

Chương 1: Giới thiệu SQL Server 2005

- Các phiên bản khác nhau của SQL Server 2005
- Cài đặt SQL Server 2005; Các thành phần của SQL Server 2005
- Các CSDL hệ thống; Cấu trúc vật lý của một DB
- Nguyên tắc hoạt động của Transaction log; Check Point
- Tham số Recovery Interval
- Cấu trúc logic của một DB
- Tạo CSDL bằng T-SQL, Management Studio
- Collation
- FileGroup
- Bảo trì CSDL
- Mô hình xác thực NSD
- Mô hình truy cập CSDL
- Tạo các partitioned table
- Select với từ khóa WITH

Các phiên bản khác nhau của SQL Server 2005

Các phiên bản 32 bit:

SQL Server 2005 (32-bit)	Processor type	Processor speed	Memory (RAM)
SQL Server 2005 Enterprise Edition SQL Server 2005 Developer Edition SQL Server 2005 Standard Edition	Pentium III compatible processor or higher required	Minimum: 500 MHz Recommended: 1 GHz or higher	Minimum: 512 MB Recommended: 1 GB or more Maximum: OS maximum : OS
SQL Server 2005 Workgroup Edition	Pentium III compatible processor or higher required	Minimum: 500 MHz Recommended: 1 GHz or higher	Minimum: 512 MB Recommended: 1 GB or more Maximum: 3 GB
SQL Server 2005 Express Edition	Pentium III compatible processor or higher required	Minimum: 500 MHz Recommended: 1 GHz or higher	Minimum: 128 MB Recommended: 512 MB or more Maximum: 1 GB

Các phiên bản khác nhau của SQL Server 2005

Các phiên bản 64 bit:

QL Server 2005 (64-bit)	Processor type	Processor speed	Memory (RAM)
SQL Server 2005 Enterprise Edition ⁴	IA64 minimum: Itanium processor or higher	IA64 minimum: 733 MHz	IA64 minimum: 512 MB
SQL Server 2005 Developer Edition	X64 minimum: AMD Opteron, AMD Athlon 64, Intel Xenon with Intel EM64T support, Intel Pentium IV with EM64T support	IA64 recommended: 733 MHz or more	IA64 recommended: 1 GB or more
SQL Server 2005 Standard Edition		X64 minimum: 1 GHz	IA64 maximum: 32 TB
		X64 recommended: 1 GHz or more	X64 minimum: 512 MB
			X64 recommended: 1 GB or more
			X64 maximum: 32 TB

So sánh giữa các phiên bản

Tính năng	Express	Workgroup	Standard	Developer	Enterprise
CPU	1	2	4	Không giới hạn	Không giới hạn
RAM	1GB	3GB	Không giới hạn	Không giới hạn	Không giới hạn
Kích thước CSDL	4GB	Không giới hạn	Không giới hạn	Không giới hạn	Không giới hạn
Phân vùng	Không	Không	Không	Có	có

Cài đặt – Yêu cầu HĐH

	Enterprise	Developer	Standard	Workgroup	Express
Windows 2000	No	No	No	No	No
Windows 2000 Professional SP4	No	Yes	Yes	Yes	Yes
Windows 2000 Server SP4	Yes	Yes	Yes	Yes	Yes
Windows 2000 Advanced Server SP4	Yes	Yes	Yes	Yes	Yes
Windows 2000 Datacenter SP4	Yes	Yes	Yes	Yes	Yes
Windows XP Professional SP2	No	Yes	Yes	Yes	Yes

Xem thêm trong file RequirementsSQL2005.htm

Cài đặt

- Cài đặt .Net 2.0
- Lựa chọn các thành phần của SQL Server 2005 cần cài đặt
- Lựa chọn User
- Lựa chọn kiểu xác thực người sử dụng
- Lựa chọn collation

Các thành phần của SQL Server 2005

- SQL Database Engine
- SQL Server Analysis Services
- SQL Server Integration Services
- SQL Server Reporting Services
- SQL Server Notification Services
- SQL Server Service Broker (Workload distribution)
- Full Text Search
- SQL Server Tools and Utilities
- MS SQL Server Management (Object Explorer, Database Engine Query, SQL Profiler, Query Analyzer, Analysis Services)
- SQL Server Books Online

Các CSDL hệ thống

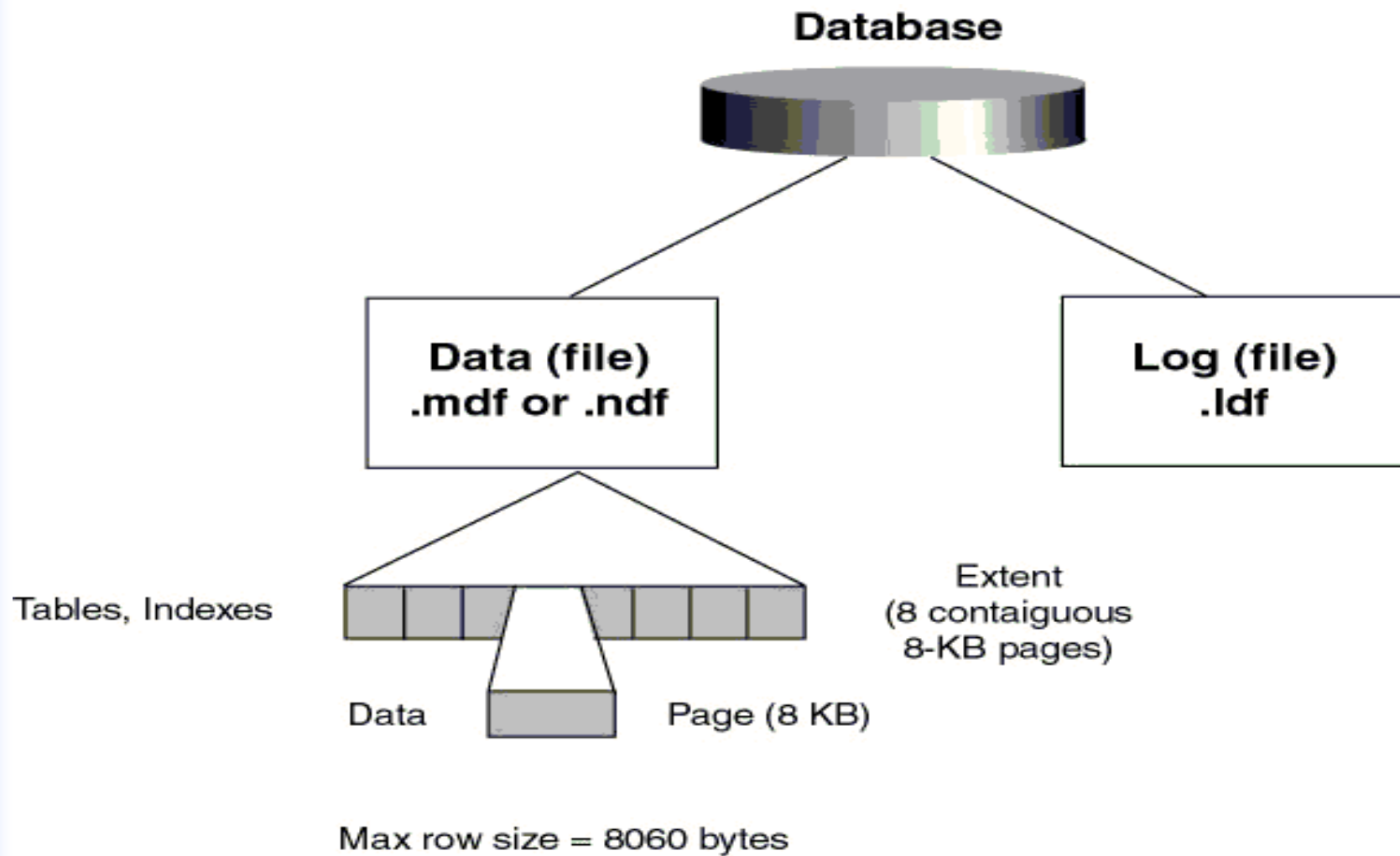
- Master: Chứa tất cả những thông tin mức hệ thống bao gồm thông tin về các database khác trong hệ thống, vị trí của các data files, các tài khoản đăng nhập và các thiết lập cấu hình hệ thống của SQL Server.
- Tempdb: Chứa tất cả những table hay stored procedure được tạm thời tạo ra trong quá trình làm việc bởi user hay do bản thân SQL Server engine. Các table hay stored procedure này sẽ biến mất khi khởi động lại SQL Server hay khi ta disconnect.
- Model: Database này đóng vai trò như một mẫu (template) cho các database khác. Nghĩa là khi một database được tạo ra thì SQL Server sẽ copy toàn bộ các system objects (tables, stored procedures...) từ Model database sang database mới vừa tạo.
- Msdb: Database này được SQL Server Agent sử dụng để lưu các công việc cần làm, các sự kiện về sao lưu phục hồi dữ liệu.
- Resource: (mới có) Là CSDL chỉ đọc, chứa các đối tượng hệ thống về các Service pack.
- Distribution: (mới có) chỉ áp dụng đối với các SQL Server trong một hệ thống phân tán.

Cấu trúc vật lý của một DB

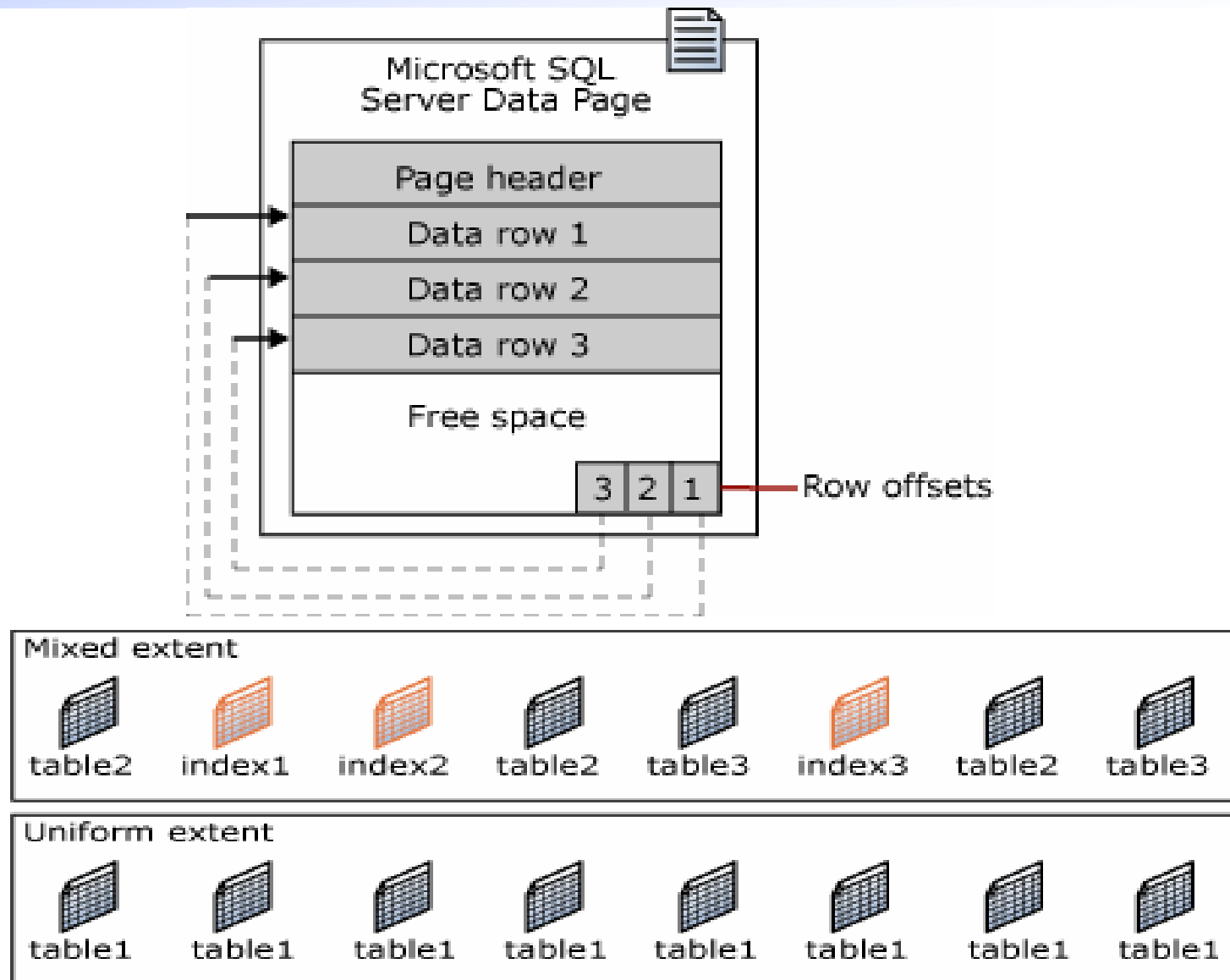
Một Database có 3 kiểu file:

- Primary data file (có phần mở rộng .mdf): đây là file chính chứa data và những system tables.
- Secondary data file (có phần mở rộng .ndf): đây là file phụ thường chỉ sử dụng khi database được phân chia để chứa trên nhiều đĩa.
- Transaction log file (có phần mở rộng .ldf): đây là file ghi lại tất cả những thay đổi diễn ra trong một database và chứa đầy đủ thông tin để có thể khôi phục lại CSDL khi cần.

Mô hình lưu trữ CSDL



Mô hình lưu trữ CSDL (2)



Mô hình lưu trữ (3)

- Dữ liệu sẽ được lưu trữ trong data file và log file
- Với một CSDL có tối thiểu 2 file (mdf, ldf)
- Các thông tin cơ bản về CSDL:
 - Khi tạo CSDL, toàn bộ CSDL Model được sao chép sang.
 - Dữ liệu được lưu trong các khối 8KB. (1MB -128 trang)
 - Trong SQL Server 2000 và các phiên bản trước một bản ghi không thể chiếm nhiều trang. Do đó độ dài tối đa của 1 bản ghi là 8060 (thực tế là 8192 nhưng phải cắt 132 byte để lưu trữ các thông tin hệ thống). Trong SQL Server 2005 với các bảng có chứa các trường kiểu **varchar**, **nvarchar**, **varbinary**, **sql_variant**, or **CLR user-defined type columns** kích thước của mỗi bản ghi có thể vượt quá 8KB.
- Các bảng, các đối tượng dữ liệu khác được lưu trữ trong các vùng (8 trang)

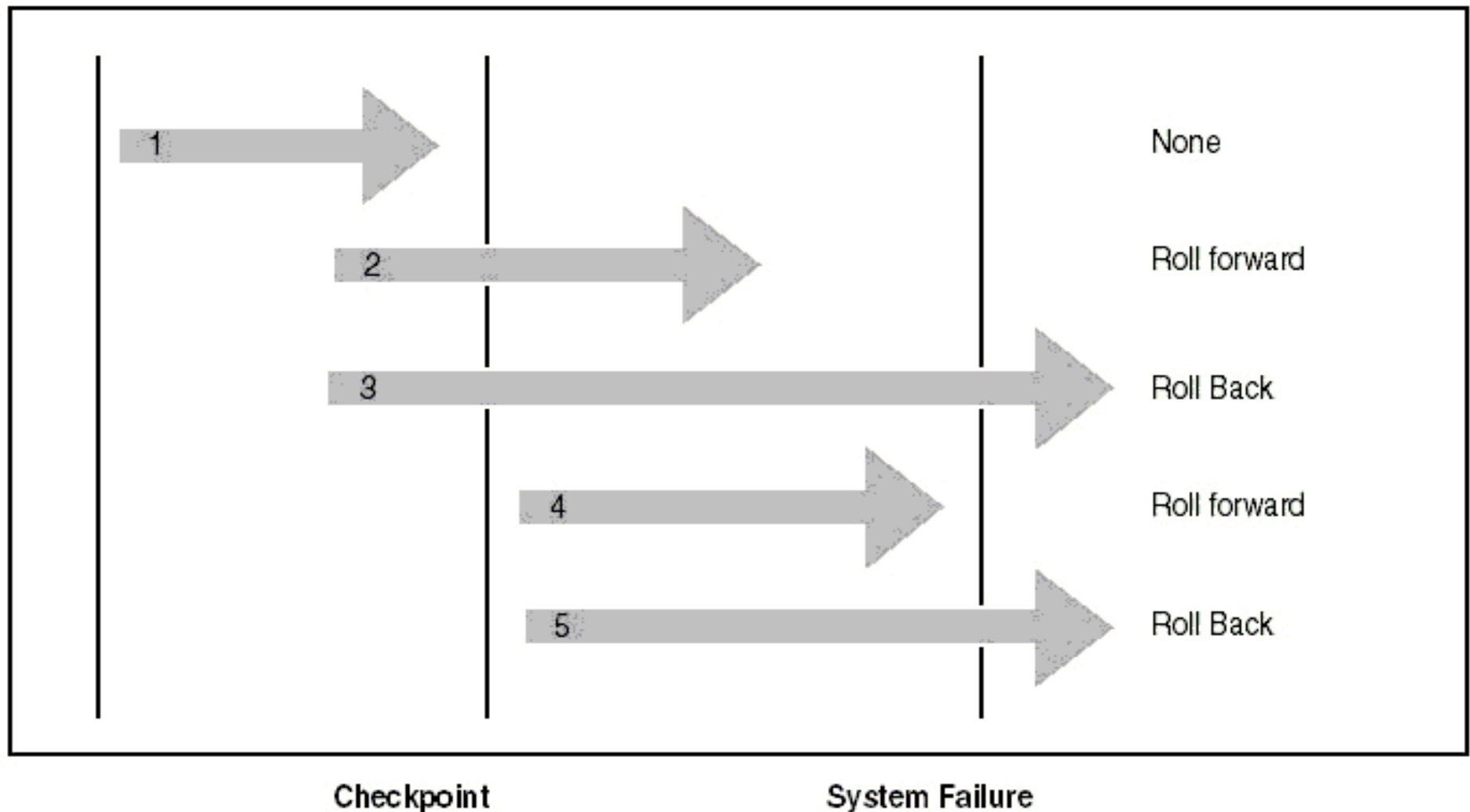
Transaction log trong SQL Server

Transaction log file trong SQL Server dùng để ghi lại các thay đổi xảy ra trong database. Quá trình này diễn ra như sau:

- Đầu tiên khi có một sự thay đổi data như: Insert, Update, Delete được yêu cầu từ các ứng dụng, SQL Server sẽ tải (load) data page tương ứng lên memory (vùng bộ nhớ này gọi là data cache), sau đó data trong data cache được thay đổi (những trang bị thay đổi còn gọi là *dirty-page*). Những *dirty-page* được ghi vào đĩa bởi một **Lazy writer**. **Lazy writer** làm việc theo một chu kỳ nhất định .
- Tiếp theo mọi sự thay đổi đều được ghi vào transaction log file cho nên người ta gọi là *write-ahead* log.
- Cuối cùng thì một quá trình gọi là **Check Point** sẽ kiểm tra và ghi tất cả những transaction đã được committed (hoàn tất) từ log file vào data file (flushing the page).

Minh họa hoạt động của CheckPoint

Transaction Recovery



CheckPoint

Checkpoints xuất hiện trong các tình huống sau:

- Thực hiện lệnh CHECKPOINT
- Thực hiện các lệnh cập nhật khối dữ liệu lớn, ví dụ: BULK INSERT
- Thực hiện việc thêm hoặc bớt file dữ liệu bằng lệnh: ALTER DATABASE.
- Thay đổi mô hình recovery của CSDL về simple recovery model
- Dừng hoạt động của SQL Server bởi lệnh SHUTDOWN hoặc dừng dịch vụ.
- **SQL Server tự động phát các lệnh checkpoint**
- Việc backup CSDL được chấp nhận.
- Việc shutdown CSDL được thực hiện, ví dụ: thuộc tính AUTO_CLOSE is ON và the last user connection to the database is closed,
- Hoặc một thuộc tính của CSDL bị thay đổi và có yêu cầu khởi động lại CSDL.

Thiết lập tham số recovery interval

Tham số **recovery interval** tác động đến việc phát ra tự động các checkpoint của SQL Server.

- In Object Explorer, right-click a server and select **Properties**.
- Click the **Database settings** node.
- Under **Recovery**, in the **Recovery interval (minutes)** box, type or select a value from 0 through 32767 to set the maximum amount of time, in minutes, that SQL Server should spend recovering each database at startup.
- The default value is 0, indicating automatic configuration.

















Checkpoint tự động

- SQL Server luôn tạo ra checkpoint một cách tự động. Thời gian giữa các lần checkpoint phụ thuộc vào số bản ghi trong log file chứ không phải là theo một thời gian định trước. Thời gian giữa các lần checkpoint có thể rất dài, có thể rất ngắn phụ thuộc vào tần suất thay đổi dữ liệu.
- Khoảng cách giữa 2 lần checkpoint được tính theo tham số **recovery interval**. Tham số này chỉ định thời gian tối đa mà SQL Server sẽ sử dụng để khôi phục dữ liệu trong khi khởi động lại.
- SQL Server sẽ ước lượng số bản ghi mà nó có thể xử lý được trong khoảng thời gian bằng giá trị của tham số **recovery interval** thời gian (phút). Giá trị số bản ghi này sẽ làm cơ sở để SQL Server phát ra checkpoint.

CheckPoint tự động (2)

- Nếu CSDL sử dụng mô hình full recovery, thì checkpoint được tự động tạo ra nếu số bản ghi trong log file đạt bằng số bản ghi mà SQL server có thể xử lý được trong khoảng thời gian bằng giá trị của tham số recovery interval.
- Nếu CSDL sử dụng mô hình simple recovery, thì checkpoint sẽ được SQL Server tạo ra khi mà:
 - Log file đầy 70% hoặc
 - Số bản ghi trong log file đạt bằng số bản ghi mà SQL server có thể xử lý được trong khoảng thời gian bằng giá trị của tham số recovery interval.

Cấu trúc logic của DB

- [-]  DB 1
 - [+]  Database Diagrams
 - [+]  Tables
 - [+]  Views
 - [+]  Synonyms
 - [-]  Programmability
 - [+]  Stored Procedures
 - [+]  Functions
 - [+]  Database Triggers
 - [+]  Assemblies
 - [+]  Types
 - [+]  Rules
 - [+]  Defaults
 - [+]  Service Broker
 - [+]  Storage
 - [+]  Security

Tạo CSDL bằng T-SQL

USE master; GO

CREATE DATABASE MyDB

ON PRIMARY (NAME='MyDB_Primary', FILENAME='c:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\data\MyDB_Prm.mdf',
SIZE=4MB, MAXSIZE=10MB, FILEGROWTH=1MB),

FILEGROUP MyDB_FG1 (NAME = 'MyDB_FG1_Dat1', FILENAME =
'c:\Program Files\Microsoft SQL
Server\MSSQL.1\MSSQL\data\MyDB_FG1_1.ndf', SIZE = 1MB,
MAXSIZE=10MB, FILEGROWTH=1MB),

(NAME = 'MyDB_FG1_Dat2', FILENAME = 'c:\Program Files\Microsoft SQL
Server\MSSQL.1\MSSQL\data\MyDB_FG1_2.ndf', SIZE = 1MB,
MAXSIZE=10MB, FILEGROWTH=1MB)

LOG ON (NAME='MyDB_log', FILENAME = 'c:\Program Files\Microsoft SQL
Server\MSSQL.1\MSSQL\data\MyDB.ldf', SIZE=1MB, MAXSIZE=10MB,
FILEGROWTH=1MB); GO

ALTER DATABASE MyDB MODIFY FILEGROUP MyDB_FG1 DEFAULT:


Tạo CSDL bằng T-SQL(2)


```
USE MyDB;
```


```
CREATE TABLE MyTable ( cola int PRIMARY KEY, colb char(8) ) ON  
MyDB_FG1;
```



```
GO
```

Vị trí CSDL My_DB trên máy

**Database: MyDB**

Primary filegroup
c:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_Prm.mdf

4 MB

Log file
c:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB.ldf

1 MB

MyDB_FG1 filegroup
c:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_FG1_1.ndf

1 MB
c:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data\MyDB_FG1_2.ndf

1 MB

Cú pháp tạo, xóa CSDL bằng T-SQL

Tạo CSDL:

CREATE DATABASE *database_name*

[ON

{ [**PRIMARY**] (NAME = *logical_file_name*, FILENAME = '*os_file_name*'
[, SIZE = *size*] [, MAXSIZE = *max_size*] [, FILEGROWTH =
growth_increment]) } [, ...*n*]]

[**LOG ON**

{ (NAME = *logical_file_name*, FILENAME = '*os_file_name*' [, SIZE =
size] [, MAXSIZE = *max_size*] [, FILEGROWTH = *growth_increment*])
} [, ...*n*]

]

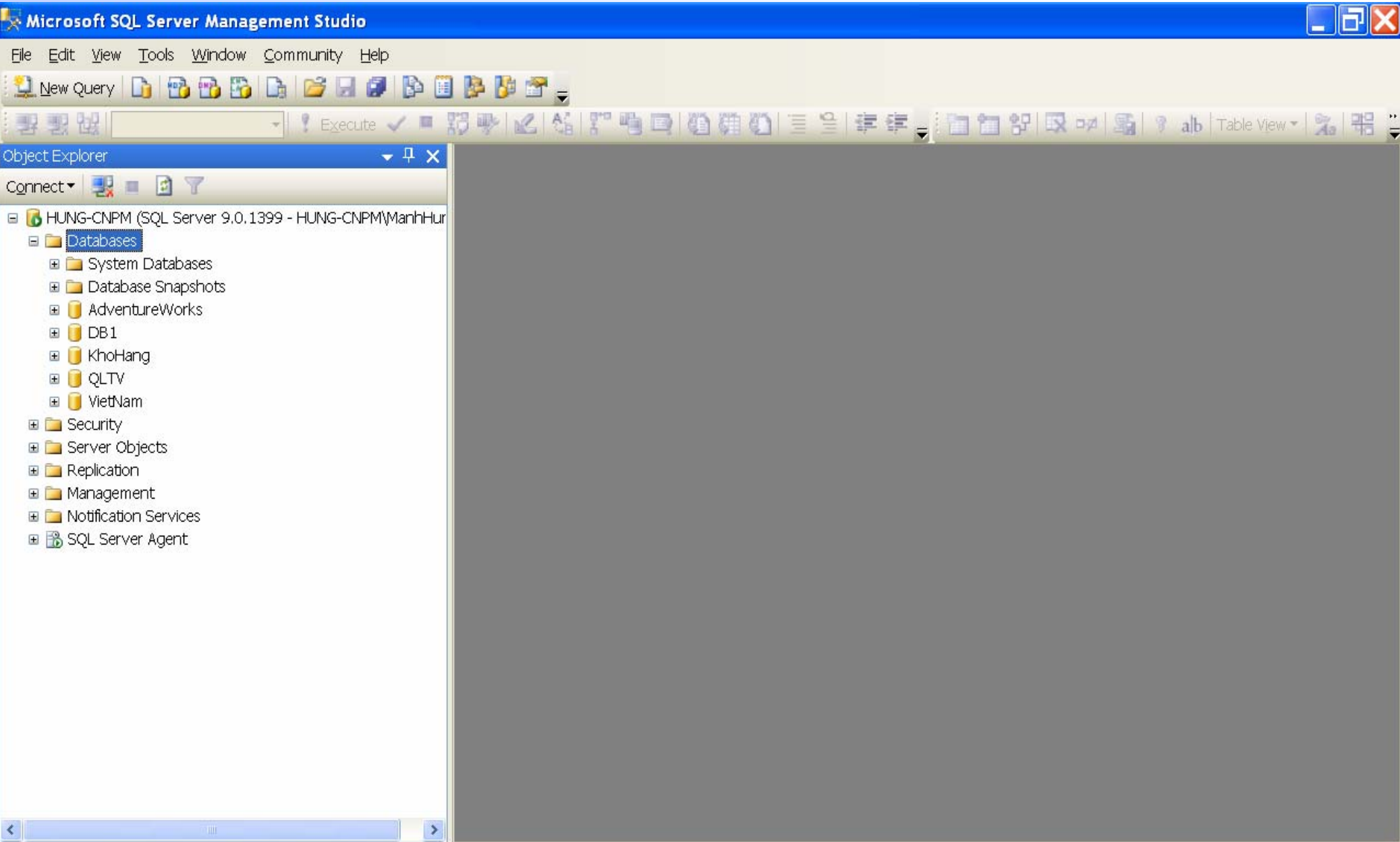
[**COLLATE** *collation_name*]

[< filegroup > ::= **FILEGROUP** *filegroup_name* < filespec > [,...*n*]]

Xóa CSDL:

DROP DATABASE *database_name*

Tạo DB bằng MS SQL Server Management



Lưu ý khi tạo DB

- Đối với các hệ thống nhỏ mà ở đó vấn đề tốc độ của server không thuộc loại nhạy cảm thì chúng ta thường chọn các giá trị mặc định (default) cho **Initial size, Automatically growth file**.
- Với database có kích thước lớn người ta không chọn Autogrowth(tự động tăng trưởng) và Autoshrink(tự động nén). Nguyên nhân là nếu chọn Autogrowth (hay Autoshrink) thì chúng ta có thể sẽ gặp 2 vấn đề sau:
 - **Performance hit:** Ảnh hưởng đáng kể đến khả năng làm việc của SQL Server. Do nó phải thường xuyên kiểm tra xem có đủ khoảng trống cần thiết hay không và nếu không đủ nó sẽ phải mở rộng bằng cách dành thêm khoảng trống từ đĩa cứng và chính quá trình này sẽ làm chậm đi hoạt động của SQL Server.
 - **Disk fragmentation :** Việc mở rộng trên cũng sẽ làm cho data không được liên tục mà chứa ở nhiều nơi khác nhau trong đĩa cứng điều này cũng gây ảnh hưởng lên tốc độ làm việc của SQL Server.

COLLATION

Collation – Là bộ qui tắc sắp xếp dữ liệu.

Dữ liệu không phải unicode: bảng mã, bảng qui tắc sắp xếp dữ liệu (collation)

Dữ liệu Unicode: collation

Như chúng ta đã biết: để lưu trữ các chữ cái, chữ số, Trên máy tính người ta sử dụng các bit 0,1.

Bảng mã ASCII sử dụng 1 byte -> mã được 256 ký hiệu

Bảng mã UNICODE 2 byte -> 65 536 ký hiệu

COLLATION (2)

CI – (case-insensitive), CS (case sensitive)

AI –(accent-insensitive), AS (accent-sensitive)

VD: - SQL_Latin1_General_Cp1_CS_AS.

VD: use tempDb

go

create table bang1 (id int primary key, col1 varchar (1) collate
SQL_Latin1_General_Cp1_CI_AS)

ID	Col1
1	a
2	A
3	b
4	B

COLLATION (3)

SELECT * FROM bang1 ORDER BY col1

ID	Col1
1	a
2	A
3	b
4	B

SELECT * FROM bang1 ORDER BY col1 COLLATE
SQL_Latin1_General_Cp1_CS_AS

ID	Col1
2	A
1	a
4	B
3	b

Khi nào cần đến FILEGROUP

- Sử dụng FILEGROUP cho phép chúng ta lưu trữ các bảng phân tán trên nhiều FILEGROUP trên các đĩa vật lý khác nhau
- FILEGROUP chia thành:
 - Chính Primary: Luôn được tạo ra tự động, lưu các bảng hệ thống
 - Filegroup do người sử dụng tạo
- Filegroup mặc định: **Primary**

Bảo trì CSDL

- Kiểm tra các thông số của database
- Sửa đổi kích thước của file data, log
- Thêm file group
- Thêm, sửa, xóa file chỉ số
- Thêm, sửa, xóa quan hệ giữa các bảng

Bảo trì CSDL (2)

- Mở rộng kích thước

```
ALTER DATABASE Products
```

```
MODIFY FILE ( NAME = 'Prods', SIZE = 20MB)
```

- Thêm một file dữ liệu vào file group PRIMARY

```
ALTER DATABASE Products
```

```
ADD FILE (NAME = 'Prods2' ,
```

```
FILENAME='c:\sqldata\prods2.ndf', SIZE=10MB ,  
MAXSIZE=20MB)
```

- Mở rộng kích thước Log file

```
ALTER DATABASE Products
```

```
MODIFY FILE ( NAME = ProdsLog', SIZE = 10MB)
```

Bảo trì CSDL(3)

- Thêm file group

```
ALTER DATABASE Products
```

```
ADD FILEGROUP ProdGroup1
```

```
GO
```

```
ALTER DATABASE Products
```

```
ADD FILE ( NAME = 'groupData1', FILENAME =  
    'D:\mssql\data\groupData1.ndf', SIZE = 5MB)
```

```
TO FILEGROUP ProdGroup1
```

```
GO
```


Cú pháp sửa đổi CSDL

ALTER DATABASE *database*

```
{ ADD FILE < filespec > [ ,...n ] [ TO FILEGROUP  
filegroup_name ]  
| ADD LOG FILE < filespec > [ ,...n ]  
| REMOVE FILE logical_file_name  
| ADD FILEGROUP filegroup_name  
| REMOVE FILEGROUP filegroup_name  
| MODIFY FILE < filespec >  
| MODIFY NAME = new_dbname  
| MODIFY FILEGROUP filegroup_name  
| SET < optionspec > [ ,...n ]  
| COLLATE < collation_name >  
}
```

Cú pháp sửa đổi CSDL(2)

< optionspec >: Gồm

- SINGLE_USER | MULTI_USER

VD:use master

go

alter database pubs set multi_user

go

- RECOVERY FULL | SIMPLE

VD:use master

go

ALTER DATABASE Products SET RECOVERY FULL

Database Consistency Checker – DBCC

Database Consistency Checker – DBCC: Bộ kiểm tra tính nhất quán của CSDL.

- CHECK
 - DBCC CHECKDB
 - DBCC CHECKTABLE
 - DBCC CHECKALLOC
 - DBCC CHECKFILEGROUP
 - DBCC CHECKCONSTRAINTS
- SHRINK
 - DBCC SHRINKDATABASE
 - DBCC SHRINKFILE
- DBCC CLEAN
- DBCC SQLPERF
- DBCC DBREINDEX

DBCC (2)

- Dùng DBCC với tham số CHECKCONSTRAINTS để phát hiện các lỗi vi phạm ràng buộc của các bản ghi trong các bảng của CSDL.
- Xem Example1, Example2 (lecture1-dbcc.doc)

DBCC (3)

Cú pháp để thu nhỏ CSDL:

DBCC SHRINKDATABASE

```
( 'database_name' | database_id | 0    [ ,target_percent ]    [ , {  
    NOTRUNCATE | TRUNCATEONLY } ]  
 ) [ WITH NO_INFOMSGS ]
```

VD: Thu nhỏ CSDL UserDB chỉ để 10% không gian còn trống.

DBCC SHRINKDATABASE (UserDB, 10)

DBCC (4)

- DBCC SQLPERF(LOGSPACE) - Provides statistics about how the transaction-log space was used in all databases.

Column name	Definition
Database Name	Name of the database for the log statistics displayed.
Log Size (MB)	Actual amount of space available for the log. This amount is smaller than the amount originally allocated for log space because the SQL Server 2005 Database Engine reserves a small amount of disk space for internal header information.
Log Space Used (%)	Percentage of the log file currently occupied with transaction log information.
Status	Status of the log file. Always 0.

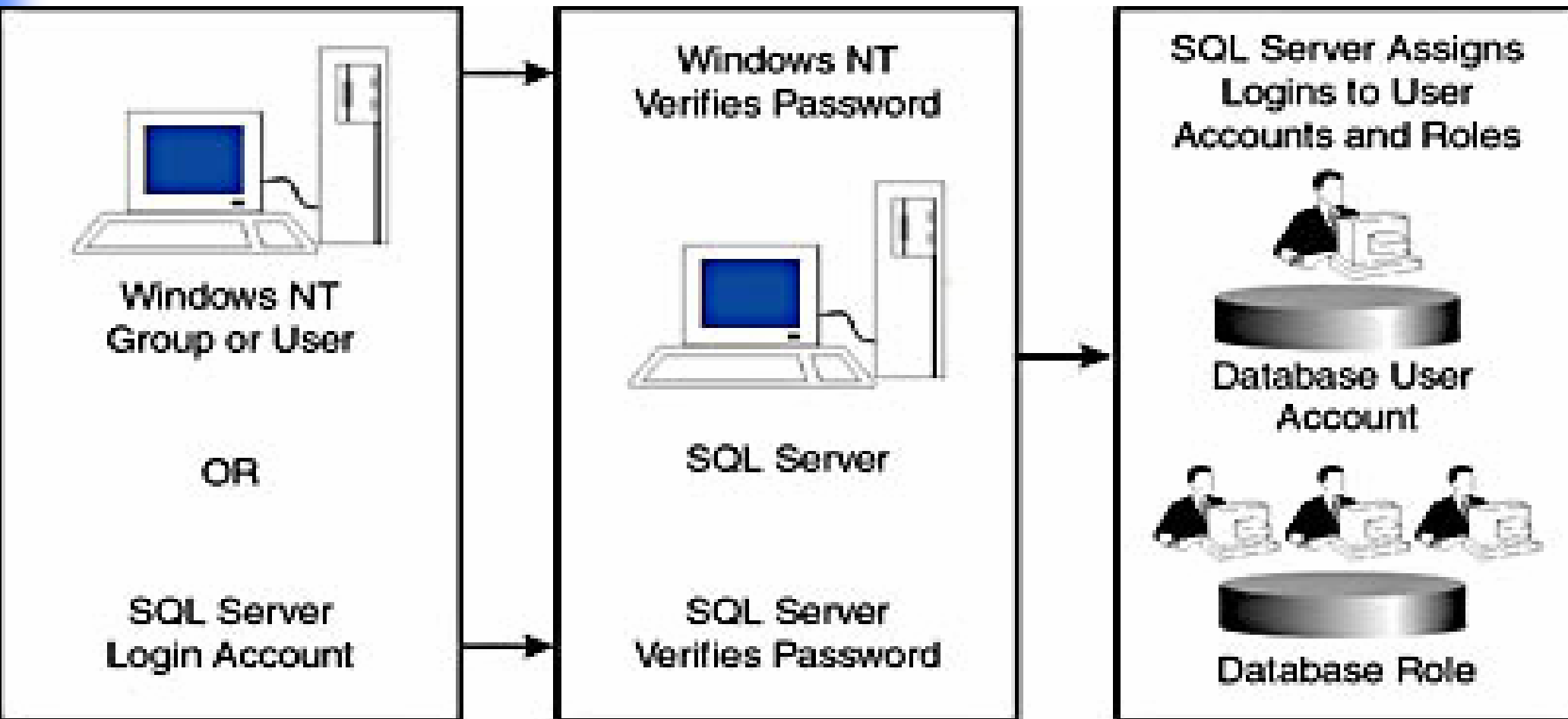
DBCC (5)

- DBCC CLEAN TABLE: Đòi lại không gian của những trường có độ dài thay đổi và trường text, đã bị xóa (Reclaims space for dropped variable length columns and text columns).
 - DBCC cleantable (Dbtest1, Nhanvien)
- DBCC DBREINDEX: Rebuilds one or more indexes for a table in the specified database.
 - DBCC DBREINDEX
('table_name' [, 'index_name' [, *fillfactor*]])
USE AdventureWorks;
GO
DBCC DBREINDEX ('HumanResources.Employee',
PK_Employee_EmployeeID,80);

Một số thủ tục hệ thống

Tên thủ tục	Ứng dụng
sp_help ['object']	Cung cấp thông tin về một database object (table, view...) hay một data type.
sp_helpdb ['database']	Cung cấp thông tin về một database cụ thể nào đó.
sp_monitor	Cho biết mức độ bận rộn của SQL Server
Sp_spaceused ['object']	Cung cấp thông tin về các khoảng trống đã được sử dụng cho một object nào đó.=> có thể sử dụng để xem kích thước của các đối tượng: Bảng,...
sp_who ['login']	Cho biết thông tin về một SQL Server user

Mô hình xác thực user



Mô hình truy cập dữ liệu

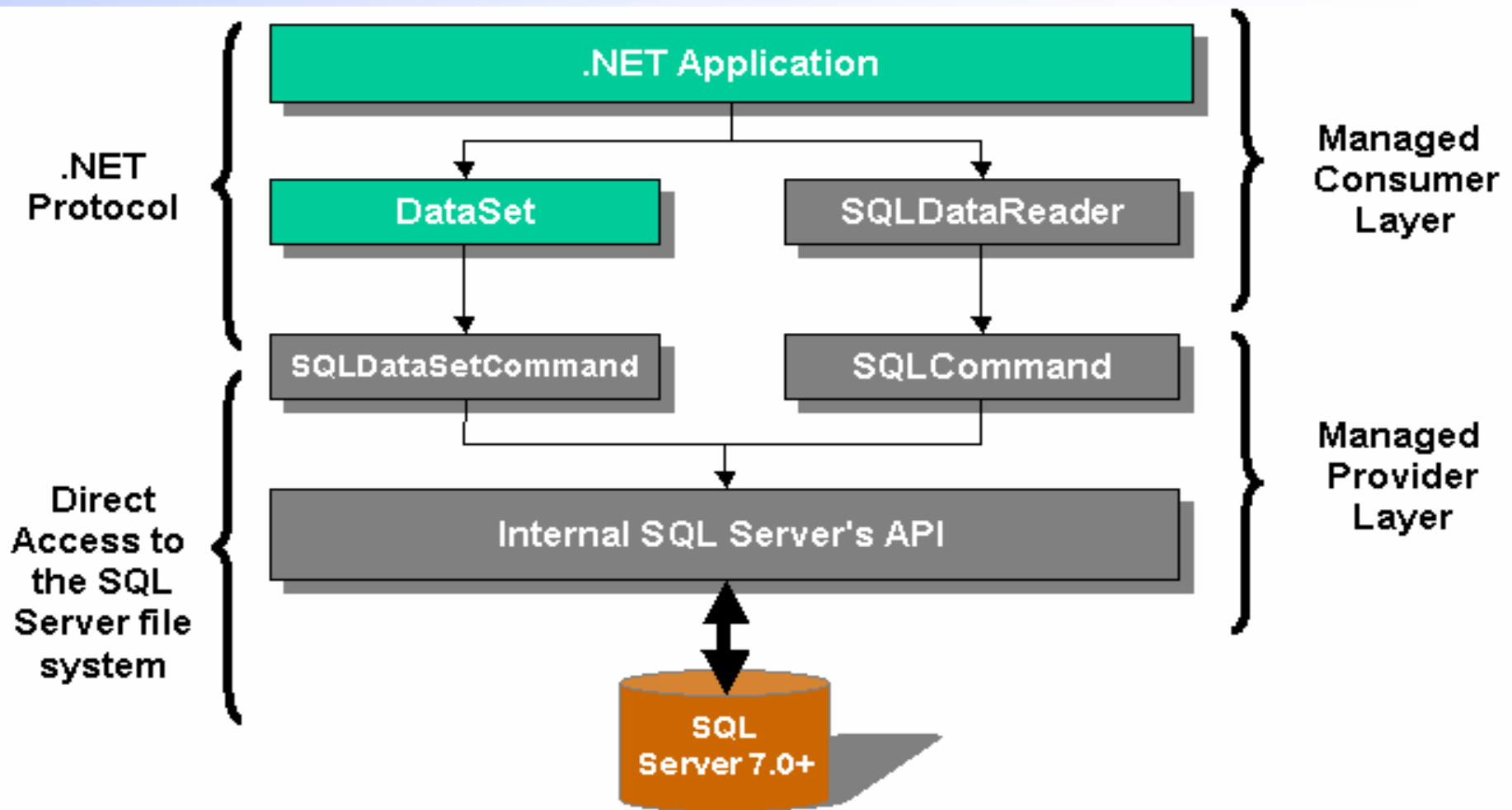


Diagram for the SQL Server managed provider

Here's the code for SQL Server

```
Dim strConn, strCmd As String
```

```
strConn = "DATABASE=Northwind;SERVER=localhost;Integrated  
Security=SSPI;"
```

```
strCmd = "SELECT * FROM Employees"
```

```
Dim oCMD As New SQLDataSetCommand(strCmd, strConn)
```

```
Dim oDS As New DataSet
```

```
oCMD.FillDataSet(oDS, "EmployeesList")
```

Mô hình truy cập dữ liệu(2)

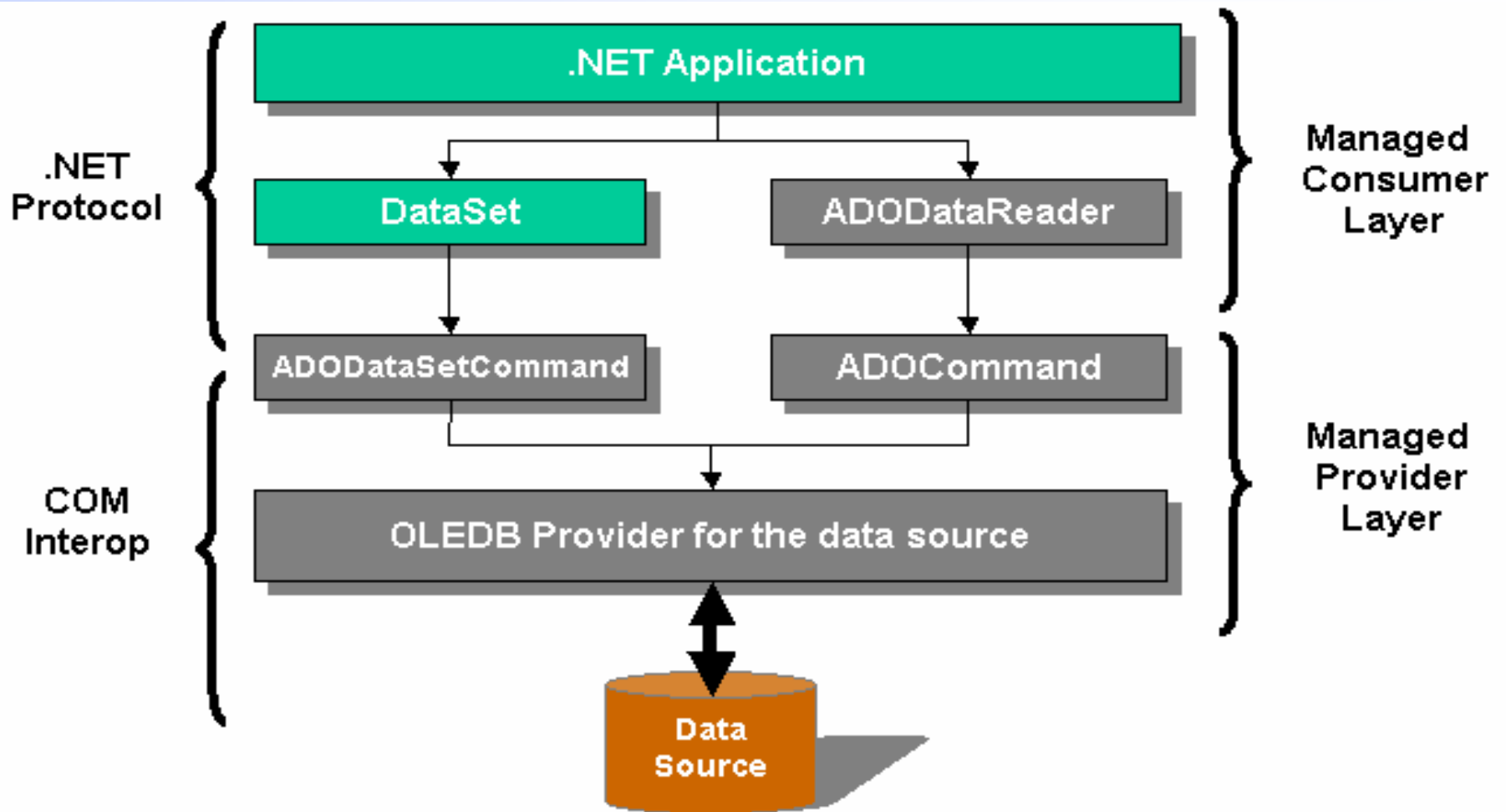


Diagram for the OLE DB managed provider

Here's the code for the OLE DB provider

```
Dim strConn, strCmd As String
```

```
strConn = "Provider=SQLOLEDB;"
```

```
strConn +=
```

```
    "DATABASE=Northwind;SERVER=localhost;Integrated  
    Security=SSPI;"
```

```
strCmd = "SELECT * FROM Employees"
```

```
Dim oCMD As New ADODatasetCommand(strCmd,  
    strConn)
```

```
Dim oDS As New DataSet
```

```
oCMD.FillDataSet(oDS, "EmployeesList")
```

Tạo các partitioned table

Các bước:

- **CREATE PARTITION FUNCTION**

```
CREATE PARTITION FUNCTION partition_function_name  
    ( input_parameter_type ) AS RANGE [ LEFT | RIGHT ]  
FOR VALUES ( [ boundary_value [ ,...n ] ] ) [ ; ]
```

- **CREATE PARTITION SCHEME**

```
CREATE PARTITION SCHEME partition_scheme_name  
AS PARTITION partition_function_name  
[ ALL ] TO ( { file_group_name | [ PRIMARY ] } [ ,...n ] )  
[ ; ]
```

Tạo các partitioned table (2)

use AdventureWorks

```
ALTER DATABASE AdventureWorks ADD FILEGROUP test1fg go
```

```
ALTER DATABASE AdventureWorks ADD FILEGROUP test2fg go
```

```
ALTER DATABASE AdventureWorks ADD FILEGROUP test3fg go
```

```
ALTER DATABASE AdventureWorks ADD FILEGROUP test4fg go
```

```
CREATE PARTITION FUNCTION myRangePF1 (int)
```

```
AS RANGE LEFT FOR VALUES (1, 100, 1000) ; GO
```

```
CREATE PARTITION SCHEME myRangePS1 AS PARTITION  
myRangePF1
```

```
TO (test1fg, test2fg, test3fg, test4fg) ; GO
```

```
CREATE TABLE PartitionTable (col1 int, col2 char(10))
```

```
ON myRangePS1 (col1) ;
```

Tạo các partitioned table (3)

- -- clear
- Drop table PartitionTable go
- DROP PARTITION SCHEME myRangePS1 go
- drop PARTITION FUNCTION myRangePF1 go
- ALTER DATABASE AdventureWorks
- REMOVE FILEGROUP test1fg go
- ALTER DATABASE AdventureWorks
- REMOVE FILEGROUP test2fg go
- ALTER DATABASE AdventureWorks
- REMOVE FILEGROUP test3fg go
- ALTER DATABASE AdventureWorks
- REMOVE FILEGROUP test4fg

(Xem thêm trong lecture1.doc)

Select dữ liệu từ bảng với từ khóa WITH

- Giả sử ta có bảng Person với các trường như sau:
 - ID kiểu int là mã của người;
 - Name kiểu nvarchar(30) là tên người;
 - Mother kiểu int và Father kiểu int là mã cha, mẹ của người.
- Bài toán đặt ra như sau:
 - Biết tên của một người nào đó.
 - Hãy hiển thị tất cả các tiền bối của người này.

Select dữ liệu từ bảng với từ khóa WITH (2)

- Tạo bảng:

```
CREATE TABLE Person(ID int, Name nvarchar(30), Mother int, Father int);
```

- Thêm dữ liệu:

```
INSERT Person VALUES(1, 'Sue', NULL, NULL);
```

```
INSERT Person VALUES(2, 'Ed', NULL, NULL);
```

```
INSERT Person VALUES(3, 'Emma', 1, 2);
```

```
INSERT Person VALUES(4, 'Jack', 1, 2);
```

```
INSERT Person VALUES(5, 'Jane', NULL, NULL);
```

```
INSERT Person VALUES(6, 'Bonnie', 5, 4);
```

```
INSERT Person VALUES(7, 'Bill', 5, 4);
```

Select dữ liệu từ bảng với từ khóa WITH (3)

- Tạo một lệnh Select đệ qui để tìm tất cả các tiền bối của Bonnie.

WITH **Generation (ID)** **AS** (*-- First: Tìm Mother của Bonnie.*

SELECT **Mother** **FROM** **Person** **WHERE** **Name** = 'Bonnie'

UNION – *Second: Tìm Father của Bonnie.*

SELECT **Father** **FROM** **Person** **WHERE** **Name** = 'Bonnie'

UNION ALL *-- First recursive member returns male ancestors of the previous generation.*

SELECT **Person.Father** **FROM** **Generation, Person** **WHERE**
Generation.ID=Person.ID

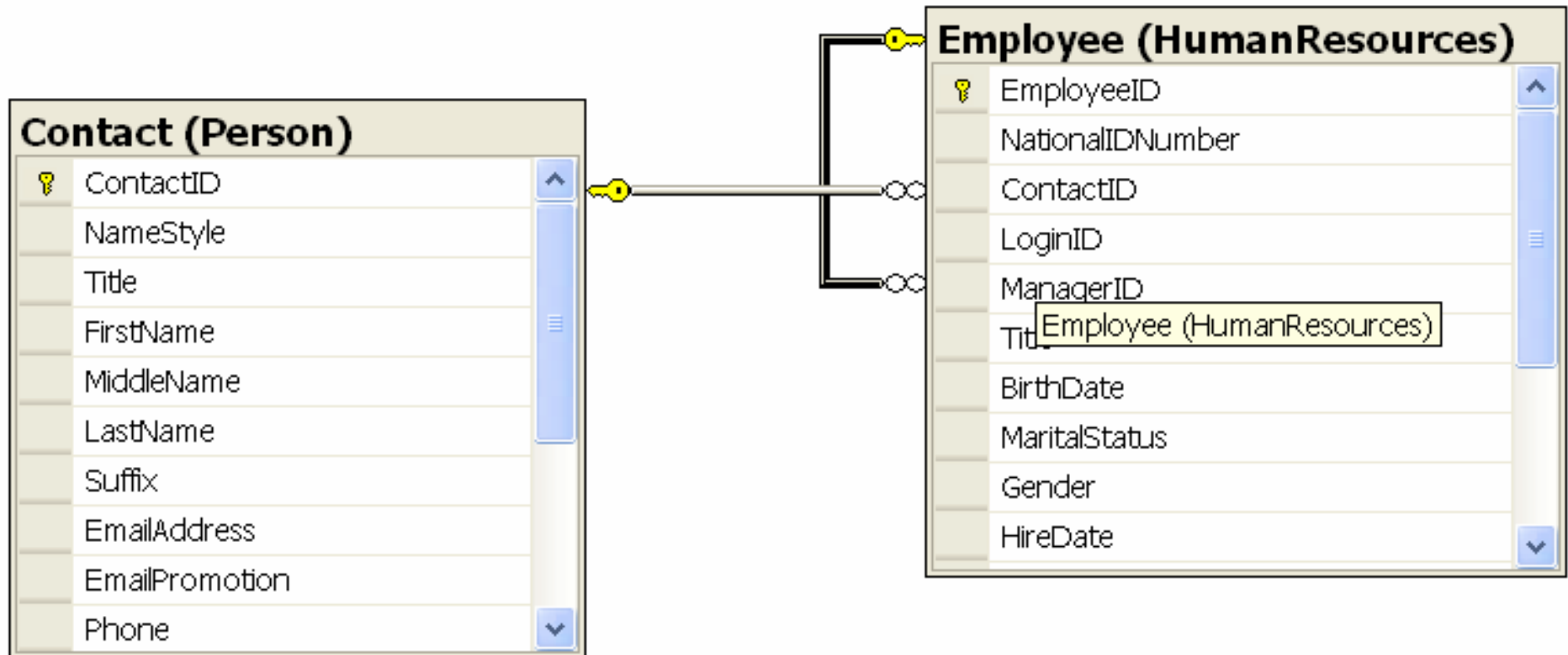
UNION ALL *-- Second recursive member returns female ancestors of the previous generation.*

SELECT **Person.Mother** **FROM** **Generation, Person** **WHERE**
Generation.ID=Person.ID)

SELECT **Person.ID, Person.Name, Person.Mother, Person.Father**
FROM **Generation, Person** **WHERE** **Generation.ID = Person.ID;**

Select dữ liệu từ bảng với từ khóa WITH (4)

- Trong CSDL AdventureWorks có hai bảng dữ liệu **HumanResources.Employee** và **Person.Contact**



Select dữ liệu từ bảng với từ khóa WITH (5)

- Bài toán đặt ra như sau: hiển thị toàn bộ nhân viên trong doanh nghiệp theo qui tắc tìm kiếm theo chiều sâu, bắt đầu từ người có chức vụ cao nhất.

USE AdventureWorks;

go

WITH DirectReports (**Name**, Title, EmployeeID, **EmployeeLevel**, **Sort**) AS

(-- Lấy tất cả nhân viên cao nhất

SELECT **CONVERT**(varchar(255), c.FirstName + ' ' + c.LastName), e.Title,
e.EmployeeID, 1, **CONVERT**(varchar(255), c.FirstName + ' ' + c.LastName)

FROM **HumanResources.Employee** AS e JOIN **Person.Contact** AS c

ON e.ContactID = c.ContactID WHERE **e.ManagerID IS NULL**

-- Để qui để lấy các nhân viên thấp hơn

UNION ALL SELECT

CONVERT(varchar(255), **REPLICATE** ('| ',EmployeeLevel) + c.FirstName + ' ' +
c.LastName), e.Title, e.EmployeeID, EmployeeLevel + 1,

CONVERT (varchar(255), RTRIM(Sort) + '|' + FirstName + ' ' + LastName)

FROM HumanResources.Employee as e

JOIN Person.Contact AS c ON e.ContactID = c.ContactID

JOIN DirectReports AS d ON e.ManagerID = d.EmployeeID

)

SELECT EmployeeID, Name, Title, EmployeeLevel, Sort FROM DirectReports
ORDER BY Sort;