ sở dữ liệu bằng việc kéo và giữ trường của bảng này thả sang trường tương ứng của bảng khác xuất hiện cửa sổ Table and Columns. Ta thực hiện điều chỉnh các tham số cho mỗi quan hệ đó.

The screenshot shows a dialog box titled "Tables and Columns". It has a blue title bar with a question mark icon and a close button. The main area is light beige. It contains the following fields:

- Relationship name:** A text box containing "FK\_HOSOSV\_LOP1".
- Primary key table:** A dropdown menu showing "LOP".
- Foreign key table:** A text box containing "HOSOSV".
- Below these, there are two columns for mapping columns. The first column is labeled "MaLop" and the second column is also labeled "MaLop".
- At the bottom right, there are two buttons: "OK" and "Cancel".

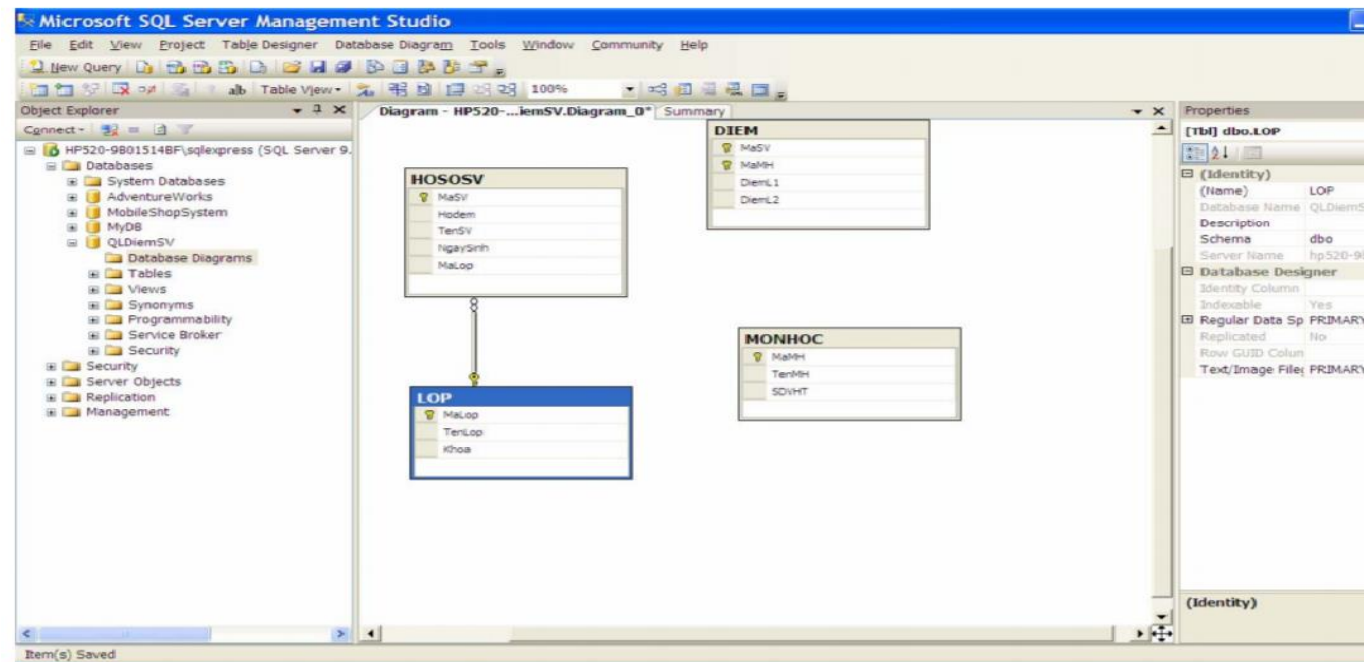


# Lược đồ - Diagrams

## b) Chỉnh sửa lược đồ:

### \*Dùng SQL Server Management Studio:

Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu sửa đổi Database Diagrams. Right click vào Diagram muốn sửa đổi và chọn Modify xuất hiện cửa sổ thiết kế Database Diagram như hình ta thực hiện sửa đổi trên cửa sổ này.



Cửa sổ thiết kế Diagram

## Lược đồ - Diagrams

c) Xóa lược đồ:

\*Dùng SQL Server Management Studio:

Trong SQL Server Management Studio, mở rộng danh mục Database, mở rộng cơ sở dữ liệu sửa đổi Database Diagrams. Right click vào Diagram muốn sửa đổi và chọn Delete xuất hiện cửa sổ xác nhận xóa và chọn OK.