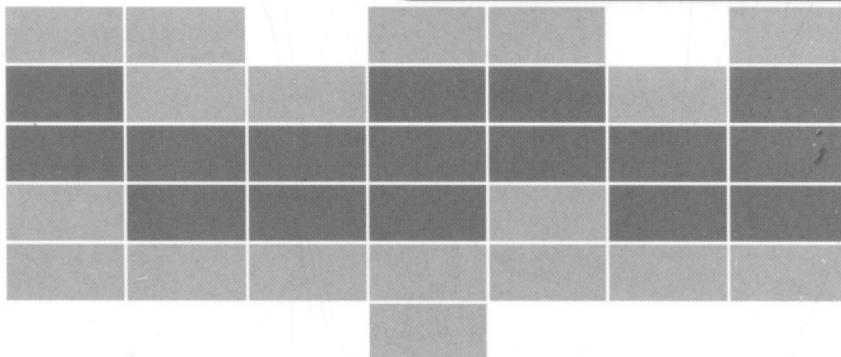




PHẠM HỮU KHANG (*Chủ biên*)
ĐOÀN THIỆN NGÂN (*Hiệu đính*)

SQL Server

Lập trình 2005
T-SQL



NHÀ XUẤT BẢN LAO ĐỘNG - XÃ HỘI

GIỚI THIỆU

3 

GIỚI THIỆU

SQL Server 2005 cung cấp nhiều tính năng mới giúp cho bạn dễ dàng xây dựng cơ sở dữ liệu trên nền Windows phục vụ cho quản lý hay kinh doanh thương mại điện tử. SQL Server 2005 giới thiệu các dịch vụ mới giúp cho nhà quản trị cơ sở dữ liệu hay lập trình ứng dụng có thể phát triển hệ cơ sở dữ liệu quan hệ để phục vụ cho việc lưu trữ, kết hợp, phân tán, liên kết, trao đổi dữ liệu, tìm kiếm hay xuất dữ liệu theo định dạng mà nhà quản lý yêu cầu.

Cuốn sách cung cấp cho bạn những kiến thức cơ bản liên quan đến việc xây dựng cơ sở dữ liệu thực tế và sử dụng phát biểu SQL dạng DDL để định nghĩa đối tượng cơ sở dữ liệu, DML dùng để thao tác dữ liệu với chức năng cơ bản là thêm, xóa, cập nhật và trình bày.

Đặc biệt giáo trình nhấn mạnh những tiện ích, dịch vụ mới, phát biểu SQL, mệnh đề và từ khóa đặc biệt, kiểu dữ liệu tiện dụng mà phiên bản SQL Server 2000 chưa có như: Kiểu dữ liệu Xml, char(max), varchar(max), đa dạng hóa mệnh đề TOP, OUTPUT, WRITE, WITH, ORDER và OVER, biểu thức bảng.

Giáo trình bao gồm 7 chương lý thuyết và 1 chương ứng dụng, xuyên suốt từ giới thiệu ngôn ngữ lập trình T-SQL trong SQL Server 2005, Management Studio, phát biểu SQL đơn giản đến nâng cao giúp cho bạn sử dụng các phát biểu này trong ứng dụng thực tế và nhiều kỹ thuật quan trọng khác cùng với nhiều ví dụ chi tiết, diễn giải rõ ràng.

Cuốn sách “SQL Server 2005-Lập trình T-SQL” nằm trong bộ giáo trình SQL Server 2005 bao gồm nhiều cuốn từ lập trình T-SQL, lập trình thủ tục và hàm, lập trình nâng cao, ứng dụng SQL Server 2005 trong hệ thống kế toán và quản trị cơ sở dữ liệu SQL Server 2005.

MK.PUB

HƯỚNG DẪN SỬ DỤNG VÍ DỤ GIÁO TRÌNH SQL SERVER 2005 LẬP TRÌNH T-SQL

Để sử dụng các ví dụ đính kèm theo sách, trước tiên bạn chép hai tập tin AccountSystem.mdf và AccountSystem.ldf vào ổ đĩa cứng, sử dụng Management Studio để tạo cơ sở dữ liệu AccountSystem dạng Attach bằng cách thực hiện như sau:

- Chọn ngăn Databases và R-Click.
- Chọn Attach... rồi chọn nút Add trong cửa sổ vừa xuất hiện.
- Chọn tập tin cơ sở dữ liệu AccountSystem.mdf và nhấn nút OK.
- Kiểm tra đường dẫn cơ sở dữ liệu sẽ lưu trữ và nhấn nút OK.

Nếu muốn phục hồi cơ sở dữ liệu từ tập tin .bak, bạn có thể thực hiện các bước như sau:

- Chọn ngăn Databases và R-Click.
- Chọn Restore Database rồi đặt tên AccountSystem trong phần “To database”.
- Chọn tùy chọn From device rồi trỏ đến tập tin cơ sở dữ liệu bạn backup là AccountSystem.bak và nhấn nút OK.
- Kiểm tra đường dẫn cơ sở dữ liệu sẽ lưu và nhấn nút OK.

Sau đó, bạn chép thư mục Projects vào ổ đĩa cứng, trong mỗi cuốn sách có nhiều dự án ứng với các phần trong hệ thống kế toán và ví dụ tham khảo cho cuốn sách được quản lý bằng Management Studio. Để mở giải pháp (Solution) trong Management Studio, bạn chọn vào File | Open Project | Solution và chọn vào tập tin AccountSystemSoln.

Ví dụ đính kèm theo sách được tổ chức theo từng Project, bao gồm nhiều Project ứng với chức năng trong hệ thống kế toán, sau khi mở Solution trong Management Studio, bạn có thể chọn từng Project để thực thi phát biểu SQL bằng cách khởi động cửa sổ Query hoặc chọn tên tập tin .sql rồi Double Click.

THƯ NGỎ

5 

THƯ NGỎ

Kính thưa quý Bạn đọc gần xa!

Trước hết, Ban xuất bản xin bày tỏ lòng biết ơn và niềm vinh hạnh được đồng đảo Bạn đọc nhiệt tình ủng hộ tủ sách MK.PUB.

Trong thời gian qua chúng tôi rất vui và cảm ơn các Bạn đã gửi e-mail đóng góp nhiều ý kiến quý báu cho tủ sách.

Mục tiêu và phương châm phục vụ của chúng tôi là:

- *Lao động khoa học nghiêm túc.*
- *Chất lượng và ngày càng chất lượng hơn.*
- *Tất cả vì Bạn đọc.*

Một lần nữa, Ban xuất bản MK.PUB xin kính mời quý Bạn đọc tiếp tục tham gia cùng chúng tôi để nâng cao chất lượng sách. Cụ thể:

Trong quá trình sử dụng sách, nếu quý Bạn phát hiện thấy bất kỳ sai sót nào (dù nhỏ) xin đánh dấu, ghi chú nhận xét ý kiến của Bạn ra bên cạnh rồi gửi cuốn sách này cho chúng tôi theo địa chỉ:

Nhà sách Minh Khai

249 Nguyễn Thị Minh Khai, Q.1, Tp. Hồ Chí Minh.

E-mail: mk.book@minhkhai.com.vn hoặc mk.pub@minhkhai.com.vn

Chúng tôi xin hoàn lại cước phí bưu điện và gửi trả lại Bạn cuốn sách cùng tên. Ngoài ra chúng tôi còn gửi tặng Bạn một cuốn sách khác trong tủ sách MK.PUB. Bạn có thể chọn cuốn sách này theo danh mục thích hợp sẽ gửi tới Bạn.

Với mục đích ngày càng nâng cao chất lượng tủ sách MK.PUB, chúng tôi rất mong nhận được sự hợp tác nhiệt tình của quý Bạn đọc gần xa.

"MK.PUB cùng Bạn đọc đồng hành" để nâng cao chất lượng sách.

Một lần nữa chúng tôi xin chân thành cảm ơn.

MK.PUB

MỤC LỤC

7

MỤC LỤC

GIỚI THIỆU.....	3
THƯ NGỎ	5
MỤC LỤC	7
Chương 1: GIỚI THIỆU CƠ SỞ DỮ LIỆU SQL SERVER 2005	15
1. Khái niệm cơ sở dữ liệu	15
2. Chuẩn hóa dữ liệu	17
2.1. <i>First Normal Form (1NF)</i>	19
2.2. <i>Second Normal Form</i>	20
2.3. <i>Third Normal Form</i>	21
3. Giới thiệu SQL Server 2005	23
3.1. <i>Phân quản trị</i>	23
3.2. <i>Phân lập trình</i>	23
3.3. <i>Tiện ích khác</i>	24
4. cài đặt SQL Server 2005	24
5. Kết chương	34
Chương 2: TÌM HIỂU CƠ SỞ DỮ LIỆU ĐÍNH KÈM.....	35
1. Giới thiệu cơ sở dữ liệu AccountSystem	35
2. Cấu trúc cơ sở dữ liệu phần General Ledger.....	37
2.1. <i>Phiếu thu</i>	37
2.2. <i>Phiếu chi</i>	40
2.3. <i>Tồn quỹ</i>	42

<i>2.4. Danh sách tài khoản</i>	45
<i>2.5. Quan hệ.....</i>	47
<i>3. Cấu trúc cơ sở dữ liệu phần Account Receivable.....</i>	47
<i> 3.1. Khách hàng.....</i>	48
<i> 3.2. Hóa đơn bán hàng.....</i>	51
<i> 3.3. Quan hệ.....</i>	54
<i>4. Cấu trúc cơ sở dữ liệu phần Account Payable</i>	55
<i> 4.1. Nhà cung cấp.....</i>	56
<i> 4.2. Hóa đơn mua hàng.....</i>	59
<i> 4.3. Quan hệ.....</i>	61
<i>5. Cấu trúc cơ sở dữ liệu phần Inventory Control</i>	62
<i> 5.1. Phần tồn kho</i>	63
<i> 5.2. Phần xuất hàng</i>	67
<i> 5.3. Phần nhập hàng</i>	69
<i>6. Cấu trúc cơ sở dữ liệu phần dùng chung</i>	72
<i> 6.1. Bảng Countries.....</i>	72
<i> 6.2. Bảng Provinces</i>	73
<i> 6.3. Bảng Banks.....</i>	73
<i> 6.4. Bảng BankOfCustomers</i>	73
<i> 6.5. Bảng BankOfSuppliers.....</i>	74
<i> 6.6. Quan hệ.....</i>	74
<i>7. Kết chương</i>	75
Chương 3: LÀM VIỆC VỚI MANAGEMENT STUDIO	77
<i>1. Giao diện SQL Server Management Studio.....</i>	78

MỤC LỤC9 

2. Khám phá cửa sổ Object Explorer.....	81
3. Đăng ký SQL Server.....	84
4. Quản lý dự án, tập tin trong MS.....	89
5. Làm việc với Visual SourceSafe 2005	93
6. Các tiện ích khác	98
6.1. <i>Tiện ích SQL Server Profiler</i>	98
6.2. <i>Tiện ích Database Engine Tuning Advisor</i>	101
7. Kết chương	104
Chương 4: THÀNH PHẦN CHÍNH TRONG CƠ SỞ DỮ LIỆU ..	105
1. Kiểu dữ liệu trong SQL Server 2005.....	106
1.1. Nhóm kiểu dữ liệu System Data Types	107
1.2. Nhóm kiểu dữ liệu User-Defined Data Types	116
2. Đối tượng Table.....	118
3. Khai báo chỉ mục	121
4. Tạo quan hệ cơ sở dữ liệu bằng đối tượng Diagram	122
5. Đối tượng View.....	127
6. Khai báo ràng buộc dữ liệu	131
7. Kết chương	134
Chương 5: PHÁT BIỂU T-SQL DẠNG ĐỊNH NGHĨA DỮ LIỆU ..	135
1. Phát biểu CREATE	135
1.1. Phát biểu CREATE DATABASE.....	136
1.2. Phát biểu CREATE TABLE.....	143
1.3. Phát biểu CREATE VIEW	153
2. Phát biểu ALTER.....	161

M[®] 10**MỤC LỤC**

2.1. Phát biểu ALTER DATABASE.....	161
2.2. Phát biểu ALTER TABLE	164
2.3. Phát biểu ALTER VIEW.....	166
3. Phát biểu DROP.....	168
3.1. Phát biểu DROP DATABASE.....	168
3.2. Phát biểu DROP TABLE	169
3.3. Phát biểu DROP VIEW.....	169
4. Kết chương	170

**Chương 6: PHÁT BIỂU T-SQL CƠ BẢN DẠNG TRUY VẤN
DỮ LIỆU** 171

1. Phát biểu SELECT	172
1.1. Phát biểu SELECT đơn giản	172
1.2. Biểu thức trong phát biểu SELECT	175
2. Mệnh đề ORDER BY	176
2.1. Phát biểu SELECT với mệnh đề ORDER BY.....	176
3. Từ khóa TOP	178
3.1. Từ khóa TOP với số cụ thể	179
3.2. Từ khóa TOP với số phần trăm	179
3.3. Từ khóa TOP với WITH TIES.....	182
4. Từ khóa DISTINCT	183
5. Phát biểu SELECT và mệnh đề INTO.....	186
5.1. Mệnh đề INTO và bảng dữ liệu tạm	186
5.2. Mệnh đề INTO và bảng trong cơ sở dữ liệu.....	186
6. Phát biểu SELECT và từ khóa AS	188

MỤC LỤC

11

7. Phát biểu SELECT và mệnh đề WHERE	191
7.1. Mệnh đề WHERE và phép toán LIKE.....	191
7.2. Mệnh đề WHERE và phép toán AND	191
7.3. Mệnh đề WHERE và phép toán IS NULL	192
7.4. Mệnh đề WHERE và phép toán IN.....	193
7.5. Mệnh đề WHERE và phép toán BETWEEN.....	193
7.6. Mệnh đề WHERE và phép toán khác.....	195
8. Phát biểu SELECT với mệnh đề GROUP BY.....	196
8.1. Sử dụng hàm MAX, MIN, COUNT, SUM, AVG.....	196
8.2. Sử dụng phép toán ROLLUP.....	197
8.3. Sử dụng phép toán CUBE.....	199
8.4. Sử dụng hàm GROUPING.....	201
8.5. Phát biểu SELECT với mệnh đề HAVING	204
9. Phát biểu SELECT với mệnh đề COMPUTE	206
9.1. Mệnh đề COMPUTE	206
9.2. Phát biểu SELECT với mệnh đề COMPUTE BY	208
10. Phát biểu SELECT với mệnh đề JOIN	209
10.1. Mệnh đề INNER JOIN	210
10.2. Mệnh đề LEFT JOIN.....	215
10.3. Mệnh đề RIGHT JOIN	218
10.4. Mệnh đề FULL JOIN.....	220
11. Phép toán UNION, EXCEPT, INTERSECT	221
11.1. Phát biểu SELECT với phép toán UNION	221
11.2. Phát biểu SELECT với phép toán EXCEPT	225

11.3. Phát biểu SELECT với phép toán INTERSECT	226
12. Phân trang với phát biểu WITH	227
13. Lấy tập mẫu tin ngẫu nhiên.....	233
14. Kết chương	234
Chương 7: PHÁT BIỂU T-SQL CƠ BẢN DẠNG XỬ LÝ DỮ LIỆU	235
1. Phát biểu INSERT	235
1.1. Phát biểu INSERT cơ bản	236
1.2. Phát biểu INSERT với cột số tự động.....	238
1.3. Phát biểu INSERT từ bảng dữ liệu.....	239
1.4. Phát biểu INSERT với đối tượng View	240
1.5. Phát biểu INSERT với mệnh đề TOP	241
1.6. Phát biểu INSERT với phát biểu EXECUTE	242
1.7. Phát biểu INSERT với phát biểu WITH	244
1.8. Phát biểu INSERT trên nhiều cơ sở dữ liệu.....	245
1.9. Phát biểu INSERT với mệnh đề OUTPUT.....	245
2. Phát biểu UPDATE.....	250
2.1. Phát biểu UPDATE đơn giản	251
2.2. Phát biểu UPDATE với phát biểu SELECT	252
2.3. Phát biểu UPDATE với mệnh đề TOP	252
2.4. Phát biểu UPDATE với mệnh đề OUTPUT.....	252
2.5. Phát biểu UPDATE với mệnh đề WITH	255
2.6. Phát biểu UPDATE với mệnh đề WRITE	256
3. Phát biểu DELETE.....	259
3.1. Phát biểu DELETE đơn giản.....	260

MỤC LỤC13 

3.2. Phát biểu <i>DELETE</i> với mệnh đề <i>WHERE</i>	260
3.3. Phát biểu <i>DELETE</i> với mệnh đề <i>TOP</i>	261
3.4. Phát biểu <i>DELETE</i> với mệnh đề <i>OUTPUT</i>	262
3.5. Phát biểu <i>DELETE</i> với mệnh đề <i>WITH</i>	263
4. Kết chương	266

Ứng dụng: SỬ DỤNG PHÁT BIỂU SQL TRONG ỨNG DỤNG..... 267

1. Phân kế toán công nợ phải thu.....	267
1.1. Doanh thu bán hàng theo thời gian	271
1.2. Doanh thu bán hàng theo hóa đơn	274
1.3. Doanh thu bán hàng theo sản phẩm	275
1.4. Doanh thu bán hàng theo khách hàng	278
1.5. Doanh thu bán hàng theo tỉnh thành.....	280
1.6. N sản phẩm có doanh thu lớn nhất.....	283
1.7. Tình hình công nợ của khách hàng.....	287
2. Phân công nợ phải trả	300
2.1. Chi phí mua hàng theo thời gian	303
2.2. Chi phí mua hàng theo hóa đơn	306
2.3. Chi phí mua hàng theo sản phẩm	307
2.4. Chi phí mua hàng theo nhà cung cấp	308
2.5. N sản phẩm có chi phí lớn nhất	309
2.6. Tình hình công nợ phải trả của nhà cung cấp	310
3. Phân tồn quỹ.....	322
3.1. Thu chi trong ngày.....	323
3.2. Thu chi trong ngày.....	325

3.3. Thu chi các ngày trong tháng.....	326
4. Phân xuất nhập tồn	330
4.1. Phân nhập hàng.....	331
4.2. Phân xuất hàng	339
4.3. Phân tồn kho	347
5. Kết chương	359

Chương 1:**GIỚI THIỆU CƠ SỞ DỮ LIỆU
SQL SERVER 2005****Tóm tắt chương 1**

Trước khi bắt đầu với SQL Server 2005, chúng ta tìm hiểu sơ lược về khái niệm cơ sở dữ liệu từ các loại cơ sở dữ liệu cho đến những đặc điểm cần và đủ cho cơ sở dữ liệu quan hệ.

Bên cạnh những khái niệm cơ sở dữ liệu, chúng ta sẽ tìm hiểu các bước chuẩn hóa để có thể tạo ra thực thể của cơ sở dữ liệu kế toán dính kèm theo sách.

Các vấn đề chính sẽ được đề cập:

- ✓ Khái niệm cơ sở dữ liệu.
- ✓ Chuẩn hóa dữ liệu.
- ✓ Giới thiệu SQL Server 2005.
- ✓ Cài đặt SQL Server 2005.

1. KHÁI NIỆM CƠ SỞ DỮ LIỆU

Cơ sở dữ liệu là nơi dùng để chứa thông tin liên quan đến công việc kinh doanh của bạn, những thông tin này có thể đang ở dạng thô hay tổng hợp tùy thuộc vào yêu cầu của người sử dụng.

Việc thông tin đang ở dạng nào thì chúng ta sẽ sử dụng các bước chuẩn hóa dữ liệu mà bạn có thể tìm thấy chi tiết trong chương kế tiếp.

Tuy nhiên, tiền thân của cơ sở dữ liệu vẫn lưu ở dạng văn bản (Text), sau nhiều thời kỳ phát triển, cơ sở dữ liệu hiện nay được sử dụng thịnh hành là mô hình cơ sở dữ liệu quan hệ (RDBMS).

Cơ sở dữ liệu có bốn tiêu chí chính là Field, Record, Table và Database. Trong đó,

- Field (trường): Dùng Field để thể hiện thông tin về một thuộc tính của thực thể trong thế giới thực.
- Record (mẫu tin): Mẫu tin nắm giữ nhiều thuộc tính của một thực thể.
- Table (bảng): Bảng dùng để nắm giữ nhiều mẫu tin của thực thể.
- Database (cơ sở dữ liệu): Dùng nắm giữ những thực thể trong hệ thống.

Như vậy, phần tử cơ bản của cơ sở dữ liệu là đối tượng Table, đối tượng này cho phép bạn lưu trữ thông tin và có quan hệ với đối tượng Table khác, sự quan hệ giữa các bảng dữ liệu chính là khái niệm cơ sở dữ liệu quan hệ (RDBMS).

Chú ý: Một kiểu cơ sở dữ liệu mới đã và đang được sử dụng rộng rãi trong thực tế là cơ sở dữ liệu dạng đối tượng (OOD).

Mặc dù cơ sở dữ liệu dạng OOD đang trở nên quen thuộc, nhưng cơ sở dữ liệu dạng RDBMS khó có thể bị thay thế bởi nó đã được sử dụng rất hiệu quả trong kinh doanh. Đó là lý do tại sao những nhà phát triển phần mềm khổng lồ như Microsoft, Oracle hay IBM không ngừng hoàn thiện cơ sở dữ liệu RDBMS của họ.

Phụ thuộc vào môi trường của ứng dụng, RDBMS vẫn chia thành hai loại, loại thứ nhất được xem như cơ sở dữ liệu dạng một người sử dụng (Single-user) và loại thứ hai là cơ sở dữ liệu đa người dùng (Multi-user) hay còn gọi là cơ sở dữ liệu trình khách và chủ (Client-Server Database).

Đại diện cho nhóm cơ sở dữ liệu dạng Single-user là cơ sở dữ liệu như: Microsoft Access, Microsoft FoxPro. Trong khi đó, cơ sở dữ liệu đa người dùng bao gồm các cơ sở dữ liệu như: Microsoft SQL Server, Oracle hay DB2.

Mỗi loại cơ sở dữ liệu trên đều có lợi ích riêng của chúng. Chẳng hạn, với cấu hình máy của bạn thấp và sử dụng hệ điều hành Windows 9x hay Windows XP thì bạn sử dụng cơ sở dữ liệu Microsoft Access cho ứng dụng đơn giản.

Nếu ứng dụng có quy mô lớn hơn cùng với cấu hình máy đủ mạnh, bạn có thể cài đặt cơ sở dữ liệu Microsoft SQL Server, nhằm cho phép nhiều người sử dụng có thể truy cập đồng thời.

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005

17

Tuy nhiên, một khi bạn xây dựng ứng dụng thương mại điện tử hay quản lý, bạn phải sử dụng cơ sở dữ liệu đa người dùng là tiêu chí bắt buộc và SQL Server là lựa chọn ưu tiên số 1 trong phương án kinh doanh.

Như vậy, cơ sở dữ liệu đa người dùng mạnh, ngoài 4 tiêu chí trên, nó còn bao gồm các khái niệm khác như: Khóa (keys), quan hệ (relations), ràng buộc (constraints), chỉ mục (indexes), quy tắc (rules), giá trị mặc định (defaults) và chuyển tác (transactions).

Chú ý: Chúng ta sẽ tìm thấy các khái niệm khóa, quan hệ, ràng buộc, chỉ mục, quy tắc, giá trị mặc định và chuyển tác trong cơ sở dữ liệu SQL Server 2005.

2. CHUẨN HÓA DỮ LIỆU

Khi bắt đầu với ứng dụng cơ sở dữ liệu, bạn có thể có nhiều phương pháp phân tích dữ liệu, nhưng vấn đề cơ bản của thời kỳ đầu là chỉ ra được các đối tượng tham gia hệ thống.

Dựa vào các đối tượng tham gia hệ thống này, bạn thực hiện tiến trình thu thập thông tin cho từng đối tượng. Mỗi đối tượng có số đặc điểm riêng của nó, nhưng chúng đều nằm trong một hệ thống và có quan hệ với nhau.

Đối với mỗi chuyên gia về phân tích thiết kế hệ thống thông tin, các bước thực hiện chi tiết là có thể khác nhau, nhưng mục đích cuối cùng là cơ sở dữ liệu do họ thiết kế phải bảo đảm 4 nguyên tắc cơ bản là: Toàn vẹn thực thể (Entity integrity), toàn vẹn miền (Domain integrity), toàn vẹn tham chiếu (Referential integrity), toàn vẹn ràng buộc do người sử dụng định nghĩa (User-defined integrity).

Mục tiêu của chuẩn hóa là tránh những sai lệch khi thao tác dữ liệu bằng cách sử dụng ba phát biểu chính là INSERT, UPDATE và DELETE.

Với 3 hành động thay đổi dữ liệu, chúng ta quan tâm đến sự trùng lặp dữ liệu khi thực hiện hành động thêm mới, mất dữ liệu khi thực hiện hành động xóa và thay đổi dữ liệu khi thực hiện hành động cập nhật.

Do vậy, chúng ta có thể tạm thời hiểu chuẩn hóa dữ liệu là quá trình phân rã những lược đồ quan hệ phức tạp chưa được chuẩn hóa ở dạng thấp thành những lược đồ quan hệ nhỏ hơn nhưng lại có dạng chuẩn hóa cao hơn.

Chú ý: Để đánh giá các sai lệch trong ba hành động trên, bạn cần phải tham khảo lý thuyết quan hệ và phụ thuộc hàm cùng với các kiến thức về lược đồ quan hệ và thuật toán tìm kiếm khóa.

Như vậy, mục đích của chuẩn hóa là tạo ra một cơ sở dữ liệu thỏa 4 tiêu chí trên, có 3 dạng chuẩn hóa: First Normal Form, Second Normal Form, Third Normal Form.

Trước khi bàn luận về 3 dạng chuẩn hóa, chúng ta hãy bắt đầu xét thực thể trong ứng dụng kế toán định kèm là hóa đơn bán hàng.

Chẳng hạn, anh Nguyễn Văn A, chị Nguyễn Thị B và chú Hoàng Văn C đã mua hàng của công ty bạn và nhận hóa đơn ứng với 3 sản phẩm là Thuốc, Băng và Kim. Khi xuất hóa đơn bán hàng, nhân viên kế toán sẽ ghi các thông tin (dùng để tham khảo) như: Số hóa đơn, tên khách hàng, địa chỉ khách hàng, ngày xuất hóa đơn, diễn giải hóa đơn, mã sản phẩm, tên kho hàng, tên sản phẩm, đơn vị sản phẩm, số lượng bán, giá bán, thành tiền và các thông tin về thuế khác.

Như vậy, nếu anh Nguyễn Văn A, chị Nguyễn Thị B và chú Hoàng Văn C đã mua hai 3 sản phẩm, thông tin thực tế sẽ được ghi lại khi xuất kho như hình 1-1.

Mã sản phẩm	Tên sản phẩm	Ngày xuất kho	Khách hàng	Kho số 1	Kho số 2	Kho số 3
A001	Thuốc	01/10/2007	Nguyễn Văn A	50	70	20
B001	Băng	05/10/2007	Nguyễn Thị B	20	30	10
C001	Kim	10/10/2007	Hoàng Văn C	45	65	75

Hình 1-1: Dữ liệu chưa áp dụng dạng chuẩn hóa.

Chú ý: Thông tin trên đã được đơn giản hóa để có thể trình bày vừa chiều ngang của trang sách.

Với dữ liệu ban đầu như hình trên, việc chuẩn hóa là quá trình tổ chức lại dữ liệu, quá trình này bao gồm các bước tạo bảng dữ liệu và thiết lập quan hệ giữa chúng theo các quy tắc để bảo vệ dữ liệu và tạo ra cơ sở dữ liệu có tính toàn vẹn thực thể, toàn vẹn miền, toàn vẹn tham chiếu, toàn vẹn ràng buộc do người sử dụng định nghĩa.

2.1. First Normal Form (1NF)

Một lược đồ được chuẩn hóa dạng 1 thỏa 3 quy tắc sau:

- Loại bỏ nhóm dữ liệu trùng lặp trong từng bảng dữ liệu.
- Tạo bảng tách biệt cho từng tập dữ liệu có quan hệ với nhau.
- Khai báo nhận dạng cho mỗi tập dữ liệu có quan hệ với nhau bằng khóa chính.

Như vậy, chuẩn hóa dạng 1 cho phép loại bỏ các thuộc tính đa trị và trùng lặp, tách dữ liệu hay nhóm dữ liệu lặp lại trong hàng ra thành bảng riêng biệt và mỗi hàng phải có nhận dạng duy nhất.

Mục tiêu chính cho chuẩn hóa dạng 1 là bảo đảm dữ liệu không trùng lặp và phải có nhận dạng cho mỗi hàng.

Trong trường hợp này, mỗi hàng có hai thuộc tính nhận dạng là mã khách hàng và số hóa đơn ứng với đối tượng khách hàng và hóa đơn mua hàng của họ như hình 1-2.

Mã phiếu xuất	Tên sản phẩm	Ngày xuất kho	Khách hàng	Số lượng	Mã kho
A001	Thuốc	01/10/2007	Nguyễn Văn A	50	Số 1
A001	Thuốc	01/10/2007	Nguyễn Văn A	70	Số 2
A001	Thuốc	01/10/2007	Nguyễn Văn A	20	Số 3
B001	Băng	05/10/2007	Nguyễn Thị B	20	Số 1
B001	Băng	05/10/2007	Nguyễn Thị B	30	Số 2
B001	Băng	05/10/2007	Nguyễn Thị B	10	Số 3
C001	Kim	10/10/2007	Hoàng Văn C	45	Số 1
C001	Kim	10/10/2007	Hoàng Văn C	65	Số 2
C001	Kim	10/10/2007	Hoàng Văn C	75	Số 3

Hình 1-2: Dữ liệu đã áp dụng chuẩn hóa dạng 1.

Mỗi khi bạn áp dụng chuẩn hóa dạng 1, bạn có thể khai báo thêm những thông tin còn thiếu cho mỗi thực thể dữ liệu (thí dụ thêm địa chỉ kho thì hay hơn).

Mã kho	Tên kho
Số 1	Kho số 1
Số 2	Kho số 2
Số 3	Kho số 2

Hình 1-3: Danh sách kho.

Tuy nhiên, bảng dữ liệu trong hình 1-2 cho thấy dữ liệu trùng lặp trong 4 cột đầu tiên. Chính vì vậy, bạn phải áp dụng chuẩn hóa dạng 2.

2.2. Second Normal Form

Một lược đồ được chuẩn hóa dạng 2 dựa trên lược đồ đã được chuẩn hóa dạng 1 bằng cách:

- Tạo các bảng dữ liệu tách biệt cho những tập giá trị dùng trong nhiều mẫu tin.
- Tạo quan hệ các bảng này thông qua các khóa ngoại.

Dựa trên lược đồ đã áp dụng trong dạng chuẩn 1, bạn tách bảng dữ liệu trong hình 1-2 thành hai bảng dữ liệu ứng với thông tin phiếu xuất và số lượng xuất theo kho tương tự như hình 1-4 và 1-5.

Mã phiếu xuất	Ngày xuất kho	Khách hàng
A001	01/10/2007	Nguyễn Văn A
B001	05/10/2007	Nguyễn Thị B
C001	10/10/2007	Hoàng Văn C

Hình 1-4: Phiếu xuất của khách hàng.

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005

21

Mã phiếu xuất	Tên sản phẩm	Mã kho	Số lượng
A001	Thuốc	Số 1	50
A001	Thuốc	Số 2	70
A001	Thuốc	Số 3	20
B001	Băng	Số 1	20
B001	Băng	Số 2	30
B001	Băng	Số 3	10
C001	Kim	Số 1	45
C001	Kim	Số 2	65
C001	Kim	Số 3	75

Hình 1-5: Chi tiết số lượng xuất theo kho.**2.3. Third Normal Form**

Một lược đồ được chuẩn hóa dạng 3 được xây dựng dựa trên lược đồ đã chuẩn hóa dạng 2 bằng cách chuyển những thuộc tính phụ thuộc vào thuộc tính không khóa thành những quan hệ riêng.

Đối với trường hợp này, bạn chuyển thuộc tính tên khách hàng và tên sản phẩm thành quan hệ riêng bằng cách tách bảng dữ liệu hình 1-4 thành 1-6 và 1-6-1.

Phiếu xuất	Ngày xuất kho	Mã khách hàng
A001	01/10/2007	C001
B001	05/10/2007	C002
C001	10/10/2007	C003

Hình 1-6: Phiếu xuất của khách hàng.

M[®] 22**Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005**

Mã khách hàng	Tên khách hàng	Địa chỉ
C001	Nguyễn Văn A	Nguyễn Văn Đậu Quận 1
C002	Nguyễn Thị B	Nguyễn Thị Bưởi Quận Tân Bình
C003	Hoàng Văn C	Hoàng Văn Thụ Quận Tân Phú

Hình 1-6-1: Danh sách khách hàng.

Tương tự như vậy, bạn áp dụng chuẩn hóa dạng 3 cho bảng 1-5 thành hai bảng dữ liệu như hình 1-7 và 1-7-1.

Mã phiếu xuất	Mã sản phẩm	Mã kho	Số lượng
A001	T00001	Số 1	50
A001	T00001	Số 2	70
A001	T00001	Số 3	20
B001	B00001	Số 1	20
B001	B00001	Số 2	30
B001	B00001	Số 3	10
C001	K00001	Số 1	45
C001	K00001	Số 2	65
C001	K00001	Số 3	75

Hình 1-7: Chi tiết số lượng xuất theo kho.

Mã sản phẩm	Tên sản phẩm	Đơn vị
T00001	Thuốc	Chai
B00001	Băng	Lọ
K00001	Kim	Lô

Hình 1-7-1: Danh sách sản phẩm.

3. GIỚI THIỆU SQL SERVER 2005

SQL Server 2005 là cơ sở dữ liệu đa người dùng do Microsoft phát triển và được sử dụng phổ biến trên thế giới nói chung và ở Việt Nam nói riêng.

Bằng việc cung cấp thêm nhiều tiện ích thông dụng, các đặc điểm, kiểu dữ liệu, hàm, mệnh đề và đối tượng mới; nó giúp nhà phát triển phần mềm lưu trữ, tính toán, thống kê, tìm kiếm và lập báo cáo cho mọi ứng dụng quản lý.

Ngoài ra, SQL Server 2005 cho phép bạn phát triển thủ tục nội tại, hàm, kích hoạt hay kiểu dữ liệu với ngôn ngữ lập trình .NET bằng cách tích hợp.NET Common Language Runtime.

Bằng việc phát triển trình quản lý cơ sở dữ liệu SQL Server từ ngôn ngữ lập trình C#, SQL Server 2005 giới thiệu đến bạn giao diện SQL Server Management Studio tương tự như Visual Studio 2005.

SQL Server 2005, cung cấp nhiều hàm, mệnh đề và một số biểu thức bảng cho phép bạn tổng kết dữ liệu và phân trang bằng phát biểu SQL thông thường thay vì sử dụng thủ tục nội hay kiểu dữ liệu cursor như phiên bản trước.

Với cách cài đặt kiểm soát lỗi bằng try catch như trong ngôn ngữ lập trình C#, Visual Basic, C++ hay Java, bạn có thể kiểm soát lỗi trong thủ tục nội tại thay vì nhận lỗi phát ra từ hệ thống.

3.1. Phần quản trị

Phần quản trị bao gồm các chức năng chính như: cấu hình, đăng ký SQL Server khác, quản lý người sử dụng, quản lý cơ sở dữ liệu, xuất và nhập dữ liệu từ cơ sở dữ liệu khác, sao lưu và phục hồi cơ sở dữ liệu, liên kết cơ sở dữ liệu, tạo thứ bản cơ sở dữ liệu.

Ngoài ra, trong phần quản trị bạn có thể khai báo các tác vụ tự động thực thi phát biểu SQL hay thủ tục nội tại theo thời gian đã định sẵn.

3.2. Phần lập trình

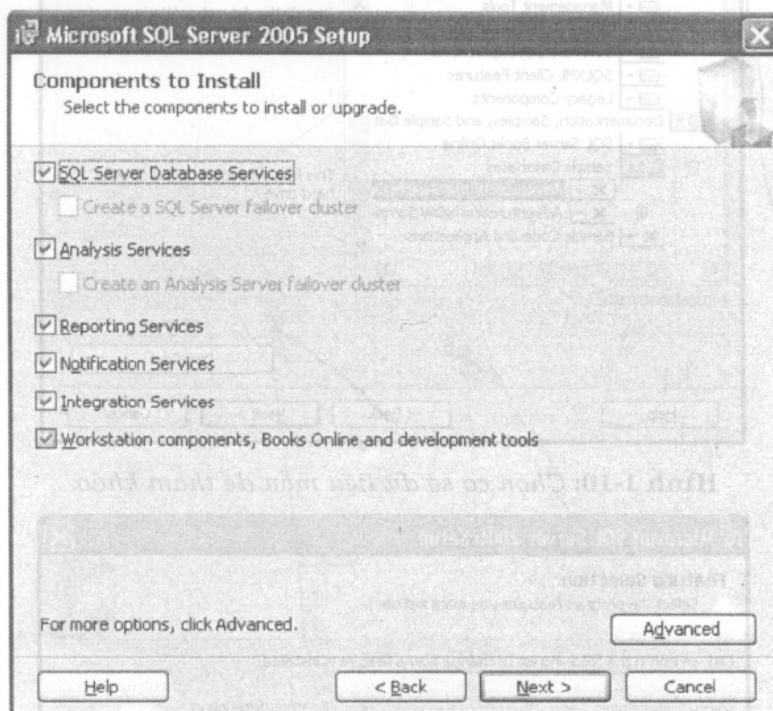
Bao gồm các chức năng cho phép bạn thiết kế cấu trúc cơ sở dữ liệu từ việc tạo bảng dữ liệu, đối tượng View, khai báo đối tượng mệnh đề và các quy tắc ràng buộc.

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005

25

Chú ý: Nếu bạn cài đặt phiên bản Enterprise trên hệ điều hành Windows XP, bạn sẽ tìm thấy một tùy chọn duy nhất là *Workstation components, books online and development tools*.

Tuy nhiên, trong trường hợp nhấn nút Next, bạn có thể tìm thấy các dịch vụ của SQL Server 2005 sẽ cài đặt như hình 1-9.



Hình 1-9: Chọn các dịch vụ SQL Server 2005.

Khi nhấn nút Next, cửa sổ kế tiếp xuất hiện như hình 1-10, mặc định SQL Server 2005 không chọn vào cơ sở dữ liệu mẫu đính kèm theo đĩa, bạn có thể chọn vào ngăn Sample Databases rồi chọn chúng để cài đặt như một phần cơ sở dữ liệu thực hành như hình 1-11.

Chú ý: bạn có thể tải cơ sở dữ liệu mẫu AdventureWorks và AdventureWorksDW đính kèm theo SQL Server 2005 tại địa chỉ <http://www.microsoft.com/downloads/details.aspx?familyid=e719ecf7-9f46-4312-af89-6ad8702e4e6&displaylang=en>.



Hình 1-11: Cửa sổ đầu tiên

Ngoài ra, bạn cũng có thể định nghĩa hàm và khai báo thủ tục nội tại để có thể thao tác dữ liệu, tính toán, thống kê, sắp xếp, kích hoạt, tìm kiếm dữ liệu.

3.3. Tiện ích khác

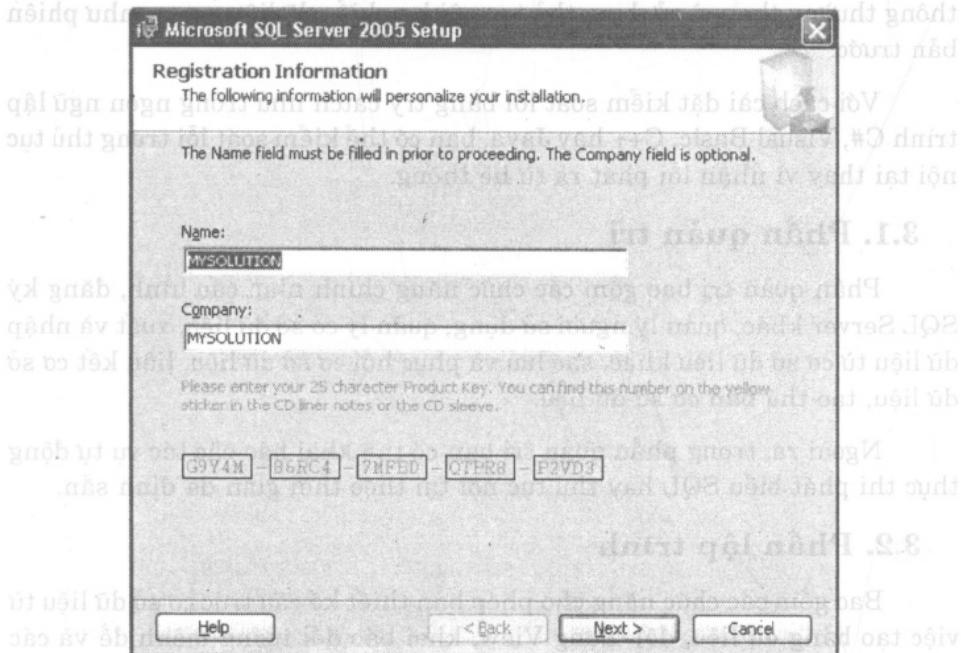
Bạn cũng có thể sử dụng các tiện ích khác để theo dõi tiến trình thực thi chuyển tác trong cơ sở dữ liệu hay các dịch vụ khác như: SQL Server Profiler và Database Engine Tuning Advisor.

4. CÀI ĐẶT SQL SERVER 2005

Nhằm tiện theo dõi nội dung trình bày trong tập sách SQL Server 2005, bạn có thể cài đặt phiên bản SQL Server 2005 Developer nếu bạn đang sử dụng hệ điều hành Windows XP.

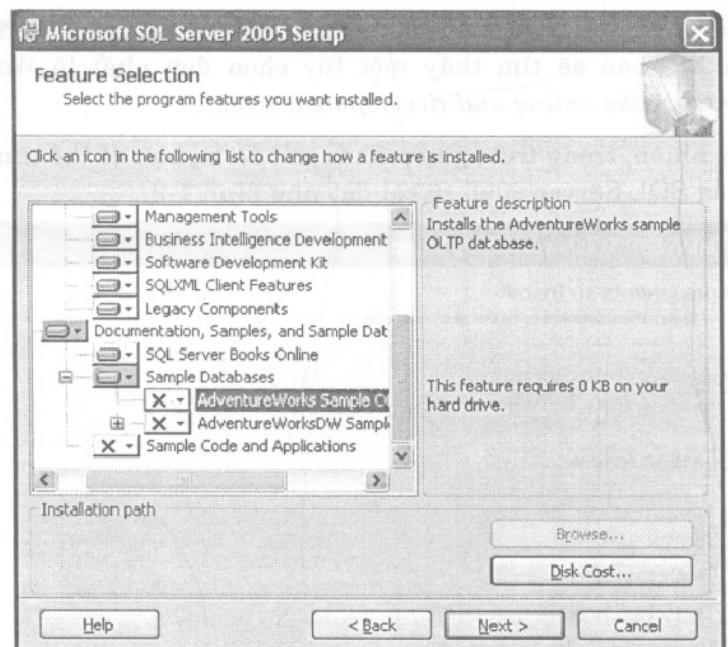
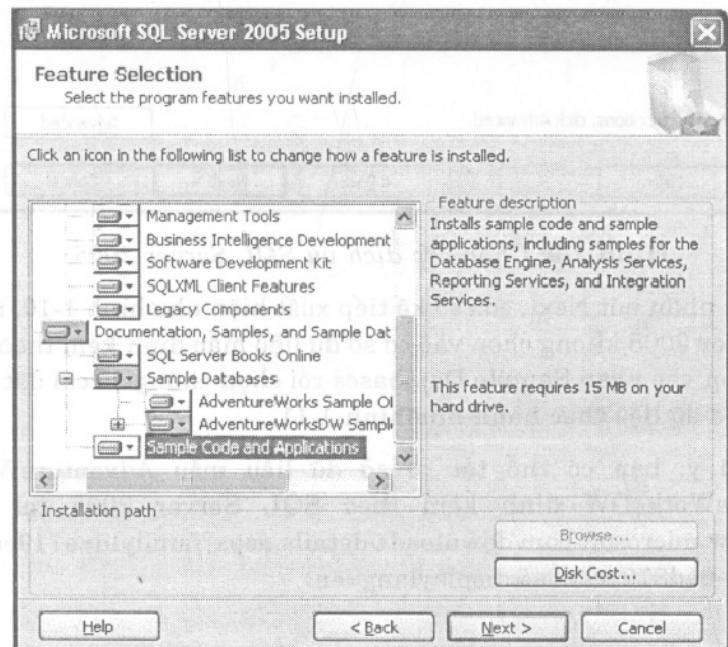
Trong trường hợp sử dụng hệ điều hành Windows Server 2003, bạn có thể cài đặt phiên bản Enterprise.

Để cài đặt SQL Server 2005 phiên bản Developer, bạn chạy tập tin setup, cửa sổ đăng ký tên công ty xuất hiện như hình 1-8.



Hình 1-8: Bắt đầu cài đặt.

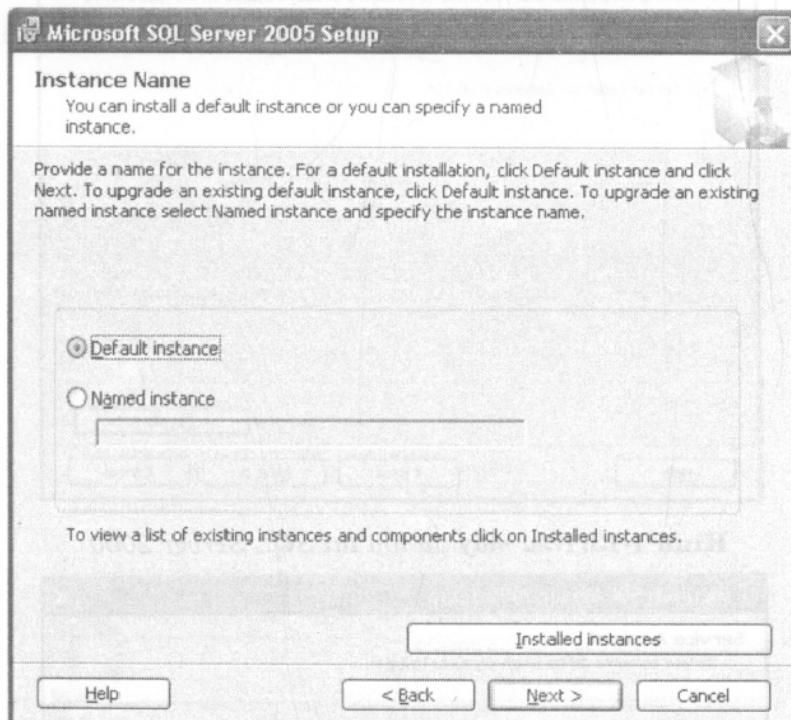
M® 26

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005**Hình 1-10:** Chọn cơ sở dữ liệu mẫu để tham khảo.**Hình 1-11:** Chọn ứng dụng mẫu.

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005

27

Sau khi chọn nút Next, cửa sổ yêu cầu bạn chọn Instance cài đặt là mặc định hay tên khác như hình 1-12. Trong trường hợp đã tồn tại cơ sở dữ liệu SQL Server 2000 trên máy, bạn chỉ có thể cài đặt SQL Server 2005 ứng với tên chỉ định.



Hình 1-12: Cài đặt SQL Server 2005 mặc định.

Chú ý: Trong trường hợp máy của bạn đã cài đặt SQL Server 2000, SQL Server 2005 sẽ là thứ bản bằng cách chọn vào Named instance ứng với tên tham chiếu trong khi truy cập từ ứng dụng khác.

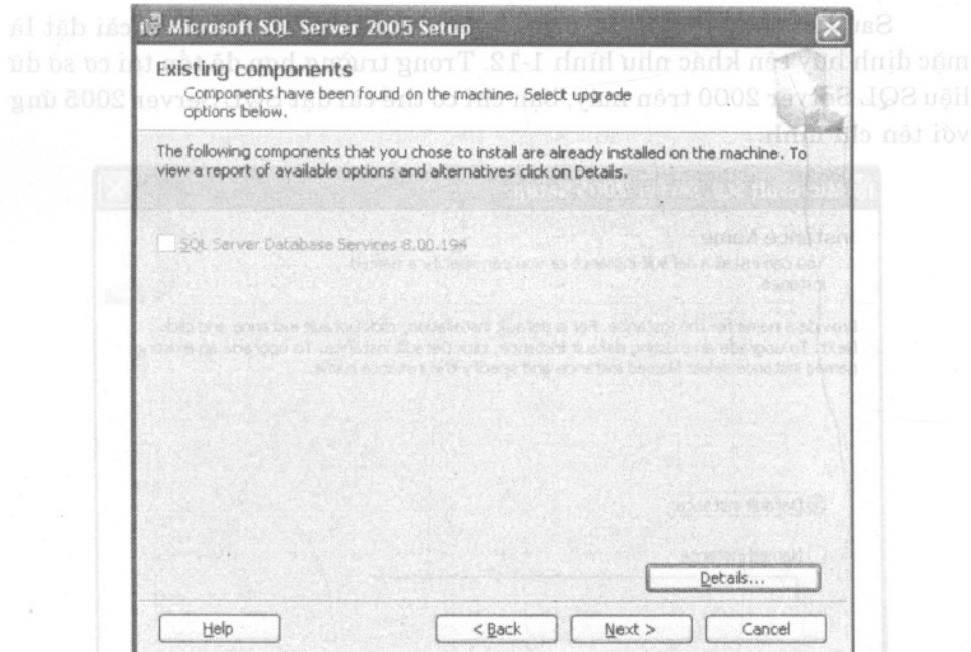
Chọn Next, cửa sổ kế tiếp trình bày như hình 1-13, nếu bạn đã cài đặt SQL Server 2000 trước đó, tùy chọn SQL Server Database Services 8.00.194 cho phép bạn nâng cấp SQL Server 2000 lên 2005.

Bước kế tiếp, bạn cần chỉ định tài khoản khởi động các dịch vụ SQL Server 2005. Nếu chỉ định một tài khoản cụ thể như hình 1-14, chỉ người sử dụng đăng nhập với tài khoản đó mới có thể vận hành dịch vụ của SQL Server 2005.

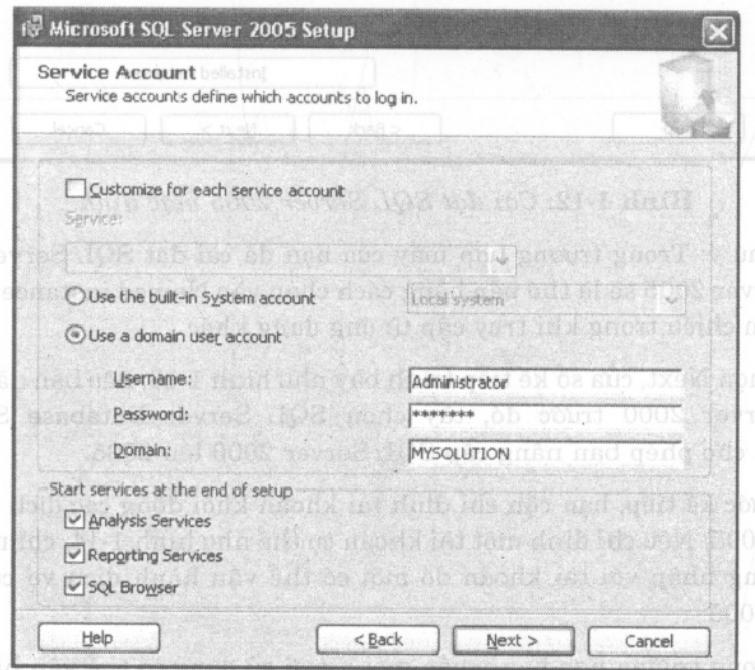
Trong trường hợp bạn muốn mọi người sử dụng có thể vận hành các dịch vụ SQL Server 2005 thì khai báo Local System như hình 1-15.

M® 28

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005



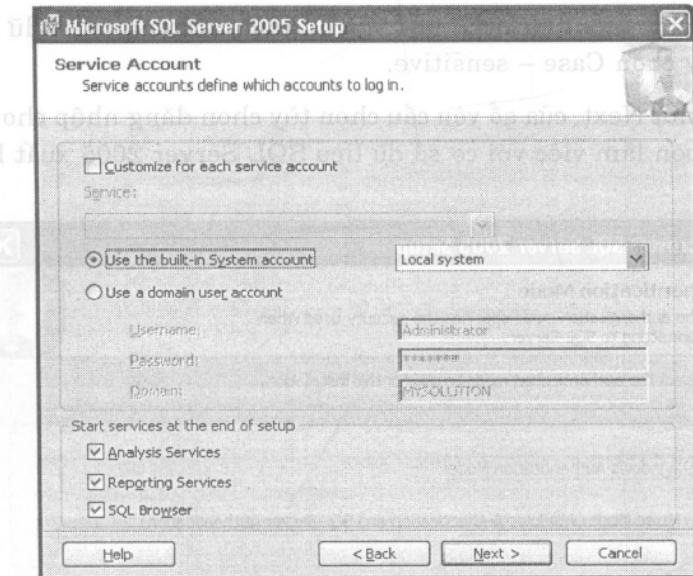
Hình 1-13: Nếu máy đã tồn tại SQL Server 2000.



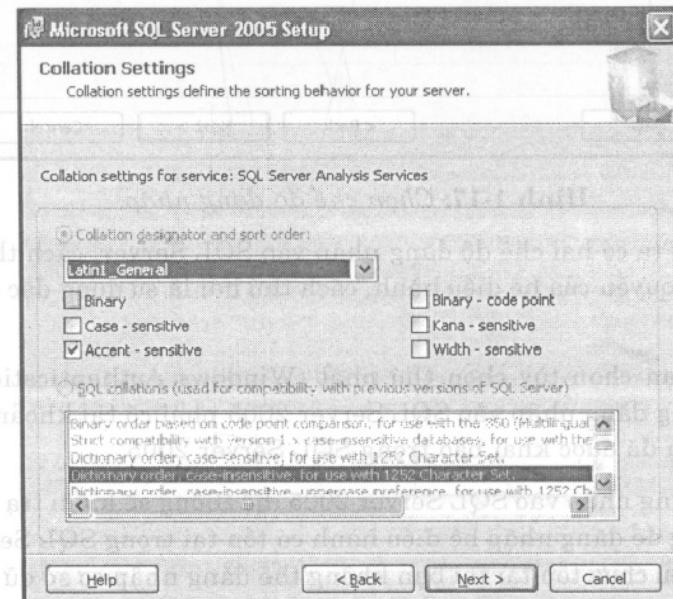
Hình 1-14: Chọn tài khoản để khởi động các dịch vụ SQL Server 2005.

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005

29 M®

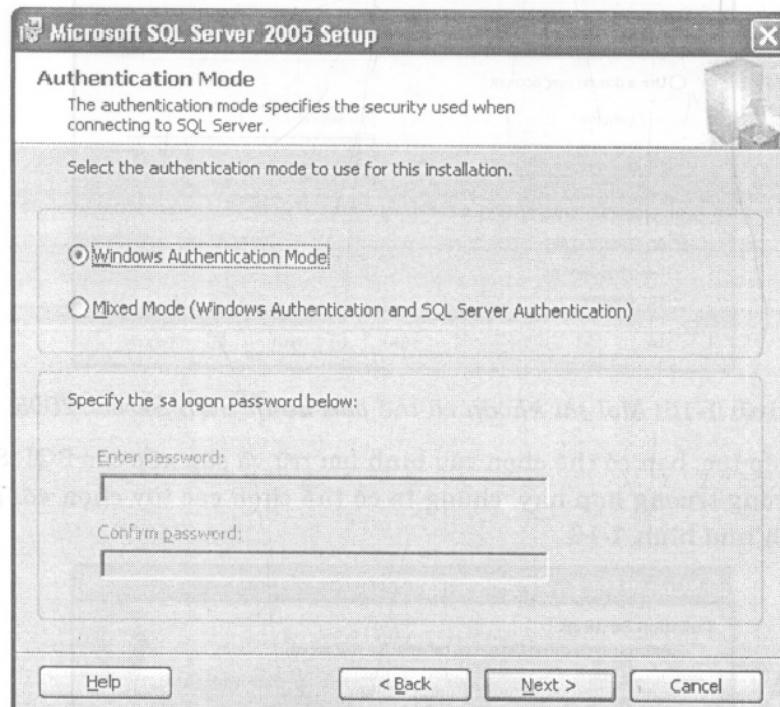
**Hình 1-15:** Mọi tài khoản có thể vận hành SQL Server 2005.

Tiếp tục, bạn có thể chọn cấu hình lưu trữ và sắp xếp của SQL Server 2005. Trong trường hợp này, chúng ta có thể chọn các tùy chọn với giá trị mặc định như hình 1-16.

**Hình 1-16:** Chọn kiểu sắp xếp dữ liệu của SQL Server 2005.

Nếu muốn phân biệt chữ hoa và chữ thường trong cơ sở dữ liệu, bạn chọn vào tùy chọn Case – sensitive.

Chọn nút Next, cửa sổ yêu cầu chọn tùy chọn đăng nhập cho người sử dụng khi muốn làm việc với cơ sở dữ liệu SQL Server 2005 xuất hiện như hình 1-17.



Hình 1-17: Chọn chế độ đăng nhập.

Chúng ta có hai chế độ đăng nhập vào SQL Server, cách thứ nhất là sử dụng đặc quyền của hệ điều hành, cách thứ hai là sử dụng đặc quyền của SQL Server.

Nếu bạn chọn tùy chọn thứ nhất (Windows Authentication mode), người sử dụng đăng nhập vào SQL Server 2005 phải có tài khoản trong hệ điều hành và đã được khai báo trong SQL Server 2005.

Khi đăng nhập vào SQL Server 2005, hệ thống sẽ kiểm tra tài khoản mà bạn dùng để đăng nhập hệ điều hành có tồn tại trong SQL Server 2005 hay chưa. Nếu chưa tồn tại thì bạn không thể đăng nhập cơ sở dữ liệu, thay vào đó bạn sẽ nhận được thông báo lỗi.

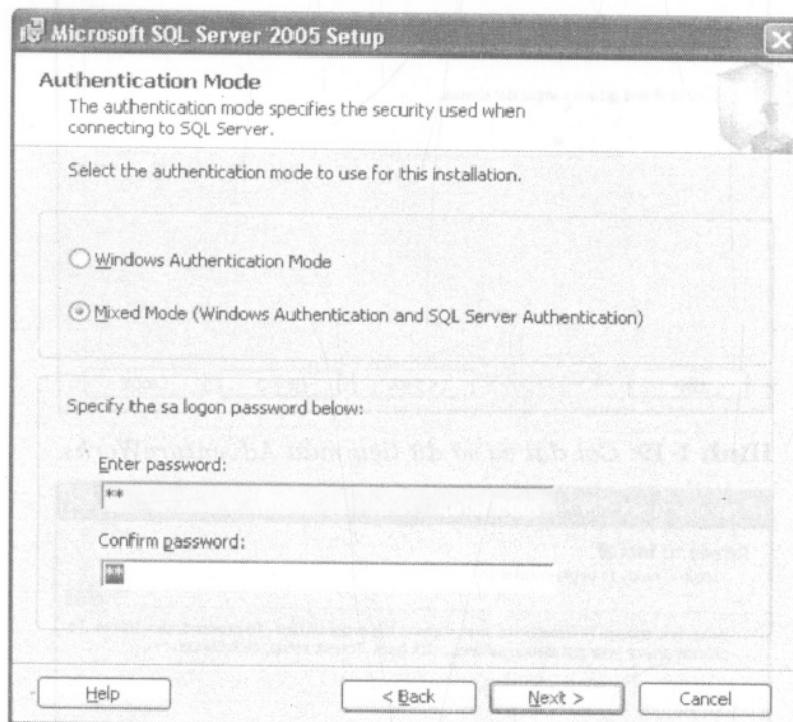
Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005

31



Tuy nhiên, bạn có thể cho phép người sử dụng có thể đăng nhập vào SQL Server 2005 bằng tài khoản của hệ điều hành hay tài khoản của SQL Server 2005 thì chọn vào tùy chọn thứ hai (Mixed mode) như hình 1-18.

Tài khoản mặc định của SQL Server 2005 là sa (system admin) luôn tồn tại khi bạn cài đặt SQL Server 2005, khi bạn chọn vào tùy chọn thứ hai thì trình cài đặt yêu cầu cung cấp mật khẩu cho tài khoản sa.



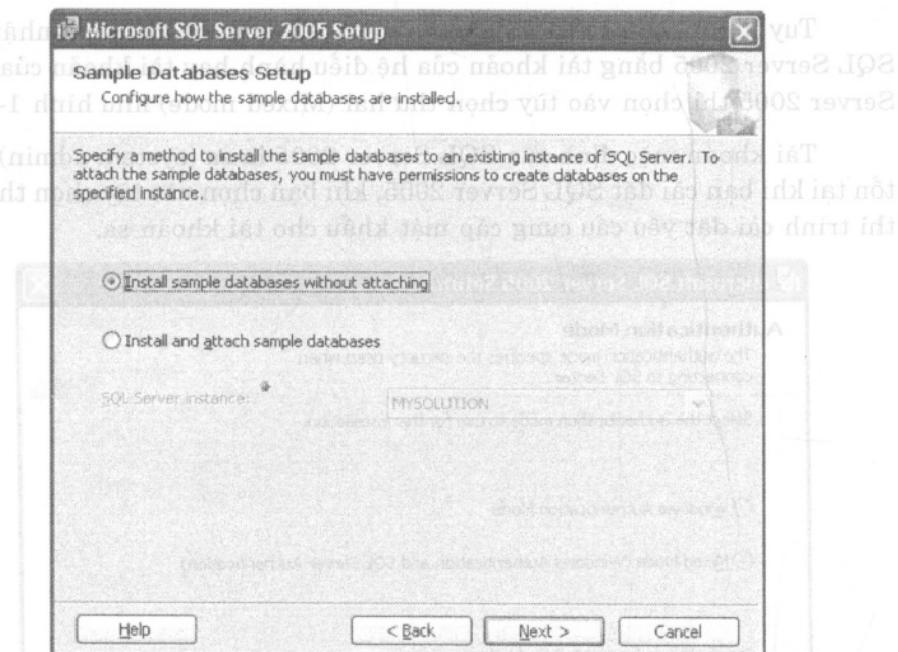
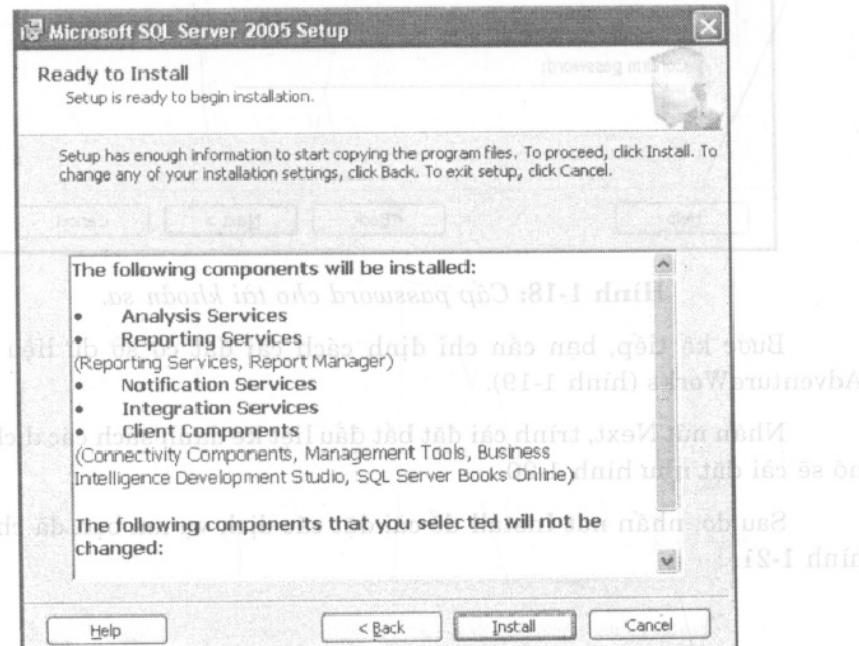
Hình 1-18: Cấp password cho tài khoản sa.

Bước kế tiếp, bạn cần chỉ định cách cài đặt cơ sở dữ liệu mẫu là AdventureWorks (hình 1-19).

Nhấn nút Next, trình cài đặt bắt đầu liệt kê danh sách các dịch vụ mà nó sẽ cài đặt như hình 1-20.

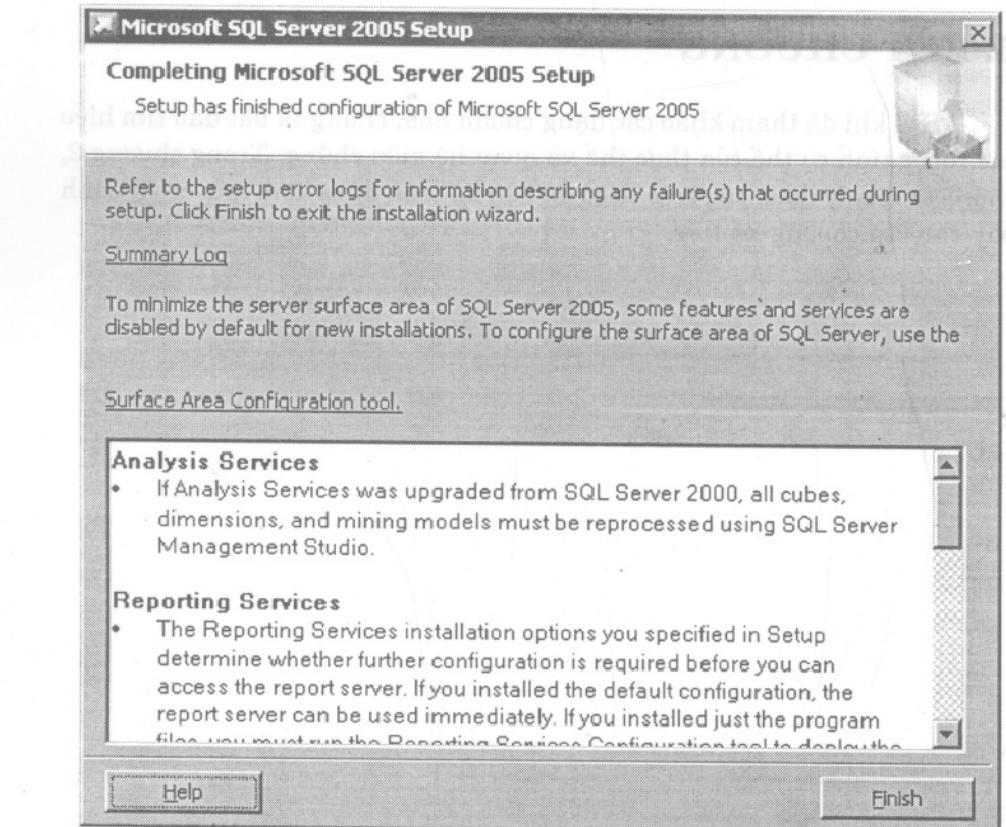
Sau đó, nhấn nút Install để cài đặt các dịch vụ mà bạn đã chọn như hình 1-21.

M® 32

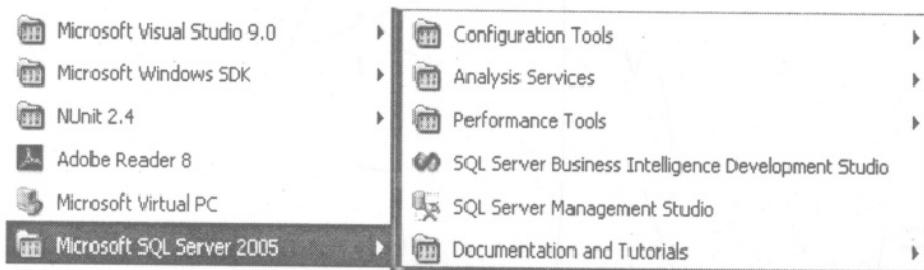
Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005**Hình 1-19:** Cài đặt cơ sở dữ liệu mẫu AdventureWorks.**Hình 1-20:** Thông báo chuẩn bị cài đặt các thành phần của SQL Server 2005.

Chương 1: Giới thiệu cơ sở dữ liệu SQL Server 2005

33 M®

**Hình 1-21:** Kết thúc cài đặt SQL Server 2005.

Sau khi cài đặt thành công, bạn có thể tìm thấy thực đơn của ứng dụng SQL Server 2005, chọn SQL Server Management Studio để mở trình làm việc với SQL Server 2005 như hình 1-22.

**Hình 1-22:** Sử dụng SQL Server Management Studio.

5. KẾT CHƯƠNG

Sau khi đã tham khảo các dạng chuẩn hóa, chúng ta bắt đầu tìm hiểu các thuộc tính cụ thể của thực thể và quan hệ giữa chúng. Trong chương 2, chúng ta sẽ xây dựng cấu trúc cơ sở dữ liệu kế toán dính kèm làm cơ sở trình bày cho các chương kế tiếp.

Chương 2:**TÌM HIỂU CƠ SỞ DỮ LIỆU
ĐÍNH KÈM****Tóm tắt chương 2**

Nhằm tiện theo dõi các chương kế tiếp, chúng ta sẽ tìm hiểu 4 phần chính trong cơ sở dữ liệu AccountSystem được đặt tên như sau: General Ledger (Kế toán tổng hợp), Account Receivable (Kế toán công nợ thu), Account Payable (Kế toán công nợ chi) và Inventory Control (Kế toán kho).

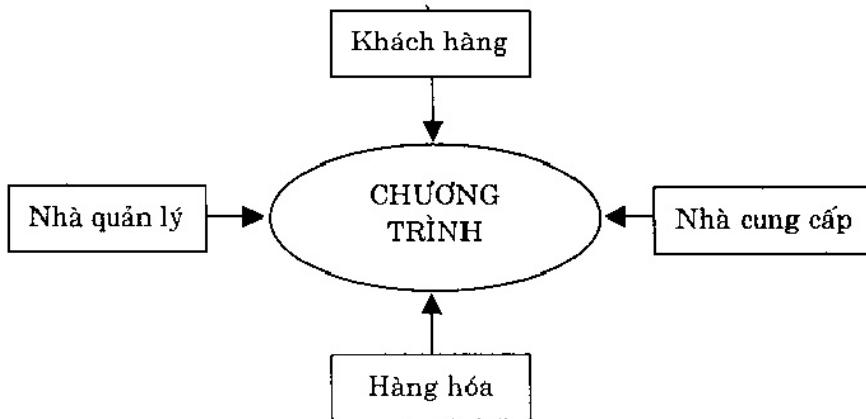
Với mong muốn sử dụng thống nhất cơ sở dữ liệu AccountSystem trong suốt tập sách SQL Server 2005, bạn có thể khám phá cấu trúc cơ sở dữ liệu của chúng qua từng bước thiết kế.

Các vấn đề chính sẽ được đề cập:

- ✓ Giới thiệu cơ sở dữ liệu AccountSystem.
- ✓ Cấu trúc cơ sở dữ liệu phần General Ledger.
- ✓ Cấu trúc cơ sở dữ liệu phần Account Receivable.
- ✓ Cấu trúc cơ sở dữ liệu phần Account Payable.
- ✓ Cấu trúc cơ sở dữ liệu phần Inventory Control.

**1. GIỚI THIỆU CƠ SỞ DỮ LIỆU
ACCOUNTSYSTEM**

Như đã giới thiệu trong chương 1, đối tượng tham gia trong mô hình kế toán bao gồm người sử dụng, khách hàng, nhà cung cấp và hàng hóa hay dịch vụ mà bạn kinh doanh như hình 2-1.



Hình 2-1: Đối tượng tham gia vào hệ thống AccountSystem.

Khi một nghiệp vụ phát sinh trong kinh doanh, chúng đều được ghi nhận vào hệ thống kế toán để cho phép nhân viên có thể lập báo cáo phục vụ cho các nhà quản lý hay cơ quan nhà nước đúng theo quy định của luật kế toán.

Ứng dụng cơ sở dữ liệu AccountSystem bao gồm 4 Module chính là General Ledger (GL), Account Receivable (AR), Account Payable (AP) và Inventory Control (IC).

Trong đó, General Ledger thực hiện chức năng quản lý các nguồn thu và chi tiền, Account Receivable ghi nhận hóa đơn bán hàng cho khách hàng để cho phép nhân viên kế toán quản lý công nợ phải thu, Account Payable ghi nhận hóa đơn mua hàng của nhà cung cấp để cho phép nhân viên kế toán quản lý công nợ phải trả và Inventory Control ghi nhận phiếu xuất hay nhập kho để cho phép nhân viên kế toán quản lý tình hình xuất nhập tồn sản phẩm.

Chú ý: Với tiêu chí là xây dựng cơ sở dữ liệu tham khảo để đính kèm theo sách, chính vì vậy chúng tôi chỉ tập trung xây dựng cấu trúc cơ sở dữ liệu chứ không đi theo từng bước của quá trình phân tích thiết kế hệ thống, do đó thiếu sót về mặt lý thuyết kế toán lẫn tiến trình chuẩn hóa là điều không tránh khỏi.

Tuy nhiên, để có được các thực thể trong từng module của ứng dụng, chúng ta cũng đã sử dụng các bước chuẩn hóa dữ liệu đã trình bày trong chương trước.

2. CẤU TRÚC CƠ SỞ DỮ LIỆU PHẦN GENERAL LEDGER

Để cho phép lưu trữ thông tin về các khoản thu chi, chúng ta cần xây dựng các bảng dữ liệu chính và bảng dữ liệu liên quan đến phần này bao gồm các nhóm bảng dữ liệu như sau:

- Phiếu thu bao gồm các bảng dữ liệu: ReceiptBatchs, Receipts, ReceiptTypes, ReceiptToAccounts.
- Phiếu chi bao gồm các bảng dữ liệu: PaymentBatchs, Payments, PaymentTypes, PaymentToAccounts.
- Tồn quỹ bao gồm các bảng dữ liệu: MonthlyCashBalances và CloseMonthCashBalances.
- Tài khoản bao gồm các bảng dữ liệu: AccountNumbers, SubAccountNumbers. Chúng ta có thể xem xét lại quá trình phân bổ tài khoản trong các tập kế tiếp.

Chú ý: Bạn có thể tham khảo phần trình bày về kiểu dữ liệu trong chương 4 nhằm kiểm tra xem chúng ta chọn kiểu dữ liệu trong các bảng dữ liệu của cơ sở dữ liệu AccountSystem có phù hợp hay không.

2.1. Phiếu thu

2.1.1. Bảng ReceiptBatchs

Bảng ReceiptBatchs dùng để chứa lô (batch) nhiều phiếu thu bao gồm các cột dữ liệu với cấu trúc như sau:

```
ReceiptBatchNo VARCHAR(10) NOT NULL PRIMARY KEY,
ReceiptBatchDate SMALLDATETIME DEFAULT GETDATE(),
Locked BIT DEFAULT 0,
UserName VARCHAR(10) NOT NULL ,
Discontinued BIT DEFAULT 0
```

Trong đó, cột:

- ReceiptBatchNo là mã số lô phiếu thu.
- ReceiptBatchDate ứng với ngày nhập lô phiếu thu.
- Discontinued ứng với trạng thái lô phiếu thu đóng hay đang còn mở để cho phép nhân viên nhập phiếu thu.
- UserName ứng với tài khoản nhân viên tạo ra lô phiếu thu.

2.1.2. *Bảng Receipts*

Bảng Receipts dùng để lưu trữ thông tin phiếu thu tiền, chúng có cấu trúc như sau:

```
ReceiptNo VARCHAR(10) NOT NULL PRIMARY KEY,
ReceiptBatchNo VARCHAR(10) NULL,
ReceiptDate SMALLDATETIMEDEFAULT GETDATE(),
ReceiptTypeID CHAR(3) NOT NULL,
CustomerID CHAR(5) NOT NULL,
CurrencyID CHAR(3) NOT NULL,
Amount DECIMAL DEFAULT 0,
ExchangeRate DECIMAL DEFAULT 0,
ReceiptAmount DECIMAL DEFAULT 0,
DescriptionInVietnamese NVARCHAR(150)NOT NULL,
DescriptionInSecondLanguage VARCHAR(150) NULL,
ShowDescriptionInVietnamese BIT DEFAULT 1,
ReceiptDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- ReceiptNo là mã số phiếu thu được ghi trên mỗi phiếu thu.
- ReceiptBatchNo là mã lô phiếu thu.
- ReceiptDate ứng với ngày thu tiền.
- CustomerID ứng với mã khách hàng trả tiền.
- CurrencyID ứng với mã tiền tệ.
- Amount là số tiền thu.
- ExchangeRate ứng với tỷ giá của tiền tệ khai báo trong cột CurrencyID so với tiền chúc năng là VND.
- ReceiptAmount là số tiền sau khi quy đổi thành tiền VND.
- DescriptionInVietnamese ứng với diễn giải cho khoản thu bằng tiếng Việt.
- DescriptionInSecondLanguage ứng với diễn giải cho khoản thu bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải của phiếu thu bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

39

- EntryDate ứng với ngày nhập phiếu thu, cột này tự động nhập ngày mà mẫu tin được thêm vào bảng dữ liệu, không cho phép người sử dụng thay đổi giá trị cho cột này trên giao diện ứng dụng.
- ReceiptDiscontinued ứng với trạng thái phiếu thu đóng hay đang còn mở cho phép nhân viên cập nhật phiếu thu.
- UserName ứng với tài khoản của nhân viên nhập thông tin của phiếu thu.

2.1.3. Bảng ReceiptTypes

Bảng ReceiptTypes dùng để lưu trữ thông tin loại phiếu thu với cấu trúc như sau:

```
ReceiptTypeID CHAR(3) NOT NULL PRIMARY KEY,
ReceiptTypeInVietnamese NVARCHAR(50) NOT NULL,
ReceiptTypeInSecondLanguage VARCHAR(50) NULL,
ShowReceiptTypeInVietnamese BIT DEFAULT 1
```

Trong đó, cột:

- ReceiptTypeID là mã loại phiếu thu.
- ReceiptTypeInVietnamese ứng với diễn giải cho loại phiếu thu bằng tiếng Việt.
- ReceiptTypeInSecondLanguage ứng với diễn giải cho loại phiếu thu bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải của loại phiếu thu bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.

2.1.4. Bảng ReceiptToAccounts

Bảng ReceiptToAccounts dùng để lưu thông tin tài khoản được phân bổ khi người sử dụng nhập thông tin phiếu thu.

```
ReceiptNo VARCHAR(10) NOT NULL,
DebitNo VARCHAR(20) NOT NULL,
CreditNo VARCHAR(20) NOT NULL,
Amount DECIMAL DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE()
PRIMARY KEY(ReceiptNo, DebitNo, CreditNo)
```

Trong đó, cột:

- ReceiptNo là mã số phiếu thu.
- DebitNo là mã tài khoản nợ.

- CreditNo là mã tài khoản có.
- Amount là số tiền phân bổ cho tài khoản nợ và có tương ứng.
- EntryDate là ngày phân bổ tài khoản khi nghiệp vụ thu tiền xảy ra.

2.2. Phiếu chi

2.2.1. Bảng PaymentBatchs

Bảng PaymentBatchs dùng để chứa lô (batch) gồm nhiều phiếu chi, chúng bao gồm các cột dữ liệu với cấu trúc như sau:

```
PaymentBatchNo VARCHAR(10)
    NOT NULL PRIMARY KEY,
PaymentBatchDate SMALLDATETIME
    DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL ,
Discontinued BIT DEFAULT 0
```

Trong đó, cột:

- PaymentBatchNo là mã số lô phiếu chi.
- PaymentBatchDate ứng với ngày nhập lô phiếu chi.
- Discontinued ứng với trạng thái lô phiếu chi đóng hay đang còn mở cho phép nhân viên nhập phiếu chi.
- UserName ứng với tài khoản của nhân viên tạo lô phiếu chi.

2.2.2. Bảng Payments

Bảng Payments dùng để lưu trữ thông tin phiếu chi tiền có cấu trúc như sau:

```
PaymentNo VARCHAR(10) NOT NULL PRIMARY KEY,
PaymentBatchNo VARCHAR(10)  NULL,
PaymentDate SMALLDATETIMEDEFAULT GETDATE(),
PaymentTypeID CHAR(3) NOT NULL,
SupplierID CHAR(5) NOT NULL ,
CurrencyID CHAR(3) NOT NULL,
Amount DECIMAL DEFAULT 0,
ExchangeRate DECIMAL DEFAULT 0,
PaymentAmount DECIMAL DEFAULT 0,
DescriptionInVietnamese NVARCHAR(150)NOT NULL,
DescriptionInSecondLanguage VARCHAR(150) NULL,
ShowDescriptionInVietnamese BIT DEFAULT 1,
PaymentDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Chương 2: Tìm hiểu cơ sở dữ liệu dính kèm

41

Trong đó, cột:

- PaymentNo là mã số phiếu chi được ghi trên mỗi phiếu thu.
- PaymentBatchNo mã lô phiếu chi
- PaymentDate ứng với ngày chi tiền.
- PaymentTypeID ứng với loại phiếu chi.
- SupplierID ứng với mã nhà cung cấp chi tiền.
- CurrencyID ứng với mã tiền tệ.
- Amount là số tiền chi.
- ExchangeRate ứng với tỷ giá của tiền tệ khai báo trong cột CurrencyID so với tiền chẵn nồng là VND.
- PaymentAmount là số tiền sau khi quy đổi thành tiền VND.
- DescriptionInVietnamese ứng với diễn giải cho khoản thu bằng tiếng Việt.
- DescriptionInSecondLanguage ứng với diễn giải cho khoản chi bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải của phiếu chi bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- EntryDate ứng với ngày nhập phiếu chi, cột này tự động nhập ngày mà mẫu tin được thêm vào bảng dữ liệu, không cho phép người sử dụng thay đổi giá trị cho cột này trên giao diện ứng dụng.
- PaymentDiscontinued ứng với trạng thái phiếu chi đóng hay đang còn mở cho phép cập nhật phiếu chi.
- UserName ứng với tài khoản nhân viên nhập thông tin của phiếu chi.

2.2.3. Bảng PaymentTypes

Bảng PaymentTypes dùng để lưu trữ thông tin loại phiếu chi với cấu trúc như sau:

```
PaymentTypeID CHAR(3) NOT NULL PRIMARY KEY,
PaymentTypeInVietnamese NVARCHAR(50) NOT NULL,
PaymentTypeInSecondLanguage VARCHAR(50) NULL,
ShowReceiptTypeInVietnamese BIT DEFAULT 1
```

Trong đó, cột:

- PaymentTypeID là mã loại phiếu chi.
- PaymentTypeInVietnamese ứng với diễn giải cho loại phiếu chi bằng tiếng Việt.
- PaymentTypeInSecondLanguage ứng với diễn giải cho loại phiếu chi bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải của loại phiếu chi bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.

2.2.4. Bảng ReceiptToAccounts

Bảng ReceiptToAccounts dùng để lưu thông tin tài khoản được phân bổ khi người sử dụng nhập thông tin phiếu thu.

```
ReceiptNo VARCHAR(10) NOT NULL,
DebitNo VARCHAR(20) NOT NULL,
CreditNo VARCHAR(20) NOT NULL,
Amount DECIMAL DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE()
PRIMARY KEY(ReceiptNo, DebitNo, CreditNo)
```

Trong đó, cột:

- ReceiptNo là mã số phiếu chi.
- DebitNo là mã tài khoản nợ.
- CreditNo là mã tài khoản có.
- Amount là số tiền phân bổ cho tài khoản nợ và có tương ứng.
- EntryDate là ngày phân bổ tài khoản khi nghiệp vụ chi tiền xảy ra.

2.3. Tôn quỹ

2.3.1. Bảng MonthlyCashBalances

Bảng MonthlyCashBalances dùng để lưu trữ dữ liệu khi tính toán tồn quỹ, bảng này chứa dữ liệu tổng kết từ hai bảng Receipts và Payments có cấu trúc tương tự như mẫu báo cáo tồn quỹ trong thực tế.

```
IssueID INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
IssueNo VARCHAR(10) NOT NULL,
IssueBatchNo VARCHAR(10) NULL,
IssueDate SMALLDATETIME DEFAULT GETDATE(),
PartnerID CHAR(5) NOT NULL,
ReceiptAmount DECIMAL DEFAULT 0,
```

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

43

```
PaymentAmount DECIMAL DEFAULT 0 ,
BalanceAmount DECIMAL DEFAULT 0 ,
Description NVARCHAR(150) NOT NULL
```

Trong đó, cột:

- IssueID ứng với mã mẫu tin (số tự động phát sinh) khi dữ liệu được thêm vào từ bảng Receipts hay Payments.
- IssueNo là mã số phiếu thu hay chi tương ứng với cột PaymentNo hay ReceiptNo được lấy từ bảng Receipts hay Payments.
- IssueBatchNo là mã số lô phiếu thu hay chi tương ứng với cột PaymentBatchNo hay ReceiptBatchNo được lấy từ bảng Receipts hay Payments.
- IssueDate ứng với ngày thu tiền hay chi tiền được lấy từ bảng Receipts hay Payments.
- PartnerID ứng với mã khách hàng khi thu tiền hay mã nhà cung cấp khi chi tiền được lấy từ bảng Receipts hay Payments.
- ReceiptAmount ứng với số tiền thu lấy từ bảng Receipts.
- PaymentAmount ứng với số tiền chi lấy từ bảng Payments.
- BalanceAmount là số tiền cân đối tại thời điểm xảy ra nghiệp vụ thu hay chi tiền.
- Description là diễn giải thu hay chi tiền lấy từ bảng Receipts hay Payments.

2.3.2. Bảng CloseMonthCashBalances

Bảng CloseMonthCashBalances dùng để lưu trữ dữ liệu khi tính toán tồn quỹ rồi đóng sổ kế toán theo định kỳ hàng tháng, bảng này chứa dữ liệu tổng kết từ hai bảng Receipts và Payments của tháng trước so với tháng đang tính sổ.

Bảng CloseMonthCashBalances có cấu trúc tương tự như mẫu báo cáo tồn quỹ trong thực tế.

```
CloseMonth CHAR(7) NOT NULL PRIMARY KEY ,
BeginningAmount DECIMAL DEFAULT 0 ,
ReceiptAmount DECIMAL DEFAULT 0 ,
PaymentAmount DECIMAL DEFAULT 0 ,
EndingAmount DECIMAL DEFAULT 0 ,
CloseMonthDate SMALLDATETIME
    NOT NULL DEFAULT GETDATE() ,
UserNameToCloseMonth VARCHAR(10) NOT NULL
```

Trong đó, cột:

- CloseMonth là tháng và năm đóng sổ của tháng trước.
- BeginingAmount là số tiền tồn quỹ đầu kỳ.
- ReceiptAmount là số tiền thu đầu kỳ.
- PaymentAmount là số tiền chi đầu kỳ.
- EndingAmount là số tiền tồn quỹ cuối kỳ.
- CloseMonthDate ứng với ngày đóng sổ kế toán.
- UserNameToCloseMonth ứng với tài khoản người sử dụng đóng sổ kế toán.

2.3.3. Bảng CloseMonthCashBalanceDetails

Bảng CloseMonthCashBalanceDetails dùng để lưu trữ dữ liệu chi tiết khi tính toán tồn quỹ rồi đóng sổ kế toán theo định kỳ hàng tháng, bảng này chứa dữ liệu tổng kết từ hai bảng Receipts và Payments của tháng trước so với tháng đang tính sổ.

Bảng CloseMonthCashBalanceDetails có cấu trúc tương tự như mẫu báo cáo tồn quỹ trong thực tế.

```
IssueID INT NOT NULL,
CloseMonth CHAR(7) NOT NULL,
IssueNo VARCHAR(10) NOT NULL,
IssueBatchNo VARCHAR(10) NOT NULL,
IssueDate SMALLDATETIME DEFAULT GETDATE(),
PartnerID CHAR(5) NOT NULL ,
FunctionalCurrencyID CHAR(3) NOT NULL,
ReceiptAmount DECIMAL DEFAULT 0,
PaymentAmount DECIMAL DEFAULT 0,
BalanceAmount DECIMAL DEFAULT 0,
Description NVARCHAR(150) NOT NULL,
PRIMARY KEY
(IssueNo, IssueBatchNo , CloseMonth)
```

Trong đó, cột:

- IssueID ứng với mã mẫu tin (số tự động phát sinh) khi dữ liệu được thêm vào từ bảng Receipts hay Payments.
- CloseMonth là tháng và năm đóng sổ của tháng trước.
- IssueNo là mã số phiếu thu hay chi tương ứng với cột PaymentNo hay ReceiptNo được lấy từ bảng Receipts hay Payments.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

45

- IssueBatchNo là mã số lô phiếu thu hay chi tương ứng với cột PaymentBatchNo hay ReceiptBatchNo được lấy từ bảng Receipts hay Payments.
- IssueDate ứng với ngày thu tiền hay chi tiền được lấy từ bảng Receipts hay Payments.
- PartnerID ứng với mã khách hàng khi chi tiền hay mã nhà cung cấp khi chi tiền được lấy từ bảng Receipts hay Payments.
- ReceiptAmount ứng với số tiền thu lấy từ bảng Receipts.
- PaymentAmount ứng với số tiền chi lấy từ bảng Payments.
- BalanceAmount là số tiền cân đối tại thời điểm xảy ra nghiệp vụ thu hay chi tiền.
- Description là diễn giải thu hay chi tiền lấy từ bảng Receipts hay Payments.

Chú ý: Bảng này dùng để lưu trữ thông tin tồn quỹ cho phép người sử dụng xem và in ấn trong tương lai.

2.4. Danh sách tài khoản

Như trình bày ở trong phần thu và chi tiền, mỗi khi nghiệp vụ thu hay chi tiền phát sinh thì người sử dụng cần phân bổ số tiền thu hay chi vào các tài khoản đã quy định trong quy chế kế toán.

Chú ý: Như giới thiệu ở trên, chúng ta có thể điều chỉnh cách phân bổ tài khoản trong những tập kế tiếp.

2.4.1. Bảng AccountNumbers

Bảng AccountNumbers chứa danh sách các tài khoản cấp 1 ứng với cấu trúc như sau:

```
AccountNo VARCHAR(10) NOT NULL PRIMARY KEY,
AccountNameInVietnamese NVARCHAR(50) NOT NULL,
AccountNameInSecondLanguage VARCHAR(50) NULL,
ShowAccountNameInVietnamese BIT DEFAULT 1,
ShowAccountNameInTrialBalanceBIT DEFAULT 1,
EntryDate SMALLDATETIME DEFAULT GETDATE()
```

Trong đó, cột:

- AccountNo là mã tài khoản trong hệ thống kế toán, chẳng hạn bạn nhập dữ liệu cho cột này là 111.

- AccountNameInVietnamese ứng với diễn giải cho mã tài khoản bằng tiếng Việt.
- AccountNameInSecondLanguage ứng với diễn giải cho mã tài khoản bằng ngoại ngữ khác.
- ShowAccountNameInVietnamese cho phép trình bày diễn giải của mã tài khoản bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- ShowAccountNameInTrialBalance cho phép trình bày mã tài khoản trong bản cân đối thử hay bảng tổng kết tài sản.
- EntryDate ứng với ngày khai báo tài khoản trong hệ thống kế toán.

2.4.2. Bảng SubAccountNumbers

Bảng SubAccountNumbers chứa danh sách các tài khoản cấp 2 ứng với cấu trúc như sau:

```
SubAccountNo VARCHAR (20) NOT NULL PRIMARY KEY,
AccountNo VARCHAR (10) NOT NULL,
ForwardAccountNo VARCHAR (20) NULL,
SubAccountNameInVietnamese NVARCHAR (50) NOT NULL,
SubAccountNameInSecondLanguage VARCHAR (50) NULL,
ShowSubAccountNameInVietnamese BIT DEFAULT 1,
EntryDate SMALLDATETIME DEFAULT GETDATE()
```

Trong đó, cột:

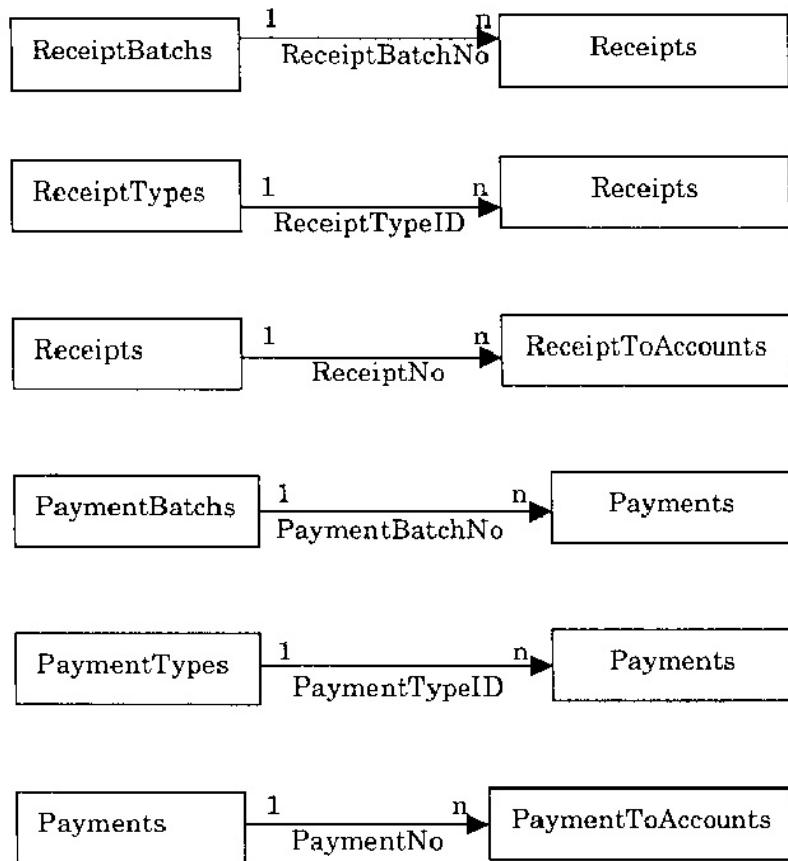
- AccountNo là mã tài khoản trong hệ thống kế toán. Chẳng hạn, bạn nhập dữ liệu cho cột này là 111.
- SubAccountNo là mã tài khoản trong hệ thống kế toán, chẳng hạn bạn nhập dữ liệu cho cột này là 111.111.
- SubAccountNameInVietnamese ứng với diễn giải cho mã tài khoản bằng tiếng Việt.
- SubAccountNameInSecondLanguage ứng với diễn giải cho mã tài khoản bằng ngoại ngữ khác.
- ShowSubAccountNameInVietnamese cho phép trình bày diễn giải của mã tài khoản bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- EntryDate ứng với ngày khai báo tài khoản trong hệ thống kế toán.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

47

2.5. Quan hệ

Trong đó, quan hệ giữa các bảng của phần kế toán tổng hợp được trình bày như hình 2-2.

**Hình 2-2: Quan hệ giữa bảng dữ liệu phần GL.**

Chú ý: Bạn có thể tìm hiểu cách tạo quan hệ giữa các bảng dữ liệu trong chương 4.

3. CẤU TRÚC CƠ SỞ DỮ LIỆU PHẦN ACCOUNT RECEIVABLE

Để cho phép lưu trữ thông tin về hóa đơn bán hàng hóa hay dịch vụ, chúng ta cần xây dựng các bảng dữ liệu chính và bảng dữ liệu liên quan đến phần này, chúng bao gồm các nhóm bảng dữ liệu như sau:

Phần thông tin khách hàng bao gồm các bảng: CustomerTypes, Customers, MonthlyAccountReceivable và CloseAccountReceivable.

Phần hóa đơn bán hàng bao gồm các bảng: SalesInvoiceTypes, SalesInvoiceBatchs, SalesPrices, SalesInvoices và SalesInvoiceDetails.

Chú ý: Trong ứng dụng dính kèm không đề cập đến phần quản lý hợp đồng mua hàng, bạn có thể tìm hiểu các phần này trong tập sách Xây dựng ứng dụng kế toán bằng SQL Server 2005 sẽ được xuất bản trong thời gian tới.

3.1. Khách hàng

3.1.1. Bảng CustomerTypes

Bảng CustomerTypes dùng để lưu trữ loại khách hàng với cấu trúc như sau:

```
CustomerTypeID CHAR(3) NOT NULL PRIMARY KEY,  
CustomerTypeName NVARCHAR(50) NOT NULL
```

Trong đó, cột:

- CustomerTypeID là mã số loại khách hàng.
- CustomerTypeName ứng với diễn giải loại khách hàng.

3.1.2. Bảng Customers

Bảng Customers dùng để lưu trữ thông tin về khách hàng, nó bao gồm các cột dữ liệu với cấu trúc như sau:

```
CustomerID CHAR(5) NOT NULL PRIMARY KEY,  
CompanyNameInVietnamese NVARCHAR(50) NOT NULL,  
CompanyNameInSecondLanguage VARCHAR(50) NOT NULL,  
ShowCompanyNameInVietnamese BIT DEFAULT 1,  
ContactName NVARCHAR(50) NOT NULL,  
ContactTitle NVARCHAR(50) NOT NULL,  
Address NVARCHAR(100) NULL,  
ProvinceID CHAR(3) NOT NULL,  
Telephone VARCHAR(20) NULL,  
FaxNumber VARCHAR(10) NULL,  
CustomerTypeID CHAR(3),  
EmailAddress VARCHAR(50) NULL,  
DueDate SMALLDATETIME DEFAULT GETDATE(),  
Discontinued BIT DEFAULT 0
```

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

49

Trong đó, cột:

- CustomerID là mã khách hàng, bạn có thể viết đoạn chương trình trong SQL Server hay trong ngôn ngữ lập trình ứng dụng để tạo ra mã bao gồm 5 ký tự theo quy định nào đó.
- CompanyNameInVietnamese ứng với tên của công ty bằng tiếng Việt.
- CompanyNameInSecondLanguage ứng với tên của công ty bằng ngoại ngữ khác.
- ShowCompanyNameInVietnamese cho phép trình bày tên của công ty bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- ContactName ứng với tên của người đại diện khách hàng là công ty khi liên hệ.
- ContactTitle ứng với chức vụ của người đại diện khách hàng là công ty khi liên hệ.
- Address là địa chỉ của khách hàng.
- ProvinceID ứng với mã tỉnh thành của khách hàng.
- Telephone ứng với số điện thoại của khách hàng.
- FaxNumber ứng với số Fax của khách hàng.
- CustomerTypeID ứng với mã loại khách hàng.
- EmailAddress ứng với địa chỉ email của khách hàng.
- DueDate ứng với ngày nhập thông tin khách hàng, cột này tự động nhập ngày mà mẫu tin được thêm vào bảng dữ liệu, không cho phép người sử dụng thay đổi giá trị cho cột này trên giao diện ứng dụng.
- Discontinued ứng với trạng thái thông tin của khách hàng đóng hay đang còn mở cho phép giao thương.

3.1.3. Bảng MonthlyAccountReceivable

Bảng MonthlyAccountReceivable dùng để lưu trữ công nợ phải thu cuối tháng, chúng có cấu trúc như sau:

```
CustomerID CHAR(5) NOT NULL PRIMARY KEY,
BeginAmount DECIMAL DEFAULT 0,
SalesAmount DECIMAL DEFAULT 0,
ReceiveAmount DECIMAL DEFAULT 0,
```

```
EndAmount DECIMAL DEFAULT 0
```

Trong đó, cột:

- CustomerID ứng với mã khách hàng.
- BeginAmount là công nợ phải thu đầu kỳ.
- SalesAmount là tổng số tiền bán trong kỳ.
- ReceiveAmount là tổng số tiền thu trong kỳ.
- EndAmount là công nợ phải thu cuối kỳ.

3.1.4. Bảng CloseAccountReceivable

Bảng CloseAccountReceivable dùng để lưu trữ công nợ phải thu khi đóng sổ kế toán, nó bao gồm các cột dữ liệu như sau:

```
CloseMonth CHAR(7) NOT NULL ,
CloseDate SMALLDATETIME DEFAULT GETDATE() ,
CloseMonthUserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- CloseMonth ứng với tháng đóng sổ.
- CloseDate là ngày đóng sổ.
- CloseMonthUserName ứng với tên người sử dụng thực hiện tác vụ đóng sổ.

3.1.5. Bảng CloseAccountReceivableDetails

Bảng CloseAccountReceivableDetails dùng để lưu trữ công nợ phải thu chi tiết khi đóng sổ kế toán, nó bao gồm các cột dữ liệu với cấu trúc như sau:

```
CustomerID CHAR(5) NOT NULL ,
CloseMonth CHAR(7) NOT NULL ,
BeginAmount DECIMAL DEFAULT 0 ,
SalesAmount DECIMAL DEFAULT 0 ,
ReceiveAmount DECIMAL DEFAULT 0 ,
EndAmount DECIMAL DEFAULT 0 ,
PRIMARY KEY (CustomerID, CloseMonth)
```

Trong đó, cột:

- CustomerID ứng với mã khách hàng.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

51

- CloseMonth ứng với tháng đóng sổ.
- BeginAmount là công nợ phải thu đầu kỳ.
- SalesAmount là tổng số tiền bán hàng trong kỳ.
- ReceiveAmount là tổng số tiền thu của khách hàng trong kỳ.
- EndAmount là công nợ phải thu cuối kỳ.

Chú ý: Bản này dùng lưu trữ thông tin công nợ phải thu của khách hàng ứng với những tháng trước đó, khi bạn tính công nợ phải thu của tháng thứ N thì số tiền của cột BeginAmount là EndAmount trong tháng N-1.

3.2. Hóa đơn bán hàng

3.2.1. *Bảng SalesInvoiceTypes*

Bảng SalesInvoiceTypes dùng để lưu loại hóa đơn bán hàng bao gồm các cột dữ liệu với cấu trúc như sau:

```
InvoiceTypeID CHAR(3) NOT NULL PRIMARY KEY,
InvoiceTypeNameInVietnamese NVARCHAR(50) NOT NULL,
InvoiceTypeNameInSecondLanguage VARCHAR(50) NOT NULL,
ShowInvoiceTypeInVietnamese BIT NOT NULL DEFAULT 1
```

Trong đó, cột:

- InvoiceTypeID là mã số loại hóa đơn bán hàng.
- InvoiceTypeNameInVietnamese ứng với diễn giải loại hóa đơn bán hàng bằng tiếng Việt.
- InvoiceTypeNameInSecondLanguage ứng với diễn giải loại hóa đơn bán hàng bằng ngoại ngữ khác.
- ShowInvoiceTypeInVietnamese cho phép trình bày loại hóa đơn bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.

3.2.2. *Bảng SalesInvoiceBatches*

Bảng SalesInvoiceBatches dùng để lưu lô (batch) bao gồm nhiều hóa đơn bán hàng ứng với cấu trúc có các cột dữ liệu như sau:

```
InvoiceBatchNo VARCHAR(10) NOT NULL PRIMARY KEY,
InvoiceBatchDate SMALLDATETIME
    DEFAULT GETDATE()
```

```
BatchDiscontinued BIT DEFAULT 0 ,
EntryDate SMALLDATETIME DEFAULT GETDATE() ,
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- InvoiceBatchNo là mã số lô hóa đơn bán hàng.
- InvoiceBatchDate ứng với ngày nhập lô hóa đơn bán hàng.
- BatchDiscontinued ứng với trạng thái lô hóa đơn bán hàng đóng hay đang còn mở cho phép nhân viên nhập hóa đơn.
- EntryDate là ngày tạo ra lô hóa đơn bán hàng.
- UserName ứng với tài khoản của nhân viên tạo lô hóa đơn bán hàng.

3.2.3. *Bảng SalesPrices*

Bảng SalesPrices dùng để lưu trữ thông tin giá bán của sản phẩm theo thời gian, chúng bao gồm các cột dữ liệu với cấu trúc như sau:

```
DateOfPrice SMALLDATETIME
NOT NULL DEFAULT GETDATE() ,
ProductID VARCHAR(10) NOT NULL,
Price DECIMAL DEFAULT 0,
CurrencyID CHAR(3) NOT NULL,
PriceDiscontinued BIT DEFAULT 0 ,
UserName VARCHAR(10) NOT NULL
PRIMARY KEY (DateOfPrice, ProductID)
```

Trong đó, cột:

- DateOfPrice là ngày thông báo giá bán sản phẩm.
- ProductID ứng với mã sản phẩm.
- Price là giá bán theo hóa đơn.
- CurrencyID ứng với mã tiền tệ.
- PriceDiscontinued ứng với trạng thái giá của sản phẩm đóng hay đang còn mở cho phép nhân viên sử dụng giá bán trong hóa đơn bán hàng.
- UserName ứng với tài khoản của nhân viên khai báo giá bán hàng.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

53

3.2.4. *Bảng SalesInvoices*

Bảng SalesInvoices dùng để lưu trữ thông tin hóa đơn bán hàng bao gồm các cột dữ liệu với cấu trúc như sau:

```
InvoiceNo VARCHAR(10) NOT NULL PRIMARY KEY,  
InvoiceBatchNo VARCHAR(10) NULL,  
DueDate SMALLDATETIME DEFAULT GETDATE(),  
InvoiceTypeID CHAR(3) NOT NULL,  
CustomerID CHAR(5) NOT NULL ,  
CurrencyID CHAR(3) NOT NULL,  
ExchangeRate DECIMAL DEFAULT 0 ,  
DescriptionInVietnamese NVARCHAR(150)NOT NULL ,  
DescriptionInSecondLanguage VARCHAR(150) NULL ,  
ShowDescriptionInVietnamese BIT DEFAULT 1 ,  
InvoiceDiscontinued BIT DEFAULT 0 ,  
EntryDate SMALLDATETIME DEFAULT GETDATE() ,  
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- InvoiceNo ứng với mã số hóa đơn bán hàng.
- InvoiceBatchNo ứng với mã số lô hóa đơn bán hàng.
- DueDate ứng với ngày xuất hóa đơn bán hàng.
- InvoiceTypeID là mã loại hóa đơn bán hàng.
- CustomerID ứng với mã khách hàng mua hàng.
- CurrencyID ứng với mã tiền tệ.
- ExchangeRate ứng với tỷ giá của tiền tệ khai báo.
- DescriptionInVietnamese ứng với diễn giải hóa đơn bán hàng bằng tiếng Việt.
- DescriptionInSecondLanguage ứng với diễn giải hóa đơn bán hàng bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải hóa đơn bán hàng bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.

- InvoiceDiscontinued ứng với trạng thái hóa đơn đóng hay đang còn mở cho phép nhân viên cập nhật thông tin hóa đơn.
- EntryDate là ngày nhập hóa đơn bán hàng.
- UserName là tài khoản người nhập.

3.2.5. *Bảng SalesInvoiceDetails*

Bảng SalesInvoiceDetails dùng để lưu trữ thông tin hóa đơn bán hàng chi tiết, nó bao gồm các cột dữ liệu như sau:

```
OrdinalNumber tinyint NOT NULL,  
InvoiceNo VARCHAR(10) NOT NULL,  
ProductID VARCHAR(10) NOT NULL,  
Quantity DECIMAL DEFAULT 0,  
Price DECIMAL DEFAULT 0,  
Discount DECIMAL DEFAULT 0,  
VATRate DECIMAL DEFAULT 0,  
PRIMARY KEY  
(OrdinalNumber, InvoiceNo, ProductID)
```

Trong đó, cột:

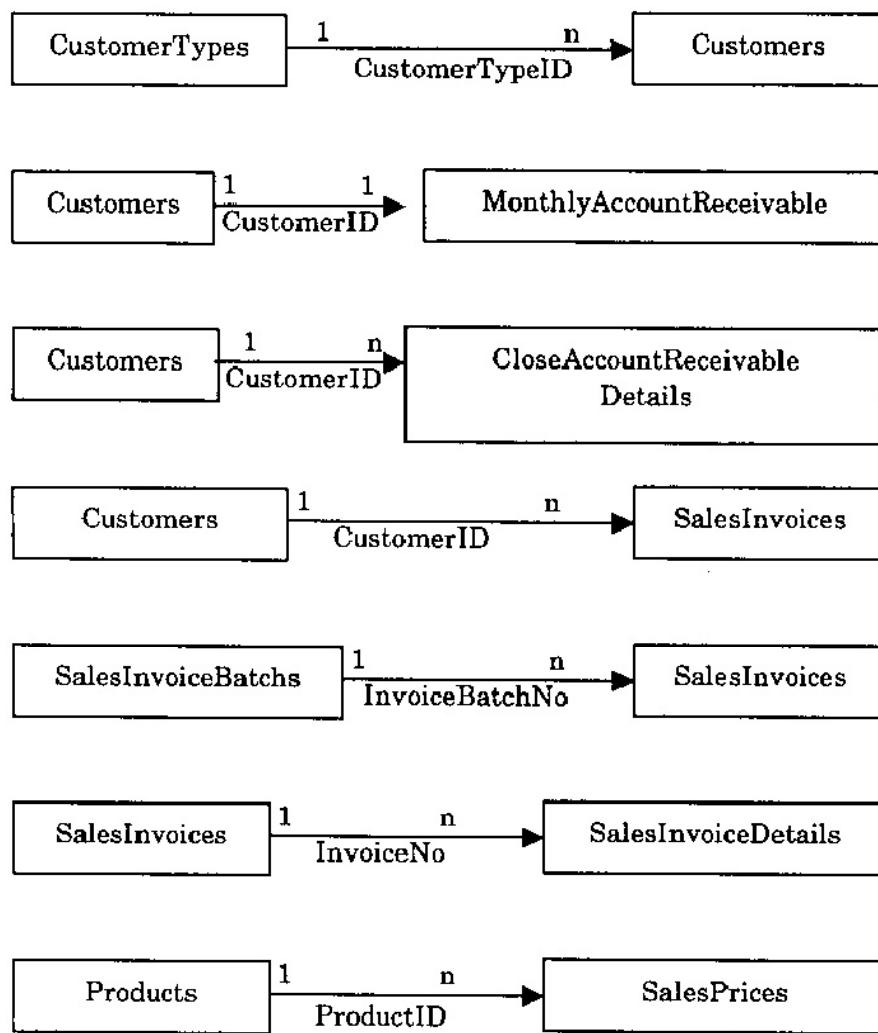
- OrdinalNumber là số thứ tự của sản phẩm ghi trong phần chi tiết hóa đơn.
- InvoiceNo ứng với mã số hóa đơn bán hàng.
- ProductID ứng với mã sản phẩm.
- Quantity là số lượng bán.
- Price là giá bán theo hóa đơn.
- Discount là số tiền giảm giá.
- VATRate là thuế suất VAT.

3.3. Quan hệ

Quan hệ giữa các bảng thuộc phần kế toán công nợ phải thu được trình bày như hình 2-3.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

55

**Hình 2-3: Quan hệ giữa bảng dữ liệu phần AR.**

4. CẤU TRÚC CƠ SỞ DỮ LIỆU PHẦN ACCOUNT PAYABLE

Tương tự như trường hợp kế toán công nợ phải thu, để cho phép lưu trữ thông tin về hóa đơn mua hàng hóa hay dịch vụ từ nhà cung cấp, chúng

ta cần xây dựng các bảng dữ liệu chính và bảng dữ liệu liên quan đến phần này, chúng bao gồm các nhóm bảng dữ liệu như sau:

Phần thông tin nhà cung cấp bao gồm các bảng: SupplierTypes, Suppliers, MonthlyAccountPayable và CloseAccountPayable.

Phần hóa đơn bán hàng bao gồm các bảng: PurchaseInvoiceTypes, PurchaseInvoiceBatchs, PurchaseInvoices và PurchaseInvoiceDetails.

4.1. Nhà cung cấp

4.1.1. *Bảng SupplierTypes*

Bảng SupplierTypes dùng để lưu trữ loại nhà cung cấp với cấu trúc như sau:

```
SupplierTypeID CHAR(3) NOT NULL PRIMARY KEY,
SupplierTypeName NVARCHAR(50) NOT NULL
```

Trong đó, cột:

- SupplierTypeID là mã số loại nhà cung cấp.
- SupplierTypeName ứng với diễn giải loại nhà cung cấp.

4.1.2. *Bảng Suppliers*

Bảng Suppliers dùng để lưu trữ thông tin về nhà cung cấp, nó bao gồm các cột dữ liệu với cấu trúc như sau:

```
SupplierID CHAR(5) NOT NULL PRIMARY KEY,
CompanyNameInVietnamese NVARCHAR(50) NOT NULL,
CompanyNameInSecondLanguage VARCHAR(50) NOT NULL,
ShowCompanyNameInVietnamese BIT DEFAULT 1,
ContactName NVARCHAR(50) NOT NULL,
ContactTitle NVARCHAR(50) NOT NULL,
Address NVARCHAR(100) NULL,
ProvinceID CHAR(3) NOT NULL,
Telephone VARCHAR(20) NULL,
FaxNumber VARCHAR(10) NULL,
SupplierTypeID CHAR(3),
EmailAddress VARCHAR(50) NULL,
DueDate SMALLDATETIME DEFAULT GETDATE(),
Discontinued BIT DEFAULT 0
```

Trong đó, cột:

- SupplierID là mã nhà cung cấp, bạn có thể viết đoạn chương trình trong SQL Server hay trong ngôn ngữ lập trình ứng dụng để tạo ra mã bao gồm 5 ký tự theo quy định nào đó.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

57

- CompanyNameInVietnamese ứng với tên của công ty bằng tiếng Việt.
- CompanyNameInSecondLanguage ứng với tên của công ty bằng ngoại ngữ khác.
- ShowCompanyNameInVietnamese cho phép trình bày tên của công ty bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- ContactName ứng với tên của người đại diện nhà cung cấp là công ty khi liên hệ.
- ContactTitle ứng với chức vụ của người đại diện nhà cung cấp là công ty khi liên hệ.
- Address là địa chỉ của nhà cung cấp.
- ProvinceID ứng với mã tỉnh thành của nhà cung cấp.
- Telephone ứng với số điện thoại của nhà cung cấp.
- FaxNumber ứng với số Fax của nhà cung cấp.
- SupplierTypeID ứng với mã loại nhà cung cấp.
- EmailAddress ứng với địa chỉ email của nhà cung cấp.
- DueDate ứng với ngày nhập thông tin nhà cung cấp, cột này tự động nhập ngày mà mẫu tin được thêm vào bảng dữ liệu, không cho phép người sử dụng thay đổi giá trị cho cột này trên giao diện ứng dụng.
- Discontinued ứng với trạng thái thông tin của nhà cung cấp đóng hay đang còn mở cho phép giao thương.

4.1.3. Bảng MonthlyAccountPayable

Bảng MonthlyAccountPayable dùng để lưu trữ công nợ phải chi cuối tháng với cấu trúc như sau:

```
SupplierID CHAR(5) NOT NULL PRIMARY KEY,
BeginAmount DECIMAL DEFAULT 0,
PurchaseAmount DECIMAL DEFAULT 0,
PaidAmount DECIMAL DEFAULT 0,
EndAmount DECIMAL DEFAULT 0
```

Trong đó, cột:

- SupplierID ứng với mã nhà cung cấp.
- BeginAmount là công nợ chi đầu kỳ.

- PurchaseAmount là tổng số tiền mua trong kỳ.
- PaidAmount là tổng số tiền trả trong kỳ.
- EndAmount là công nợ chi cuối kỳ.

4.1.4. Bảng CloseAccountPayable

Bảng CloseAccountPayable dùng để lưu trữ công nợ phải chi khi đóng sổ kế toán, nó bao gồm các cột dữ liệu với cấu trúc như sau:

```
CloseMonth CHAR (7) NOT NULL PRIMARY KEY,
CloseDate SMALLDATETIME DEFAULT GETDATE(),
CloseMonthUserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- CloseMonth ứng với tháng đóng sổ.
- CloseDate là ngày đóng sổ.
- CloseMonthUserName ứng với tên người sử dụng thực hiện tác vụ đóng sổ.

4.1.5. Bảng CloseAccountPayableDetails

Bảng CloseAccountPayableDetails dùng để lưu trữ công nợ phải chi khi đóng sổ kế toán, nó bao gồm các cột dữ liệu với cấu trúc như sau:

```
SupplierID CHAR (5) NOT NULL,
CloseMonth CHAR (7) NOT NULL ,
BeginAmount DECIMAL DEFAULT 0,
PurchaseAmount DECIMAL DEFAULT 0,
PaidAmount DECIMAL DEFAULT 0,
EndAmount DECIMAL DEFAULT 0,
PRIMARY KEY (SupplierID, CloseMonth)
```

Trong đó, cột:

- SupplierID ứng với mã nhà cung cấp.
- CloseMonth ứng với tháng đóng sổ.
- BeginAmount là công nợ chi đầu kỳ.
- PurchaseAmount là tổng số tiền mua trong kỳ.
- PaidAmount là tổng số tiền trả trong kỳ.
- EndAmount là công nợ chi cuối kỳ.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

59

Chú ý: Bảng này dùng lưu trữ thông tin công nợ phải trả cho nhiều nhà cung cấp ứng với nhiều tháng trước đó, khi bạn tính công nợ phải trả của tháng thứ N thì số tiền của cột BeginAmount chính là EndAmount trong tháng N-1.

4.2. Hóa đơn mua hàng

4.2.1. *Bảng PurchaseInvoiceTypes*

Bảng PurchaseInvoiceTypes dùng để lưu loại hóa đơn mua hàng, bao gồm các cột dữ liệu với cấu trúc như sau:

```
InvoiceTypeID CHAR(3) NOT NULL PRIMARY KEY,
InvoiceTypeName NVARCHAR(50) NOT NULL,
InvoiceTypeNameInSecondLanguage VARCHAR(50) NOT NULL
ShowInvoiceTypeInVietnamese BIT NOT NULL DEFAULT 1
```

Trong đó, cột:

- InvoiceTypeID là mã số loại hóa đơn mua hàng.
- InvoiceTypeName ứng với diễn giải loại hóa đơn mua hàng.
- InvoiceTypeNameInSecondLanguage ứng với diễn giải loại hóa đơn mua hàng bằng ngoại ngữ khác.
- ShowInvoiceTypeInVietnamese cho phép trình bày loại hóa đơn mua hàng bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.

4.2.2. *Bảng PurchaseInvoiceBatchs*

Bảng PurchaseInvoiceBatchs dùng để chứa lô (batch) nhiều hóa đơn mua hàng, bao gồm các cột dữ liệu với cấu trúc như sau:

```
InvoiceBatchNo VARCHAR(10)
NOT NULL PRIMARY KEY,
InvoiceBatchDate SMALLDATETIME
DEFAULT GETDATE(),
BatchDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- InvoiceBatchNo là mã số lô hóa đơn mua hàng.
- InvoiceBatchDate ứng với ngày nhập lô hóa đơn mua hàng.

- BatchDiscontinued ứng với trạng thái lô hóa đơn mua hàng đóng hay đang còn mở cho phép nhập hóa đơn.
- UserName ứng với tài khoản nhân viên tạo lô hóa đơn mua hàng.

4.2.3. *Bảng PurchaseInvoices*

Bảng PurchaseInvoices dùng để lưu trữ thông tin hóa đơn mua hàng, bao gồm các cột dữ liệu với cấu trúc như sau:

```
TInvoiceNo VARCHAR(10) NOT NULL PRIMARY KEY,
InvoiceBatchNo VARCHAR(10) NULL,
DueDate SMALLDATETIME DEFAULT GETDATE(),
InvoiceTypeID CHAR(3) NOT NULL,
SupplierID CHAR(5) NOT NULL ,
CurrencyID CHAR(3) NOT NULL,
ExchangeRate DECIMAL DEFAULT 0,
DescriptionInVietnamese NVARCHAR(150) NOT NULL,
DescriptionInSecondLanguage VARCHAR(150) NULL,
ShowDescriptionInVietnamese BIT DEFAULT 1,
InvoiceDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- InvoiceNo ứng với mã số hóa đơn mua hàng.
- InvoiceBatchNo ứng với mã số lô hóa đơn mua hàng.
- DueDate ứng với ngày xuất hóa đơn mua hàng.
- InvoiceTypeID là mã loại hóa đơn xuất hàng.
- SupplierID ứng với mã nhà cung cấp bán hàng.
- CurrencyID ứng với mã tiền tệ.
- ExchangeRate ứng với tỷ giá của tiền tệ khai báo.
- DescriptionInVietnamese ứng với diễn giải hóa đơn mua hàng bằng tiếng Việt.
- DescriptionInSecondLanguage ứng với diễn giải hóa đơn mua hàng bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải hóa đơn mua hàng bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

61

- InvoiceDiscontinued ứng với trạng thái hóa đơn đóng hay đang còn mở cho phép cập nhật thông tin hóa đơn.
- EntryDate là ngày nhập hóa đơn xuất hàng.
- UserName là tài khoản người nhập liệu.

4.2.4. Bảng PurchaseInvoiceDetails

Bảng PurchaseInvoiceDetails dùng để lưu trữ thông tin hóa đơn mua hàng chi tiết, nó bao gồm các cột dữ liệu như sau:

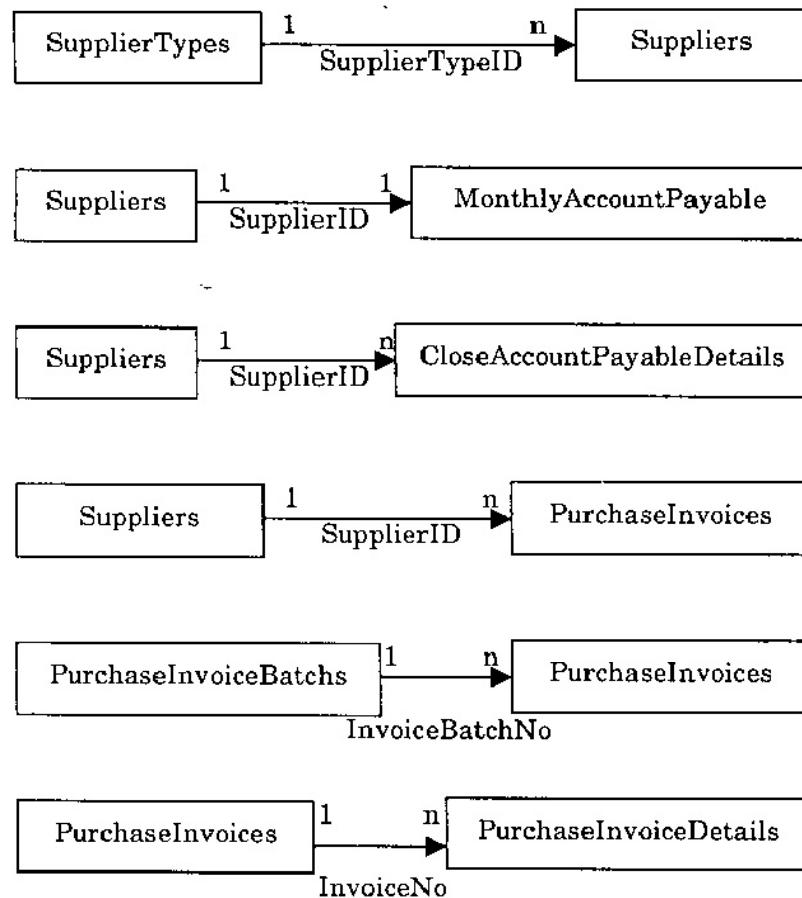
```
OrdinalNumber tinyint NOT NULL,
InvoiceNo VARCHAR(10) NOT NULL,
ProductID VARCHAR(10) NOT NULL,
Quantity DECIMAL DEFAULT 0,
Price DECIMAL DEFAULT 0,
Discount DECIMAL DEFAULT 0,
VATRate DECIMAL DEFAULT 0,
PRIMARY KEY (OrdinalNumber, InvoiceNo, ProductID)
```

Trong đó, cột:

- OrdinalNumber là số thứ tự của sản phẩm ghi trong hóa đơn chi tiết.
- InvoiceNo ứng với mã số hóa đơn mua hàng.
- ProductID ứng với mã sản phẩm.
- Quantity là số lượng bán.
- Price là giá mua theo hóa đơn.
- Discount là số tiền giảm giá.
- VATRate là thuế suất VAT.

4.3. Quan hệ

Tương tự như phần công nợ phải thu, quan hệ giữa các bảng của phần công nợ phải trả được trình bày như hình 2-4.

M® 62**Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm****Hình 2-4: Quan hệ giữa các bảng dữ liệu AP.**

5. CẤU TRÚC CƠ SỞ DỮ LIỆU PHẦN INVENTORY CONTROL

Đối với phần quản lý tồn kho, để cho phép lưu trữ thông tin về phiếu xuất hay nhập hàng hóa, chúng ta cần xây dựng các bảng dữ liệu chính và bảng dữ liệu liên quan đến phần này, chúng bao gồm các nhóm bảng dữ liệu như sau:

Phần thông tin sản phẩm và báo cáo tồn kho bao gồm các bảng: Categories, Products, MonthlyInventoryControl và CloseInventoryControl.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

63

Chú ý: Do chúng ta cho phép quản lý tồn kho trên nhiều kho, nên bạn cần khai báo thêm hai bảng dữ liệu Stocks và ProductInStocks.

5.1. Phần tồn kho

5.1.1. *Bảng Categories*

Bảng Categories dùng để lưu trữ danh sách loại sản phẩm với cấu trúc như sau:

```
CategoryID      INT IDENTITY(1,1)
                NOT NULL PRIMARY KEY,
CategoryName   NVARCHAR(50) NOT NULL,
Discontinued  BIT DEFAULT 0
```

Trong đó, cột:

- CategoryID là mã số loại sản phẩm.
- CategoryName ứng với diễn giải loại sản phẩm.
- Discontinued ứng với trạng thái loại sản phẩm đóng hay đang còn mở cho phép cập nhật thông tin liên quan đến loại sản phẩm.

5.1.2. *Bảng Products*

Bảng Products dùng để lưu trữ danh sách sản phẩm với cấu trúc như sau:

```
ProductID VARCHAR(10) NOT NULL PRIMARY KEY,
CategoryID      INT NOT NULL,
ProductNameInVietnamese NVARCHAR(150) NOT NULL,
ProductNameInSecondLanguage VARCHAR(150) NOT NULL,
ShowProductNameInVietnamese BIT DEFAULT 1,
ProductUnit NVARCHAR(20) NOT NULL,
ProductDescriptionInVietnamcse NVARCHAR(max) NULL,
ProductDescriptionInSecondLanguage  VARCHAR(max) NULL,
DueDate SMALLDATETIME DEFAULT GETDATE(),
Discontinued BIT DEFAULT 0
```

Trong đó, cột:

- ProductID là mã số sản phẩm.
- CategoryID là mã số loại sản phẩm.
- ProductNameInVietnamese ứng với tên loại sản phẩm bằng tiếng Việt.
- ProductNameInSecondLanguage ứng với tên loại sản phẩm bằng thứ tiếng của nhà sản xuất.

- ShowProductNameInVietnamese cho phép trình bày tên sản phẩm bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- ProductUnit là đơn vị sản phẩm.
- ProductDescriptionInVietnamese ứng với diễn giải sản phẩm bằng tiếng Việt.
- ProductDescriptionInSecondLanguage ứng với diễn giải sản phẩm bằng tiếng của nhà sản xuất.
- DueDate là ngày bắt đầu kinh doanh sản phẩm.
- Discontinued ứng với trạng thái loại sản phẩm đóng hay đang còn mở cho phép cập nhật thông tin liên quan đến sản phẩm.

5.1.3. Bảng MonthlyInventoryControl

Bảng MonthlyInventoryControl dùng để lưu trữ tình hình tồn kho cuối tháng với cấu trúc như sau:

```
ProductID VARCHAR(10) NOT NULL,
StockID CHAR(5) NOT NULL,
BeginQuantity DECIMAL DEFAULT 0,
ImportQuantity DECIMAL DEFAULT 0,
ExportQuantity DECIMAL DEFAULT 0,
EndQuantity DECIMAL DEFAULT 0
PRIMARY KEY (ProductID, StockID)
```

Trong đó, cột:

- ProductID ứng với mã khách hàng.
- StockID là mã số kho hàng.
- BeginQuantity là số tồn kho đầu kỳ.
- ImportQuantity là tổng số nhập trong kỳ.
- ExportQuantity là tổng số xuất trong kỳ.
- EndQuantity là số tồn kho cuối kỳ.

5.1.4. Bảng CloseInventoryControl

Bảng CloseInventoryControl dùng để lưu trữ tình hình tồn kho cuối kỳ khi đóng sổ kế toán, nó bao gồm các cột dữ liệu với cấu trúc như sau:

```
CloseMonth CHAR(7) NOT NULL,
ProductID VARCHAR(15) NOT NULL,
```

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm65 

```

StockID CHAR(5) NOT NULL,
BeginQuantity DECIMAL DEFAULT 0,
ImportQuantity DECIMAL DEFAULT 0,
ExportQuantity DECIMAL DEFAULT 0,
EndQuantity DECIMAL DEFAULT 0,
CloseMonthUserName VARCHAR(10) NOT NULL,
PRIMARY KEY (ProductID, CloseMonth, StockID)

```

Trong đó, cột:

- CloseMonth ứng với tháng đóng sổ.
- ProductID ứng với mã sản phẩm.
- StockID là mã số kho hàng.
- BeginQuantity là số tồn kho đầu kỳ.
- ImportQuantity là tổng số nhập hàng trong kỳ.
- ExportQuantity là tổng số xuất hàng trong kỳ.
- EndQuantity là số tồn kho cuối kỳ.
- CloseMonthUserName ứng với tên người sử dụng thực hiện tác vụ đóng sổ.

5.1.5. Bảng Stocks

Bảng Stocks dùng để lưu trữ danh sách tên kho hàng với cấu trúc như sau:

```

StockID CHAR(5) NOT NULL PRIMARY KEY,
StockName NVARCHAR(50) NOT NULL,
StockAddress NVARCHAR(50) NULL,
Discontinued BIT DEFAULT 0

```

Trong đó, cột:

- StockID là mã số kho hàng.
- StockName ứng với tên kho hàng.
- StockAddress ứng với địa chỉ kho hàng.
- Discontinued ứng với trạng thái kho hàng đóng hay đang còn mở cho phép cập nhật thông tin liên quan đến kho hàng.

5.1.6. Bảng ProductInStocks

Bảng ProductInStocks dùng để lưu trữ danh sách mã sản phẩm ứng với mã kho cùng với số lượng hiện hành của nó với cấu trúc như sau:

```

ProductID VARCHAR(10) NOT NULL,
StockID CHAR(5) NOT NULL ,
GoodProductInStock DECIMAL DEFAULT 0 ,
BadProductInStock DECIMAL DEFAULT 0
PRIMARY KEY (ProductID, StockID)

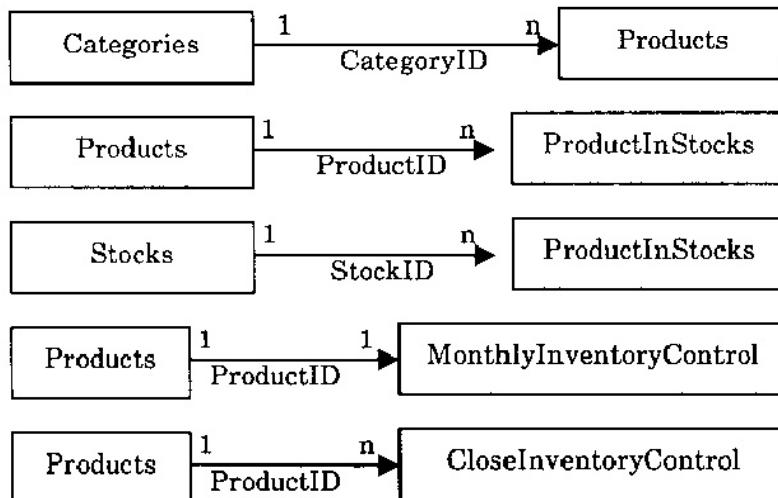
```

Trong đó, cột:

- ProductID là mã số sản phẩm.
- StockID là mã số kho hàng.
- GoodProductInStock ứng với số lượng sản phẩm có thể giao dịch đang có trong kho.
- BadProductInStock ứng với số lượng sản phẩm không thể giao dịch đang có trong kho.

5.1.7. Quan hệ

Bạn có thể tạo quan hệ giữa các bảng của phần sản phẩm và tồn kho được trình bày như hình 2-5.



Hình 2-5: Quan hệ sản phẩm với báo cáo tồn kho.

Chú ý: Bản này dùng lưu trữ thông tin tồn kho cho mỗi sản phẩm của nhiều tháng trước đó, khi bạn tính tồn kho của tháng thứ N thì số liệu của cột BeginQuantity là EndQuantity trong tháng N-1.

5.2. Phần xuất hàng.

Đối với phần xuất hàng bao gồm các bảng: ExportTypes, ExportBatchs, Exports và ExportDetails.

5.2.1. Bảng ExportTypes

Bảng ExportTypes dùng để lưu trữ danh sách loại phiếu xuất hàng với cấu trúc như sau:

```
ExportTypeID CHAR(3) NOT NULL PRIMARY KEY,
ExportTypeName NVARCHAR(50) NOT NULL
```

Trong đó, cột:

- ExportTypeID là mã số phiếu xuất hàng.
- ExportTypeName ứng với tên phiếu xuất hàng.

5.2.2. Bảng ExportBatchs

Bảng ExportBatchs dùng để lưu trữ danh sách lô phiếu xuất hàng với cấu trúc như sau:

```
ExportBatchNo VARCHAR(10) NOT NULL PRIMARY KEY,
ExportBatchDate SMALLDATETIME DEFAULT GETDATE(),
ExportDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- ExportBatchNo là mã số lô phiếu xuất hàng.
- ExportBatchDate ứng với ngày nhập lô phiếu xuất hàng.
- ExportDiscontinued ứng với trạng thái lô phiếu xuất hàng đóng hay đang còn mở cho phép nhập phiếu xuất.
- EntryDate là ngày tạo lô phiếu xuất kho.
- UserName ứng với tài khoản nhân viên tạo lô phiếu xuất hàng.

5.2.3. Bảng Exports

Bảng Exports dùng để lưu trữ thông tin phiếu xuất hàng bao gồm các cột dữ liệu với cấu trúc như sau:

```
ExportNo VARCHAR(10) NOT NULL PRIMARY KEY,
ExportBatchNo VARCHAR(10) NULL,
ExportDate SMALLDATETIME DEFAULT GETDATE(),
ExportTypeID CHAR(3) NOT NULL,
CustomerID CHAR(5) NOT NULL,
```

```
DescriptionInVietnamese NVARCHAR(150) NOT NULL,
DescriptionInSecondLanguage VARCHAR(150) NOT NULL,
ShowDescriptionInVietnamese BIT DEFAULT 1,
ExportDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- ExportNo ứng với mã số phiếu xuất hàng.
- ExportBatchNo ứng với mã số lô phiếu xuất hàng.
- ExportDate ứng với ngày xuất hàng.
- ExportTypeID là mã loại phiếu xuất.
- CustomerID ứng với mã khách hàng.
- DescriptionInVietnamese ứng với diễn giải phiếu xuất hàng bằng tiếng Việt.
- DescriptionInSecondLanguage ứng với diễn giải phiếu xuất hàng bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải phiếu xuất hàng bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- ExportDiscontinued ứng với trạng thái phiếu xuất hàng đóng hay đang còn mở cho phép cập nhật thông tin phiếu xuất.
- EntryDate là ngày nhập phiếu xuất hàng.
- UserName là tài khoản người nhập liệu.

5.2.4. Bảng ExportDetails

Bảng ExportDetails dùng để lưu trữ thông tin phiếu xuất hàng chi tiết, nó bao gồm các cột dữ liệu như sau:

```
OrdinalNumber tinyint NOT NULL,
ExportNo VARCHAR(10) NOT NULL,
ProductID VARCHAR(10) NOT NULL,
Quantity DECIMAL DEFAULT 0,
StockID CHAR(5) NOT NULL
PRIMARY KEY (OrdinalNumber, ExportNo, ProductID, StockID)
```

Trong đó, cột:

- OrdinalNumber là số thứ tự của sản phẩm ghi trong phiếu xuất chi tiết.

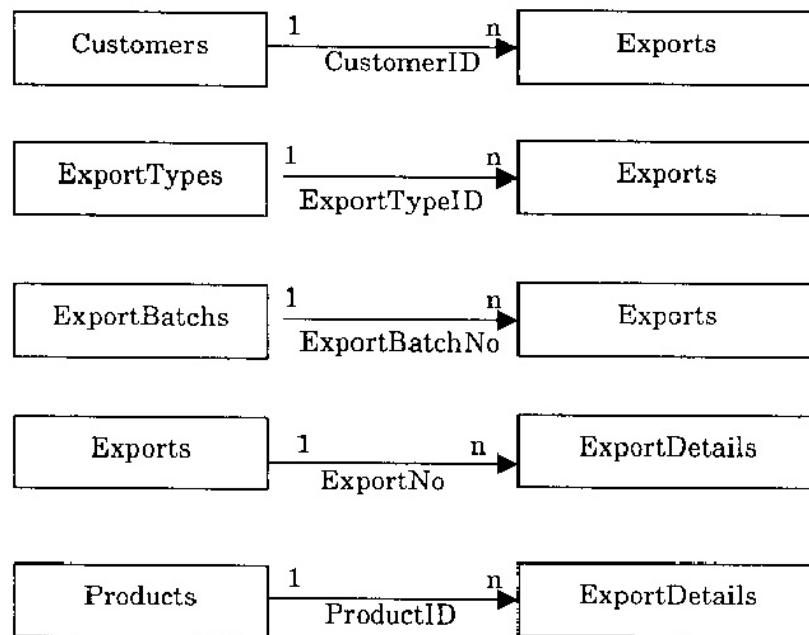
Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

69

- ExportNo ứng với mã số phiếu xuất hàng.
- ProductID ứng với mã sản phẩm.
- Quantity là số lượng xuất.
- StockID là mã số kho hàng.

5.2.5. Quan hệ

Bạn có thể tạo quan hệ giữa các bảng của phần xuất hàng được trình bày như hình 2-6.



Hình 2-6: Quan hệ các bảng phần xuất hàng.

5.3. Phần nhập hàng

Tương tự như phần xuất hàng, phần nhập hàng bao gồm các bảng: ImportTypes, ImportBatchs, Imports và ImportDetails.

5.3.1. Bảng ImportTypes

Bảng ImportTypes dùng để lưu trữ danh sách loại phiếu nhập hàng với cấu trúc như sau:

ImportTypeID CHAR(3) NOT NULL PRIMARY KEY,
ImportTypeName NVARCHAR(50) NOT NULL

Trong đó, cột:

- ImportTypeID là mã số loại phiếu nhập hàng.
- ImportTypeName ứng với tên loại phiếu nhập hàng.

5.3.2. Bảng ImportBatches

Bảng ImportBatches dùng để lưu trữ danh sách lô phiếu nhập hàng với cấu trúc như sau:

```
ImportBatchNo VARCHAR(10) NOT NULL PRIMARY KEY,
ImportBatchDate SMALLDATETIME DEFAULT GETDATE(),
ImportDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- ImportBatchNo là mã số lô phiếu nhập hàng.
- ImportBatchDate ứng với ngày nhập lô phiếu nhập hàng.
- ImportDiscontinued ứng với trạng thái lô phiếu nhập hàng đóng hay đang còn mở cho phép nhập phiếu nhập.
- UserName ứng với tài khoản nhân viên tạo lô phiếu nhập hàng.

5.3.3. Bảng Imports

Bảng Imports dùng để lưu trữ thông tin phiếu nhập hàng, bao gồm các cột dữ liệu với cấu trúc như sau:

```
ImportNo VARCHAR(10) NOT NULL PRIMARY KEY,
ImportBatchNo VARCHAR(10) NULL,
ImportDate SMALLDATETIME DEFAULT GETDATE(),
ImportTypeID CHAR(3) NOT NULL,
SupplierID CHAR(5) NOT NULL,
DescriptionInVietnamese NVARCHAR(150) NOT NULL,
DescriptionInSecondLanguage VARCHAR(150) NOT NULL,
ShowDescriptionInVietnamese BIT DEFAULT 1,
ImportDiscontinued BIT DEFAULT 0,
EntryDate SMALLDATETIME DEFAULT GETDATE(),
UserName VARCHAR(10) NOT NULL
```

Trong đó, cột:

- ImportNo ứng với mã số phiếu nhập hàng.
- ImportBatchNo ứng với mã số lô phiếu nhập hàng.
- ImportDate ứng với ngày nhập hàng.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm71 

- ImportTypeID là mã loại phiếu nhập.
- SupplierID ứng với mã khách hàng.
- DescriptionInVietnamese ứng với diễn giải phiếu nhập hàng bằng tiếng Việt.
- DescriptionInSecondLanguage ứng với diễn giải phiếu nhập hàng bằng ngoại ngữ khác.
- ShowDescriptionInVietnamese cho phép trình bày diễn giải phiếu nhập hàng bằng tiếng Việt hay ngoại ngữ khác, mặc định trong trường hợp này là tiếng Việt.
- ImportDiscontinued ứng với trạng thái phiếu nhập hàng đóng hay đang còn mở cho phép cập nhật thông tin phiếu nhập.
- EntryDate là ngày nhập phiếu nhập hàng.
- UserName là tài khoản người nhập liệu.

5.3.4. Bảng ImportDetails

Bảng ImportDetails dùng để lưu trữ thông tin phiếu nhập hàng chi tiết, nó bao gồm các cột dữ liệu như sau:

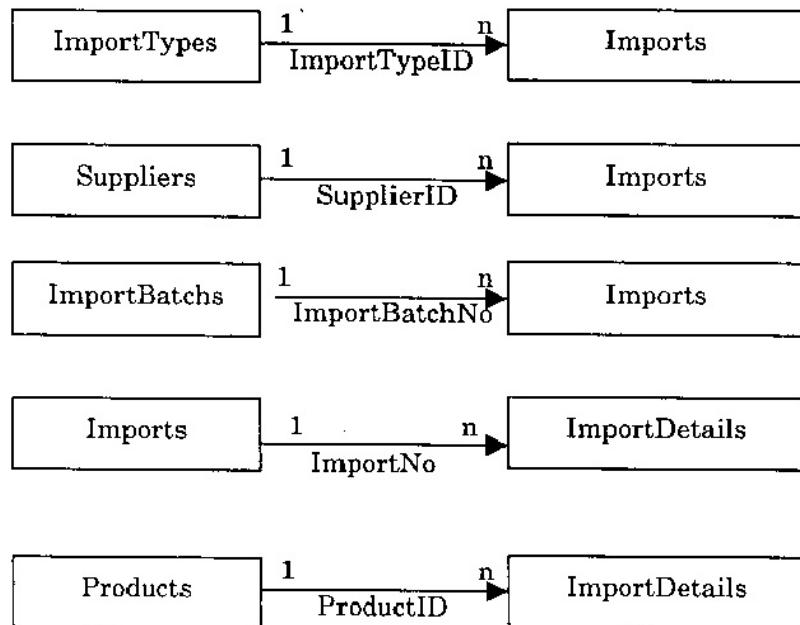
```
OrdinalNumber tinyint NOT NULL,
ImportNo VARCHAR(10) NOT NULL,
ProductID VARCHAR(10) NOT NULL,
StockID CHAR(5) NOT NULL,
Quantity DECIMAL DEFAULT 0,
PRIMARY KEY (OrdinalNumber, ExportNo, ProductID,
StockID)
```

Trong đó, cột:

- OrdinalNumber là số thứ tự của sản phẩm ghi trong phiếu nhập chi tiết.
- ImportNo ứng với mã số phiếu nhập hàng.
- ProductID ứng với mã sản phẩm.
- StockID là mã số kho hàng.
- Quantity là số lượng nhập.

5.3.5. Quan hệ

Bạn có thể tạo quan hệ giữa các bảng của phần nhập hàng được trình bày như hình 2-7.

M[®] 72**Chương 2: Tìm hiểu cơ sở dữ liệu dính kèm****Hình 2-7: Quan hệ giữa các bảng xuất hàng.**

6. CẤU TRÚC CƠ SỞ DỮ LIỆU PHẦN DÙNG CHUNG

Ngoài các bảng dữ liệu đã trình bày trong phần kế toán tổng hợp, công nợ phải thu, công nợ phải trả và xuất nhập tồn kho, chúng ta còn có một số bảng dữ liệu dùng chung.

Trong cơ sở dữ liệu dính kèm AccountSystem, chúng ta có 3 bảng dữ liệu chưa trình bày trong các phần trên là Countries, Provinces và Banks.

Chú ý: Trong quá trình thực thi ứng dụng, bạn có thể tự thêm những bảng dữ liệu còn thiếu hay thay đổi, thêm, xóa các cột dữ liệu không có nhu cầu.

6.1. Bảng Countries

Bảng Countries dùng để lưu trữ thông tin quốc gia, nó bao gồm các cột dữ liệu như sau:

CountryID CHAR(3) NOT NULL PRIMARY KEY,
CountryName NVARCHAR(30) NOT NULL

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

73

Trong đó, cột:

- CountryID ứng với mã số quốc gia.
- CountryName ứng với tên quốc gia.

6.2. Bảng Provinces

Bảng Provinces dùng để lưu trữ thông tin tỉnh thành, nó bao gồm các cột dữ liệu như sau:

ProvinceID CHAR (3) NOT NULL PRIMARY KEY,
 CountryID CHAR (3) NOT NULL,
 ProvinceName NVARCHAR (30) NOT NULL

Trong đó, cột:

- ProvinceID ứng với mã tỉnh thành.
- CountryID ứng với mã số quốc gia.
- ProvinceName ứng với tên tỉnh thành.

6.3. Bảng Banks

Bảng Banks dùng để lưu trữ thông tin Ngân hàng, nó bao gồm các cột dữ liệu như sau:

BankID CHAR (3) NOT NULL PRIMARY KEY,
 ProvinceID CHAR (3) NOT NULL,
 BankName NVARCHAR (50) NOT NULL,
 BankAddress NVARCHAR (50) NOT NULL

Trong đó, cột

- ProvinceID ứng với mã tỉnh thành.
- BankID ứng với mã số Ngân hàng.
- BankName ứng với tên Ngân hàng.
- BankAddress là địa chỉ Ngân hàng.

6.4. Bảng BankOfCustomers

Bảng BankOfCustomers chứa thông tin về tài khoản ngân hàng của khách hàng, nó bao gồm các cột như sau:

CustomerID CHAR (5) NOT NULL,
 BankID CHAR (3) NOT NULL,
 VNDAccountNo VARCHAR (10) NULL,
 USDAccountNo VARCHAR (10) NULL,
 OtherAccount VARCHAR (10) NULL
 Primary key (CustomerID , BankID)

Trong đó,

- CustomerID là mã khách hàng.
- BankID ứng với mã ngân hàng của khách hàng.
- VNDAccountNo ứng với mã tài khoản VND trong ngân hàng của khách hàng.
- USDAccountNo ứng với mã tài khoản USD trong ngân hàng của khách hàng.
- OtherAccountNo ứng với mã tài khoản loại tiền tệ khác trong ngân hàng của khách hàng.

6.5. Bảng BankOfSuppliers

Bảng BankOfSuppliers chứa thông tin về tài khoản ngân hàng của nhà cung cấp, nó bao gồm các cột như sau:

```
SupplierID CHAR (5) NOT NULL,
BankID CHAR (3) NOT NULL,
VNDAccountNo VARCHAR(10) NULL,
USDAccountNo VARCHAR(10) NULL,
OtherAccount VARCHAR(10) NULL
Primary key (SupplierID , BankID)
```

Trong đó,

- SupplierID là mã nhà cung cấp.
- BankID ứng với mã ngân hàng của nhà cung cấp.
- VNDAccountNo ứng với mã tài khoản VND trong ngân hàng của nhà cung cấp.
- USDAccountNo ứng với mã tài khoản USD trong ngân hàng của nhà cung cấp.

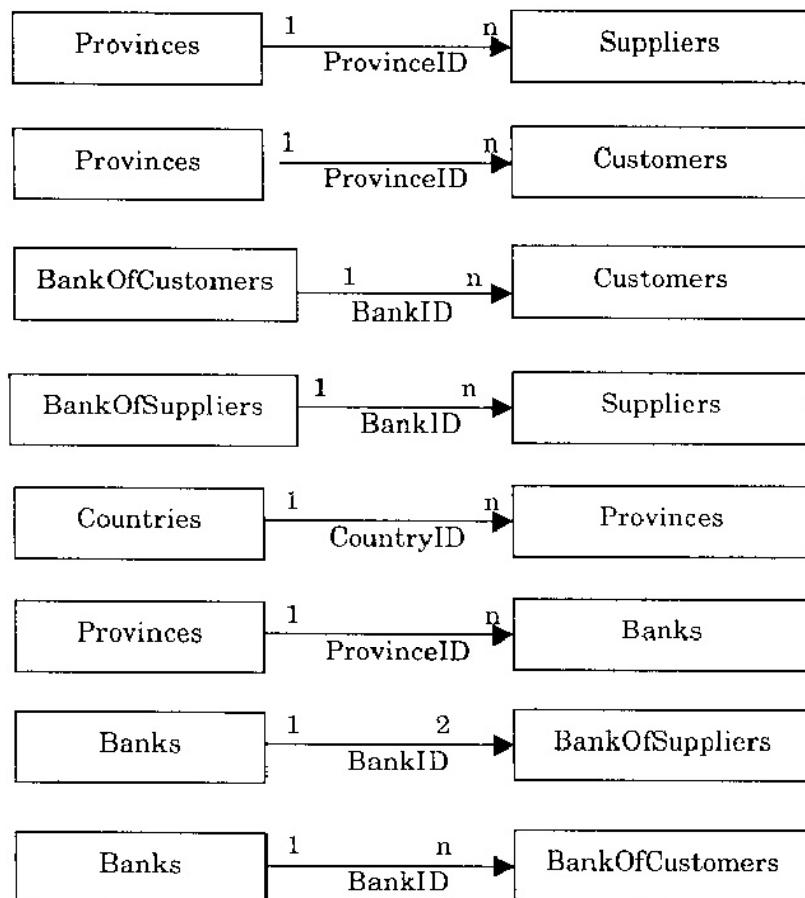
OtherAccountNo ứng với mã tài khoản loại tiền tệ khác trong ngân hàng của nhà cung cấp.

6.6. Quan hệ

Do các bảng dùng chung, bạn có thể thiết lập quan hệ của ba bảng trên với các bảng liên quan như hình 2-8.

Chương 2: Tìm hiểu cơ sở dữ liệu đính kèm

75

**Hình 2-8: Quan hệ giữa các bảng dùng chung.**

7. KẾT CHƯƠNG

Trong chương này, chúng ta tập trung tìm hiểu cấu trúc của ứng dụng được dùng để trình bày trong phần giới thiệu các đối tượng cơ sở dữ liệu, phát biểu SQL, thủ tục nội tại, hàm và Trigger.

Bạn có thể tìm thấy các phát biểu CREATE TABLE dùng để tạo bảng dữ liệu trong các Module vừa trình bày ở trên trong các tập tin .sql tương ứng được đính kèm theo đĩa.



Trong chương kế tiếp, khác với trình điều khiển Enterprise Manager Console của SQL Server 2000, chúng ta sẽ tìm hiểu cụ thể cách làm việc với SQL Server 2005 bằng trình điều khiển mới là Studio Management.

Chương 3:

LÀM VIỆC VỚI MANAGEMENT STUDIO

Tóm tắt chương 3

Từ khi phiên bản Visual Studio .NET 2002 ra đời, Microsoft giới thiệu môi trường phát triển tích hợp (IDE) có giao diện và nhiều tiện ích vượt bậc hơn nhiều so với IDE của bộ Visual Studio 6.0.

Tiếp tục cung cấp trong phiên bản Visual Studio .NET 2003 và bạn có thể tìm thấy trình IDE này được phong phú hóa và cải tiến nhiều tiện ích nâng cao trong phiên bản Visual Studio 2005.

Một lần nữa, bạn có thể tìm thấy trình IDE tương tự như Visual Studio 2005 IDE trong SQL Server 2005, bởi vì SQL Server Management Studio được xây dựng trên nền Microsoft .NET.

Trong chương này, chúng ta tập trung tìm hiểu công cụ ứng dụng này trước khi tiến hành khám phá những kỹ thuật mới trong bộ sách SQL Server 2005.

Các vấn đề chính sẽ được đề cập:

- ✓ Giao diện SQL Server Management Studio (MS).
- ✓ Khám phá cửa sổ Object Explorer.
- ✓ Đăng ký SQL Server.
- ✓ Cách quản lý dự án, tập tin trong MS.
- ✓ Làm việc với Visual SourceSafe 2005.
- ✓ Các tiện ích khác.

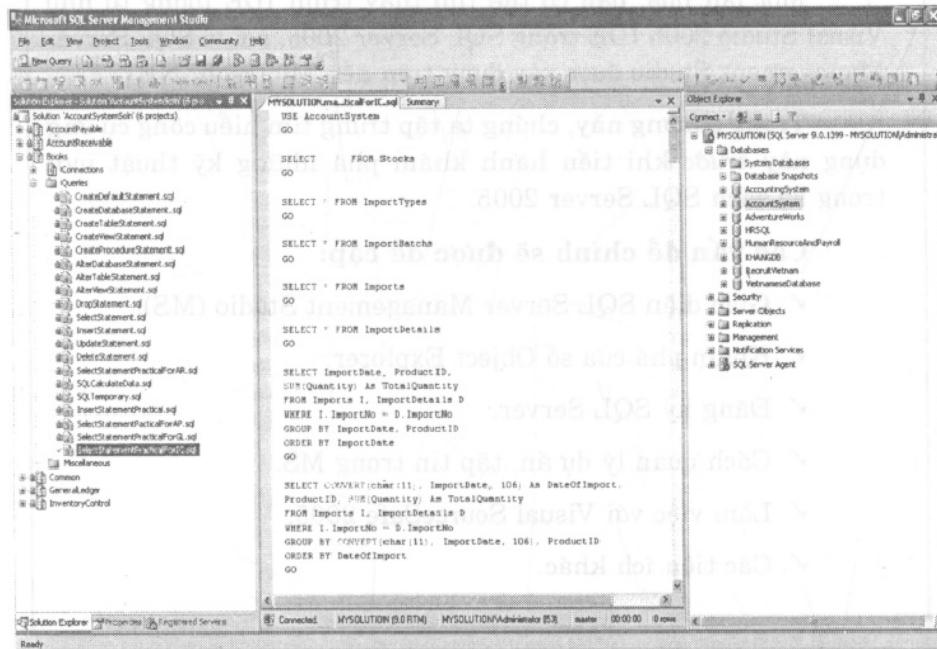
1. GIAO DIỆN SQL SERVER MANAGEMENT STUDIO

SQL Server Management Studio (MS) được cài đặt như một phần của SQL Server 2005, cung cấp các chức năng cho phép bạn quản lý cơ sở dữ liệu dễ dàng. Bạn có thể bắt đầu với ứng dụng này bằng cách chọn Start | Microsoft SQL Server 2005 | SQL Server Management Studio.

Management Studio tích hợp những tác vụ quản lý cơ sở dữ liệu và lập trình trong cùng một màn hình giao diện với người dùng.

Chú ý: Nếu bạn đã làm việc với SQL Server 2000, tiện ích cho phép bạn quản lý cơ sở dữ liệu là SQL Server Enterprise Manager Console (EMC), trong trường hợp khai báo và thực thi câu truy vấn thì bạn sử dụng SQL Query Analyzer.

Hầu hết cửa sổ trong MS đều có thể xuất hiện tại vị trí bất kỳ hay gắn kế cạnh lề của MS, bạn cũng có thể sử dụng che dấu hay xuất hiện một cửa sổ không có nhu cầu làm việc nhằm tăng không gian sử dụng cho các cửa sổ còn lại như hình 3-1.

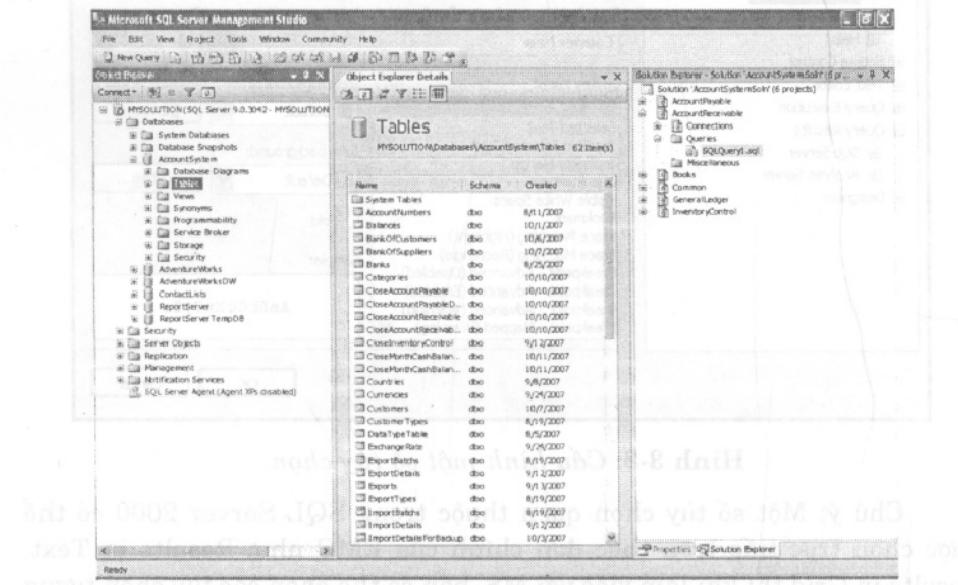


Hình 3-1: Màn hình MS.

Chương 3: Làm việc với Management Studio

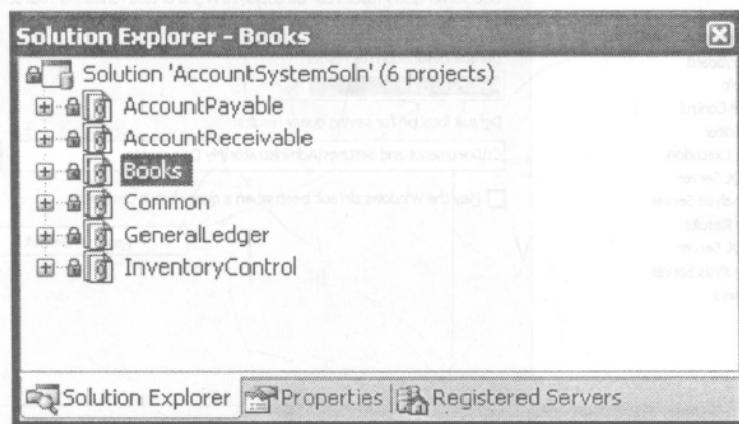
79 

Tuy nhiên, do không gian MS có hạn, nên bạn cũng có thể sắp xếp các cửa sổ trên cùng một vùng và thể hiện cửa sổ qua từng ngăn tương ứng như hình 3-2.



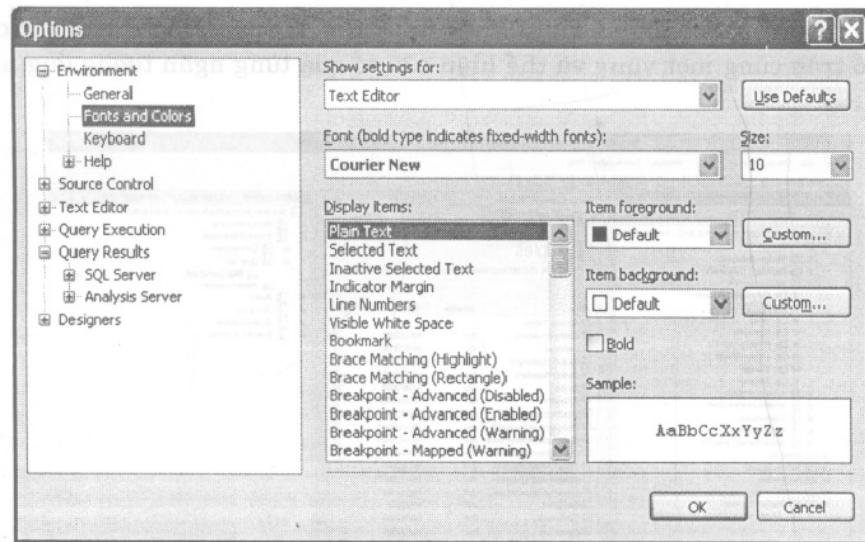
Hình 3-2: Sắp xếp cửa sổ dạng Tab.

Hoặc bạn cũng có thể chồng nhiều cửa sổ trên cùng một không gian như hình 3-2-1.

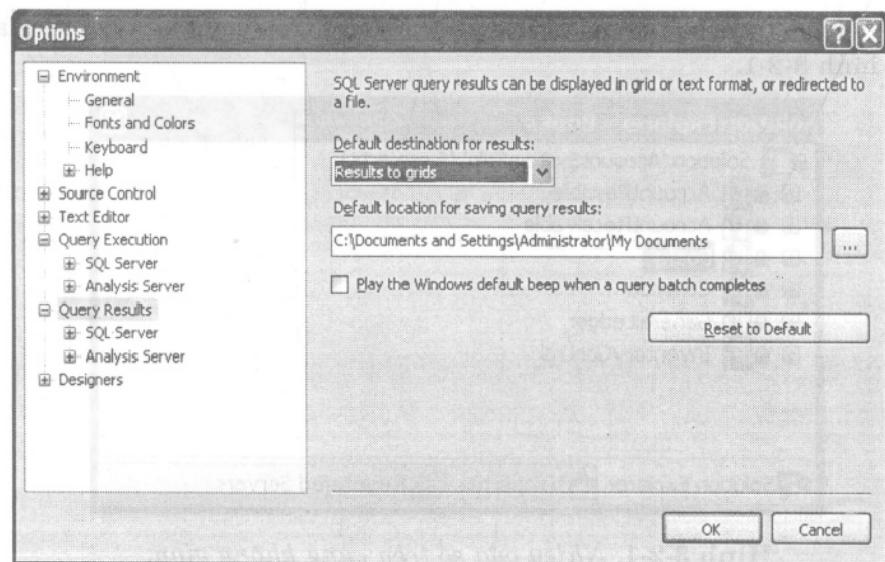


Hình 3-2-1: Nhiều cửa sổ trên cùng không gian.

Khi làm việc với SQL Server 2005, bạn có thể thay đổi một số cấu hình bằng cách chọn vào Tools | Options, cửa sổ xuất hiện như hình 3-3.

Chương 3: Làm việc với Management Studio**Hình 3-3: Cấu hình một số tùy chọn.**

Chú ý: Một số tùy chọn quen thuộc trong SQL Server 2000 có thể được chọn trực tiếp trên thực đơn chính của EMC như: Results in Text, Results in Grid thì khi làm việc với MS, bạn có thể chọn các tùy chọn tương ứng trong phần Query Results như hình 3-4.

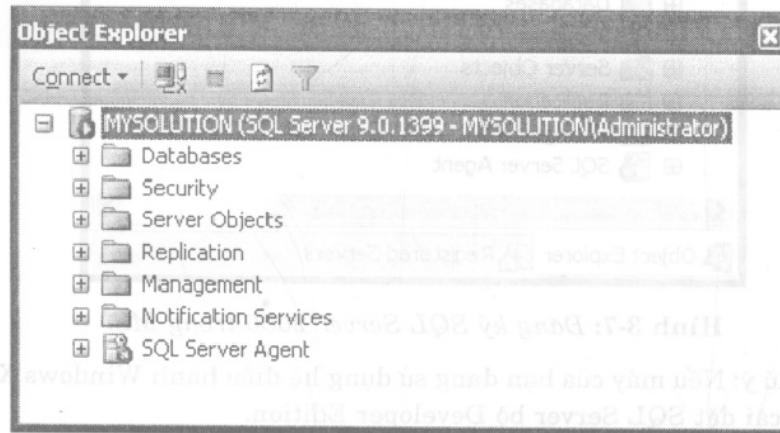
**Hình 3-4: Tùy chọn trình bày kết quả thực thi.**

Chương 3: Làm việc với Management Studio

81 M®

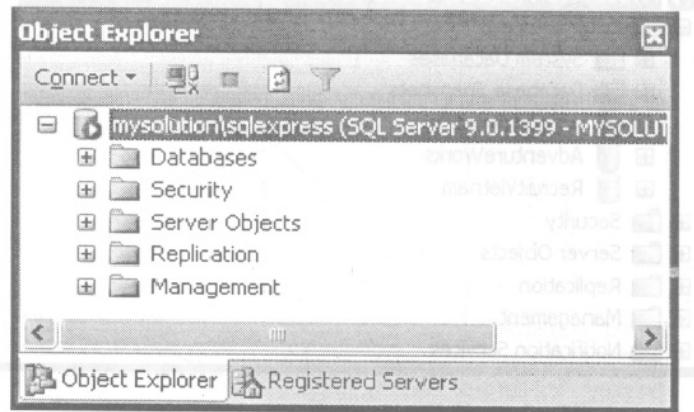
2. KHÁM PHÁ CỦA SỔ OBJECT EXPLORER

Cửa sổ Object Explorer cung cấp góc nhìn dạng cây chứa danh sách các đối tượng của đối tượng SQL Server đã kết nối (SQL Server Instance), bao gồm danh sách tên Server đã đăng ký, bên trong mỗi Server, bạn có thể tìm thấy các thành phần chính như: Databases, Security, Server Objects, Replication và Management trông giống như hình 3-5.



Hình 3-5: Cửa sổ Object Explorer.

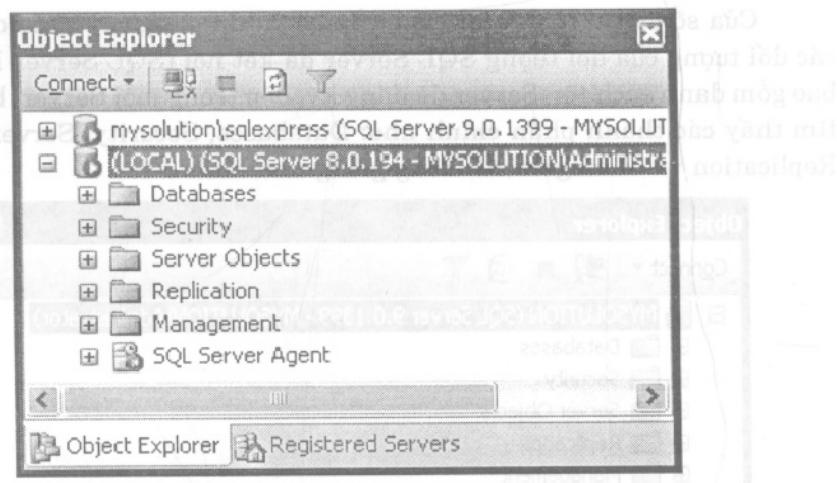
Chú ý: Nếu như bạn đã cài đặt SQL Server 2005 (Express) trước khi cài đặt Component của SQL Server 2005 thì cửa sổ Object Explorer có thể tương tự như hình 3-6.



Hình 3-6: Cửa sổ Object Explorer.

Chương 3: Làm việc với Management Studio

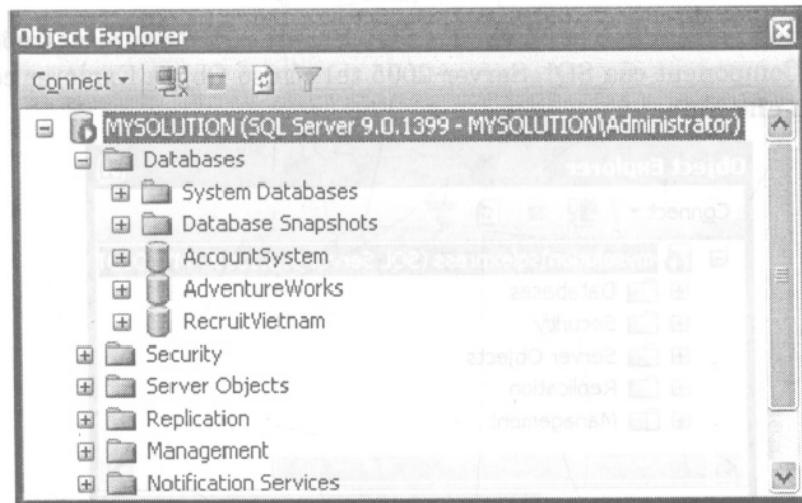
Bạn cũng có thể đăng ký SQL Server 2000 trong MS trông giống như hình 3-7.



Hình 3-7: Đăng ký SQL Server 2000 trong MS.

Chú ý: Nếu máy của bạn đang sử dụng hệ điều hành Windows XP thì bạn cần cài đặt SQL Server bộ Developer Edition.

Trong đó, ngăn Databases chứa danh sách cơ sở dữ liệu như hình 3-8.



Hình 3-8: Danh sách cơ sở dữ liệu.

Chú ý: Bạn có thể tham khảo cách tạo cơ sở dữ liệu trong chương kế tiếp.

Chương 3: Làm việc với Management Studio

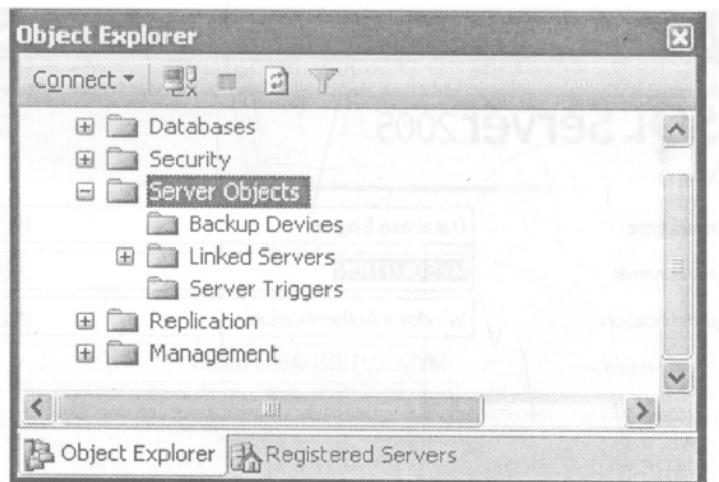
83 M®

Ngoài ra, bạn có thể tìm thấy nhóm và người sử dụng của SQL Server 2005 trong ngăn Security như hình 3-9.



Hình 3-9: Nhóm và người sử dụng.

Tương tự như vậy, bạn có thể tìm thấy đối tượng bản sao (backup) cơ sở dữ liệu, liên kết cơ sở dữ liệu (linked) trong ngăn Server Objects như hình 3-10.



Hình 3-10: Đối tượng trình chủ SQL Server.

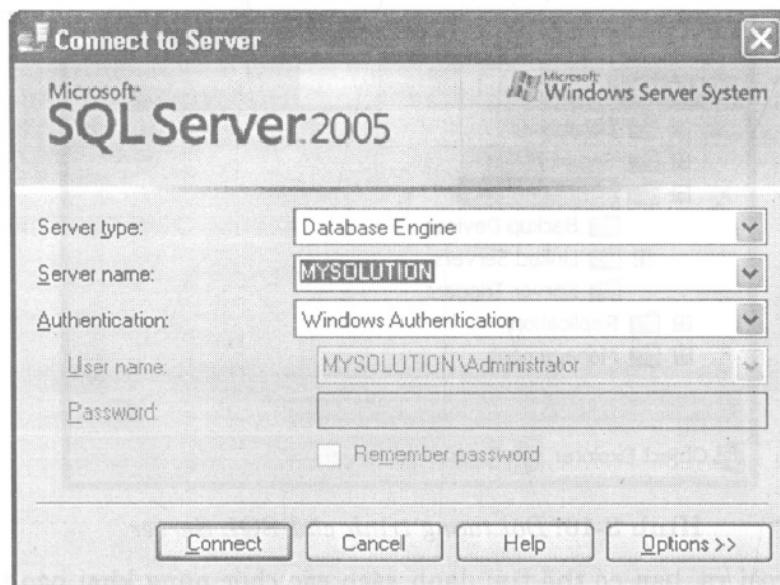
Ngoài ra, bạn có thể tìm danh sách các chức năng khai báo tác vụ thực thi tự động trong ngăn Management như hình 3-11.



Hình 3-11: Chức năng tự động thực thi.

3. ĐĂNG KÝ SQL SERVER

Để sử dụng MS, bạn cần đăng nhập vào SQL Server 2005 bằng cửa sổ như hình 3-12. Nếu bạn chọn tên Server cài đặt SQL Server nào thì Server đó sẽ được chọn mặc định để làm việc trong MS.



Hình 3-12: Đăng nhập MS.

Chương 3: Làm việc với Management Studio

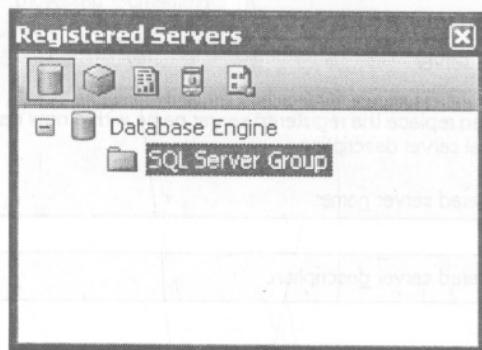
85

Chú ý: Hình vừa trình bày ở trên được cài đặt từ SQL Server 2005 bộ Developer Edition.

Khi đăng nhập MS, bạn cũng có thể chọn một trong hai tùy chọn đặc quyền đăng nhập, đặc quyền hệ điều hành (Windows Authentication) hay đặc quyền SQL Server (SQL Server Authentication).

Chú ý: Đặc quyền hệ điều hành là người sử dụng sử dụng quyền của tài khoản đã đăng nhập vào hệ điều hành để truy cập vào cơ sở dữ liệu SQL Server. Trong khi đó, đặc quyền SQL Server là sử dụng tài khoản tạo trong SQL Server để đăng nhập cơ sở dữ liệu.

Dù chọn một trong hai đặc quyền trên để đăng nhập, nếu đăng nhập thành công, bạn có thể đăng ký SQL Server khác vào MS. Để đăng ký SQL Server vào MS, bạn sử dụng cửa sổ Registered Servers như hình 3-13.

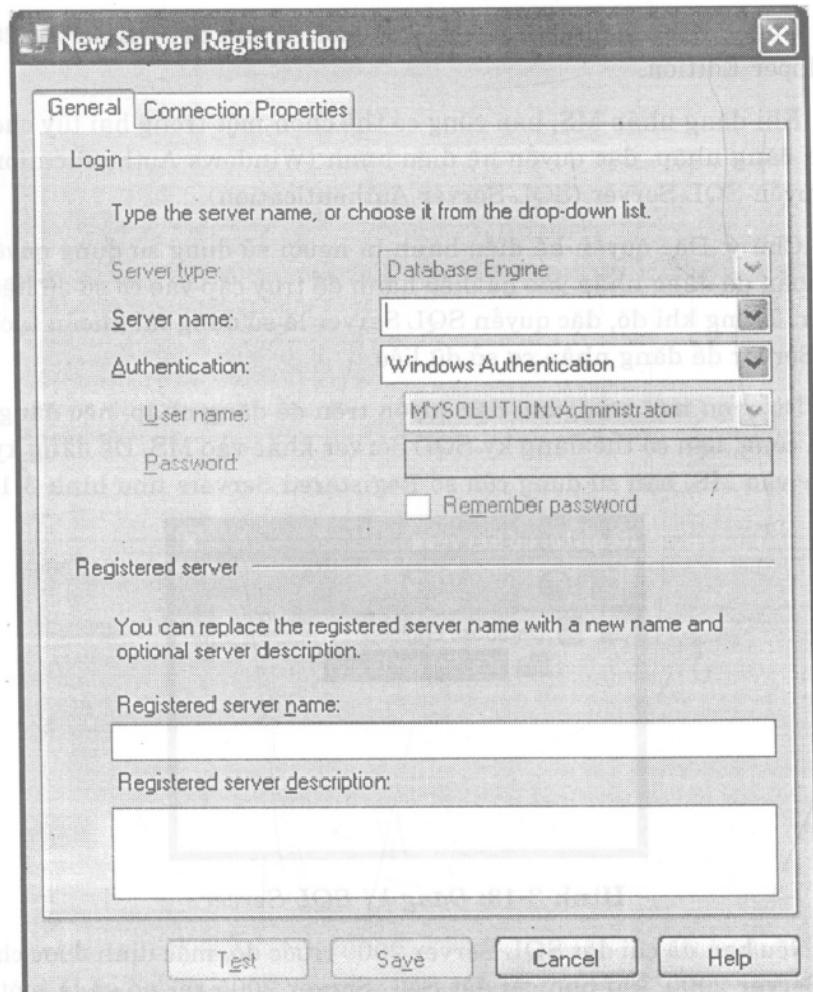


Hình 3-13: Đăng ký SQL Server.

Nếu bạn đã cài đặt SQL Server 2000 trước đó, mặc định được chọn là SQL Server 2000, khi bạn cài đặt SQL Server 2005 thì nó sẽ là một SQL Server Instance kế tiếp.

Tuy nhiên, trong trường hợp bạn cài đặt SQL Server 2005 phiên bản Developer hay Enterprise thì chúng được đăng ký mặc định.

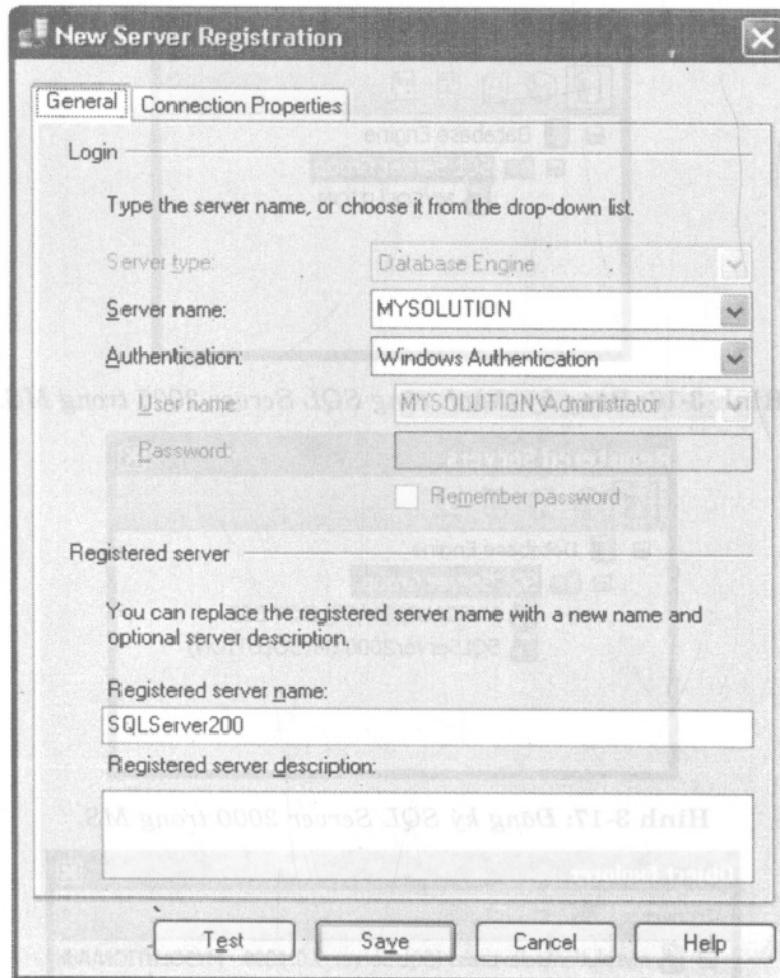
Để đăng ký SQL Server vào MS, bạn chọn Database Engine rồi R-Click | New | Server Registration, cửa sổ đăng ký xuất hiện như hình 3-14.

**Hình 3-14:** Đăng ký SQL Server.

Bằng cách chọn tên Server trong danh sách Server Name, bạn tiếp tục chọn đặc quyền đăng nhập trong phần Authentication, rồi chọn nút Save, Server vừa chọn đã được đăng ký và xuất hiện trong cửa sổ Registered Servers.

Chú ý: Nếu bạn đăng ký SQL Server trong EMC của SQL Server, bạn không thể thay đổi tên của chúng.

Tuy nhiên, khi làm việc với SQL Server 2005, bạn có thể khai báo tên đăng ký trong phần Registered server name như **Hình 3-15**.

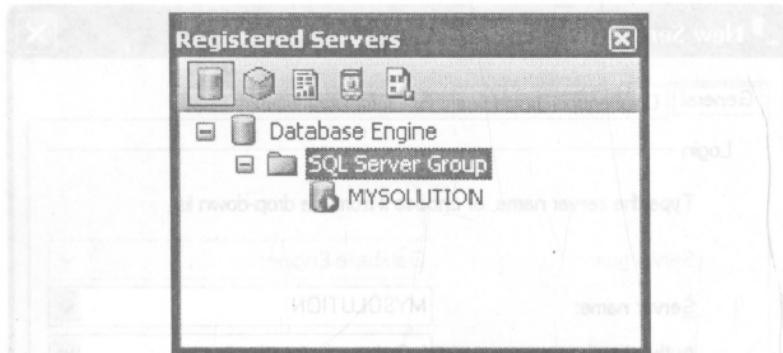
Chương 3: Làm việc với Management Studio87 **Hình 3-15:** Đăng ký tên khác tên Server.

Chú ý: Bạn có thể tham khảo chi tiết về hai loại đặc quyền này trong phần trình bày quyền của người sử dụng thuộc chương kế tiếp.

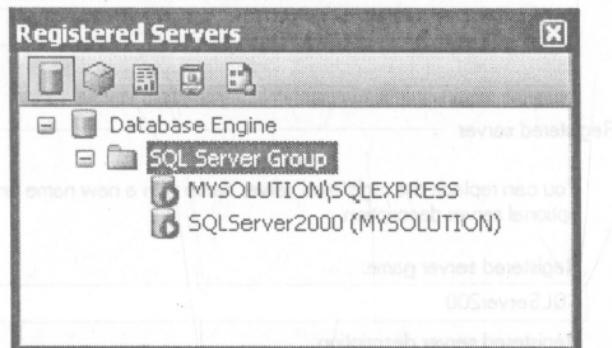
Sau khi đăng ký SQL Server thành công, bạn có thể tìm thấy SQL Server vừa đăng ký trong cửa sổ Registered Servers tương tự như hình 3-16.

Chú ý: Bạn cũng có thể đăng ký SQL Server 2005, SQL Server 2000 trong cửa sổ Registered Servers như hình 3-17.

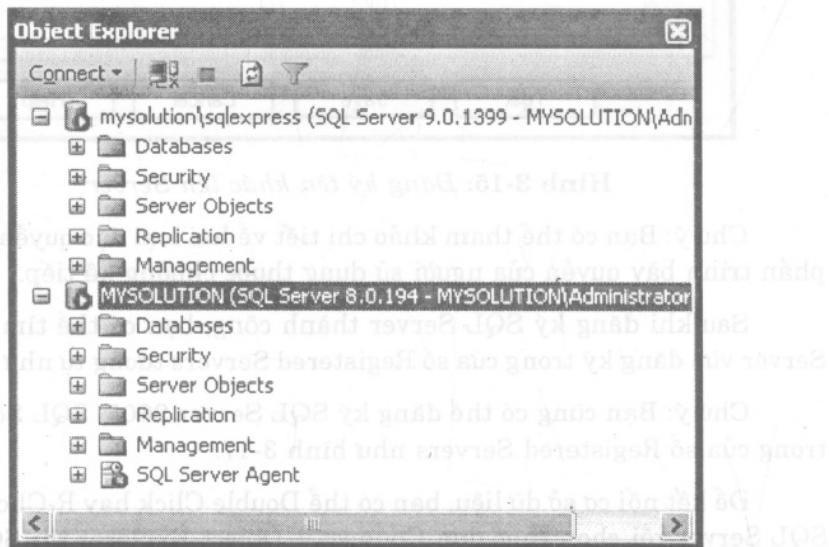
Để kết nối cơ sở dữ liệu, bạn có thể Double Click hay R-Click trên tên SQL Server rồi chọn thực đơn Connect | Object Explorer thì SQL Server này sẽ xuất hiện trên cửa sổ Object Explorer như hình 3-18.



Hình 3-16: Đăng ký thành công SQL Server 2005 trong MS.



Hình 3-17: Đăng ký SQL Server 2000 trong MS.



Hình 3-18: Kết nối SQL Server 2000.

Ngoài ra, bạn cũng có thể kết nối SQL Server đã đăng ký để khai báo và thực thi phát biểu T-SQL trong cửa sổ Query thì R-Click trên tên SQL Server và chọn Connect | New Query thay vì Object Explorer.

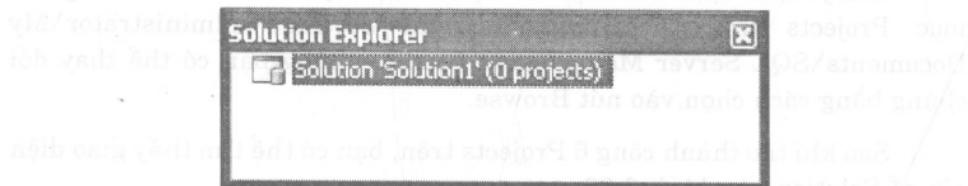
4. QUẢN LÝ DỰ ÁN, TẬP TIN TRONG MS

Một trong những đặc điểm mới của MS giới thiệu trong phiên bản này là chức năng quản lý Solution, Project và File như Visual Studio 2005.

Do MS cho phép bạn quản lý kịch bản T-SQL thành từng tập tin có tên mở rộng .sql, chính vì vậy bạn cần tạo ra Project để tập trung các tập tin theo nhóm.

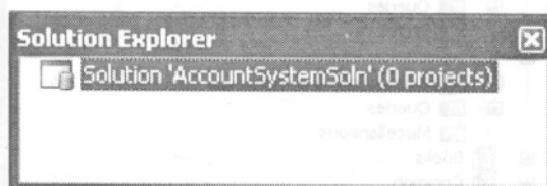
Chú ý: Tên tập tin mở rộng của Solution là .ssmssqln và Project là .ssmssqlproj.

Đối với cơ sở dữ liệu lớn, chúng ta có nhiều người cùng làm việc trên dự án, việc chia ứng dụng ra thành nhiều Project là điều có thể và sử dụng Solution để quản lý các Project này, đây chính là lý do tại sao bạn có thể tìm thấy cửa sổ Solution Explorer trong MS như hình 3-19.



Hình 3-19: Cửa sổ Solution Explorer.

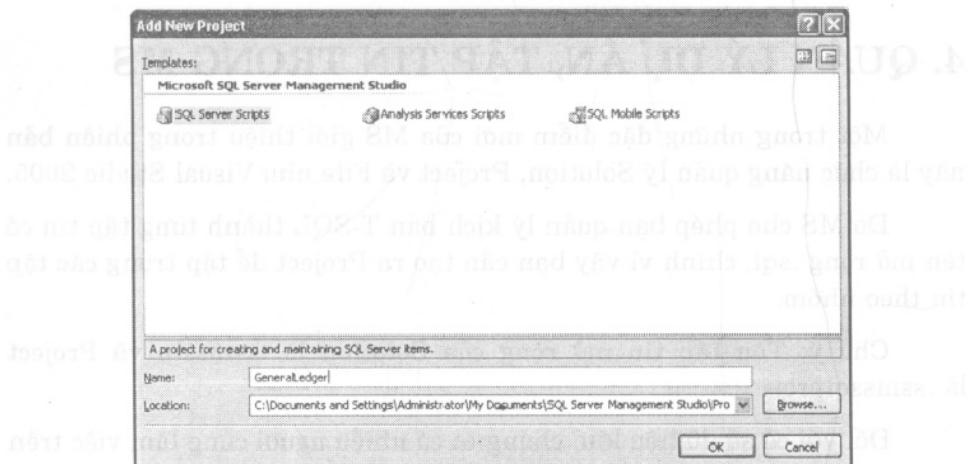
Chú ý: Bạn có thể đổi tên Solution1 thành AccountSystemSoln bằng cách R-Click | Rename và tên mới xuất hiện như hình 3-20.



Hình 3-20: Đổi tên Solution.

Như giới thiệu trong chương 2, chúng ta sẽ có 4 Module chính trong ứng dụng kế toán tham khảo dính kèm, bạn có thể tạo 4 Project tương ứng là GeneralLedger, AccountReceivable, AccountPayable, InventoryControl.

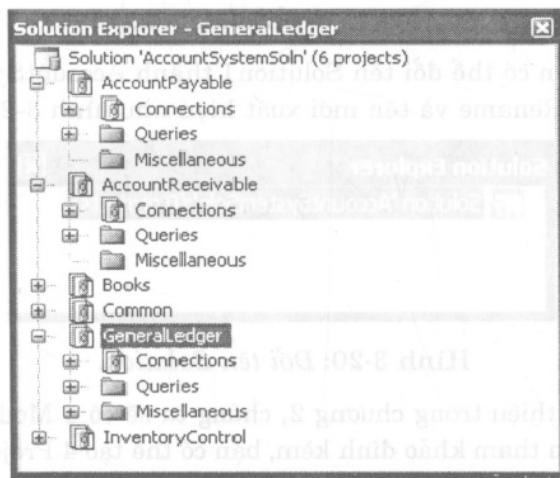
Ngoài ra, bạn cũng tạo thêm hai Project phụ Common và Books. Chẳng hạn, để tạo mới Project bạn R-Click trên tên Solution và chọn New Project và đặt tên Project như hình 3-21.



Hình 3-21: Tạo mới và đặt tên Project.

Lưu ý: Mặc định thư mục lưu dự án của SQL Server nằm trong thư mục Projects thuộc C:\Documents and Settings\Administrator\My Documents\SQL Server Management Studio, nhưng bạn có thể thay đổi chúng bằng cách chọn vào nút Browse.

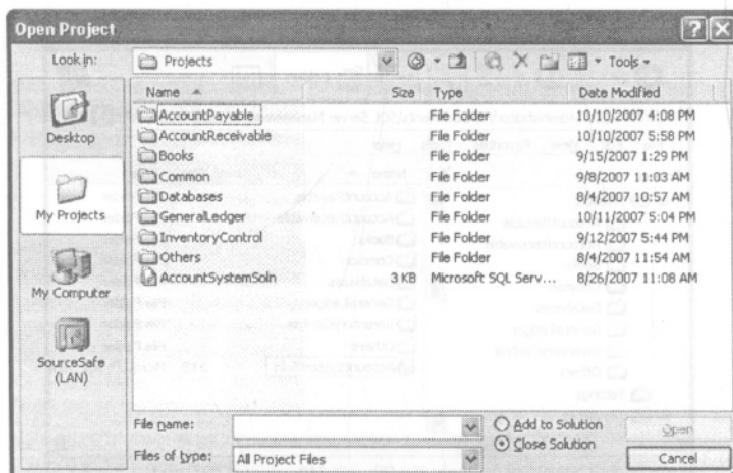
Sau khi tạo thành công 6 Projects trên, bạn có thể tìm thấy giao diện cửa sổ Solution như hình 3-22.



Hình 3-22: Danh sách các Project vừa tạo.

Chương 3: Làm việc với Management Studio91 

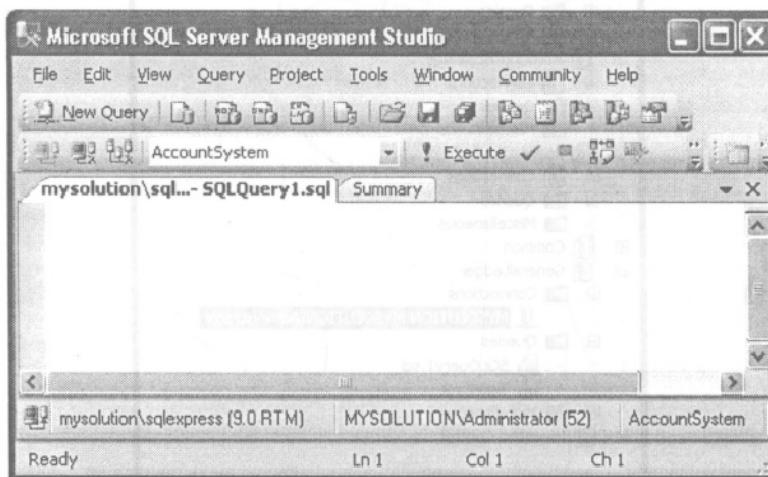
Chú ý: Để mở Solution đã tồn tại có tên AccountSystemSoln, bạn chọn vào thực đơn File | Open | Project/Solution và trỏ đến tập tin AccountSystemSoln.sln tương tự như hình 3-23.



Hình 3-23: Mở Solution đã tồn tại.

Trong mỗi Project, bạn có thể tạo mới kết nối cơ sở dữ liệu bằng cách R-Click trên ngăn Connections rồi chọn New Connection hay Query bằng cách R-Click trên ngăn Queries rồi chọn New Query.

Chẳng hạn, bạn muốn tạo mới cửa sổ Query thì chọn vào nút New Query trên thanh công cụ, lập tức cửa sổ New Query trình bày như hình 3-24.

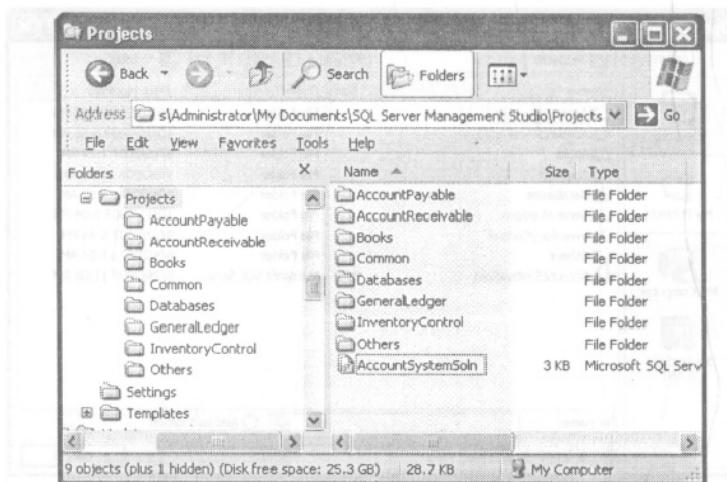


Hình 3-24: Vùng làm việc của cửa sổ New Query.

[A]® 92

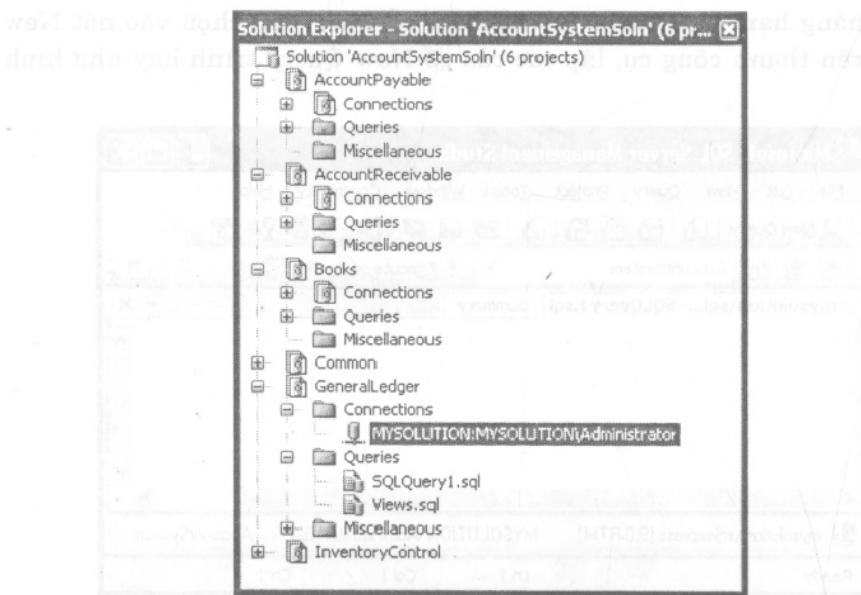
Chương 3: Làm việc với Management Studio

Bạn có thể tìm thấy thư mục chứa Project và tập tin .sql của từng Project trong thư mục C:\Documents and Settings\Administrator\My Documents\SQL Server Management Studio\Projects\ như hình 3-25.



Hình 3-25: Cấu trúc thư mục của Solution.

Mỗi lần tạo mới tập tin .sql, cửa sổ đăng nhập SQL Server 2005 xuất hiện yêu cầu bạn chọn đặc quyền kết nối, nếu kết nối thành công, lập tức kết nối xuất hiện trong ngăn Connections như hình 3-26.

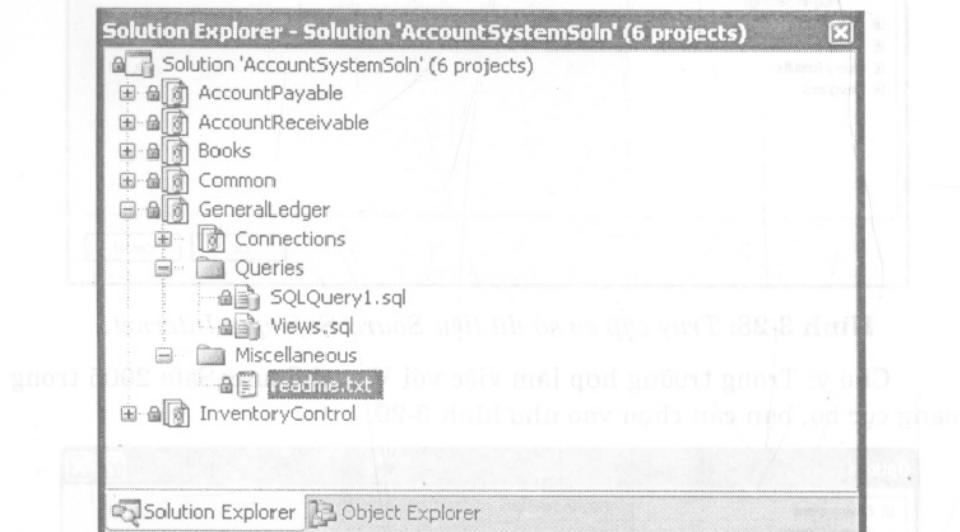


Hình 3-26: Tạo kết nối cơ sở dữ liệu SQL Server 2005.

Chương 3: Làm việc với Management Studio93 

Chú ý: Chúng ta sẽ tiếp tục tìm hiểu cách tạo và khai báo kết nối cùng với phát biểu T-SQL trong chương 5.

Trong trường hợp bạn muốn quản lý các loại tập tin khác trong Project, tập tin thêm vào sẽ nằm trong ngăn Miscellaneous. Chẳng hạn, bạn thêm tập tin readme.txt vào Project có tên GeneralLedger trông giống như hình 3-27.



Hình 3-27: Tập tin khác.

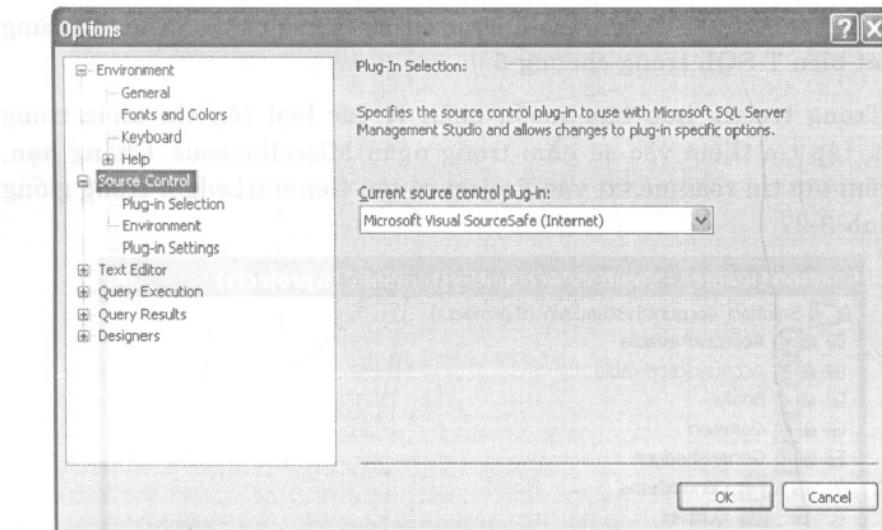
Chú ý: Để tìm hiểu thêm cách cài đặt và cấu hình SQL Server 2005, bạn có thể tham khảo chi tiết trong phần phụ lục.

5. LÀM VIỆC VỚI VISUAL SOURCESAFE 2005

Giả sử, bạn đã cài đặt Visual SourceSafe 2005, nếu muốn lưu các thành phần trong Solution vừa tạo ở trong phần 4 vào máy khác trên mạng cục bộ, bạn cần cài đặt Visual SourceSafe 2005 trên máy đó.

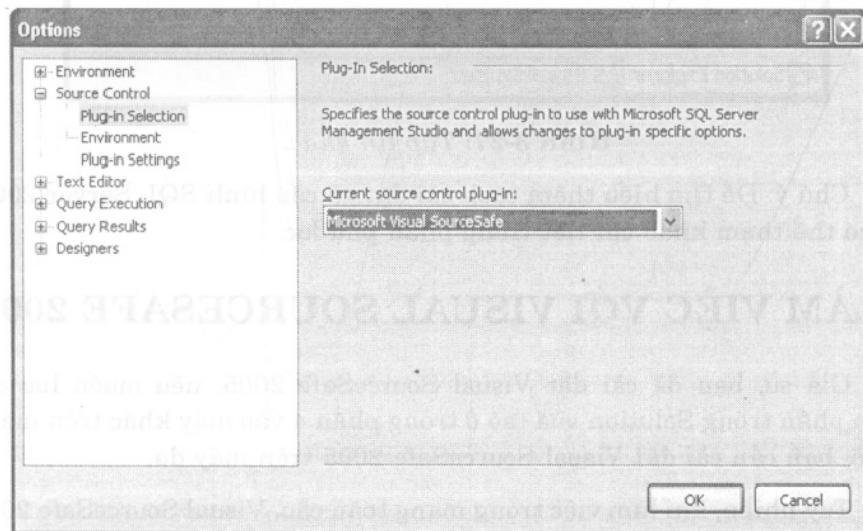
Tuy nhiên, khi làm việc trong mạng toàn cầu, Visual SourceSafe 2005 đặt trên máy Server có thể nằm ở một nơi nào đó trên thế giới, để truy cập vào cơ sở dữ liệu Visual SourceSafe 2005, bạn có thể sử dụng VPN (Virtual Private Network) hoặc bạn có thể sử dụng chức năng quản lý mã nguồn (Source Control) trong MS và cấu hình trong Tools | Options thông qua HTTP như hình 3-28.

Chương 3: Làm việc với Management Studio



Hình 3-28: Truy cập cơ sở dữ liệu SourceSafe qua Internet.

Chú ý: Trong trường hợp làm việc với Visual SourceSafe 2005 trong mạng cục bộ, bạn cần chọn vào như hình 3-29.

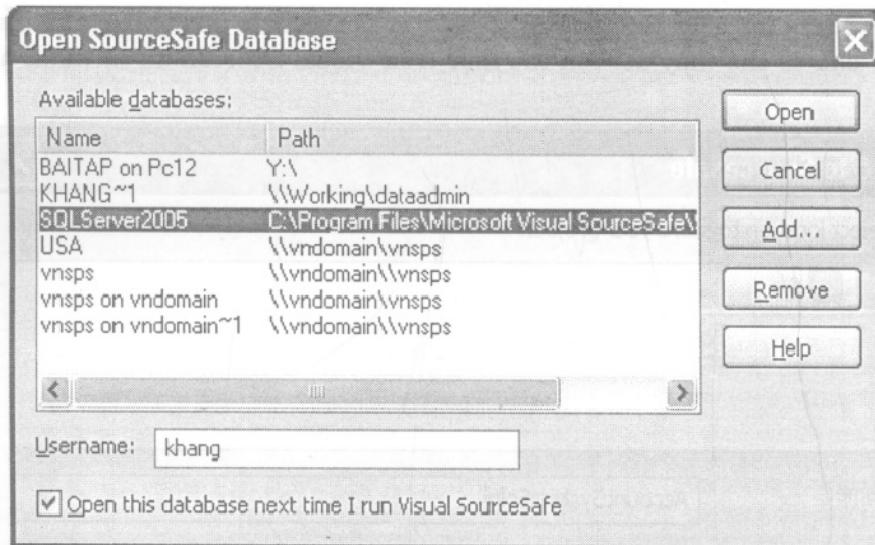


Hình 3-29: Làm việc với Visual SourceSafe 2005 trong mạng cục bộ.

Sau khi đã chọn được máy chủ nơi cài đặt Visual SourceSafe 2005, bạn có thể chọn vào File | Source Control | Add Solution to Source Control, cửa sổ Open SourceSafe Database xuất hiện như hình 3-30.

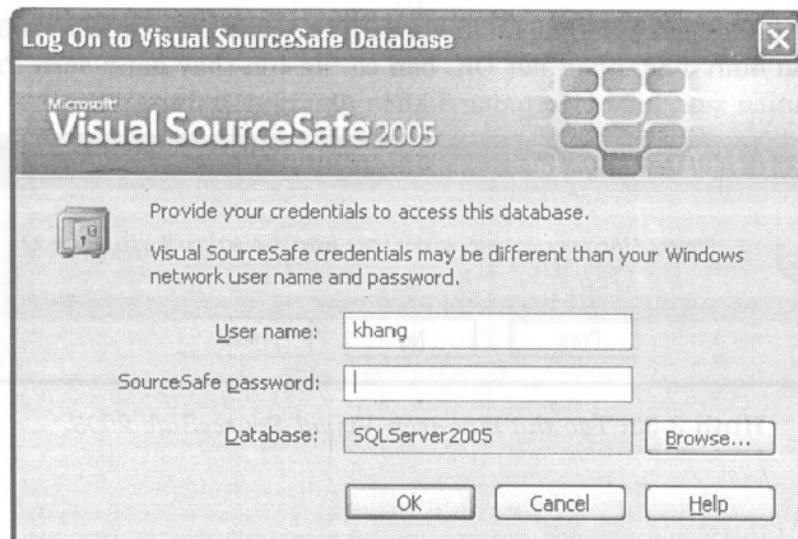
Chương 3: Làm việc với Management Studio

95

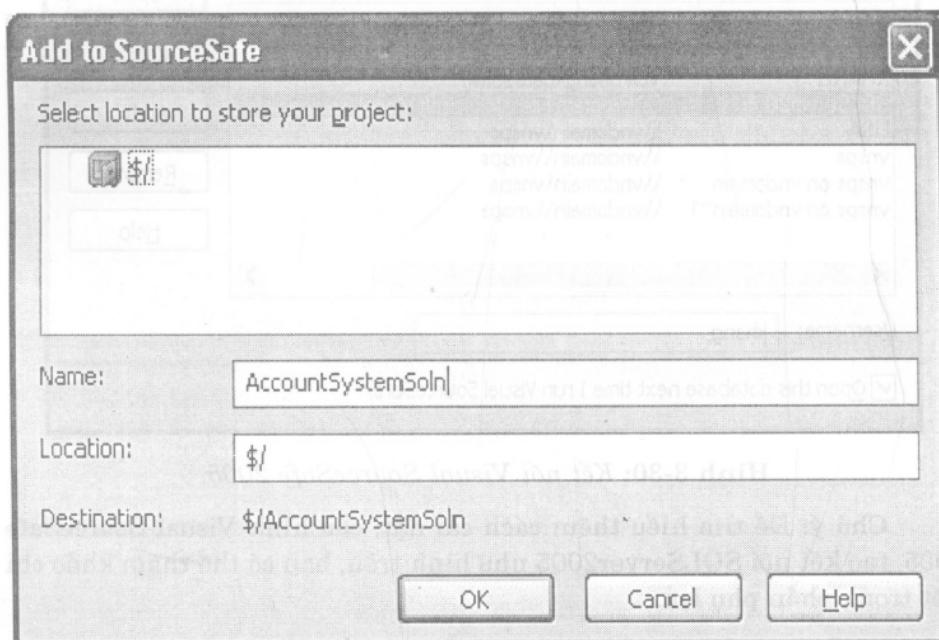
**Hình 3-30:** Kết nối Visual SourceSafe 2005.

Chú ý: Để tìm hiểu thêm cách cài đặt, cấu hình Visual SourceSafe 2005, tạo kết nối SQLServer2005 như hình trên, bạn có thể tham khảo chi tiết trong phần phụ lục.

Nhấn nút Open, lập tức cửa sổ yêu cầu đăng nhập vào Visual SourceSafe Database xuất hiện như hình 3-31.

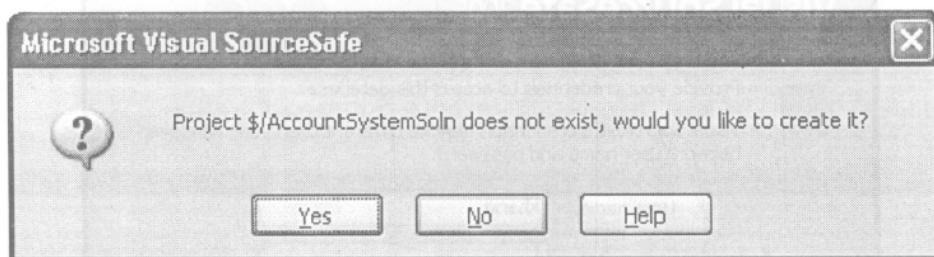
**Hình 3-31:** Đăng nhập Visual SourceSafe 2005.

Nhấn nút OK, lập tức cửa sổ Add to SourceSafe xuất hiện như hình 3-32, bạn có thể thay đổi tên thư mục trên Visual SourceSafe và nhấn nút OK.



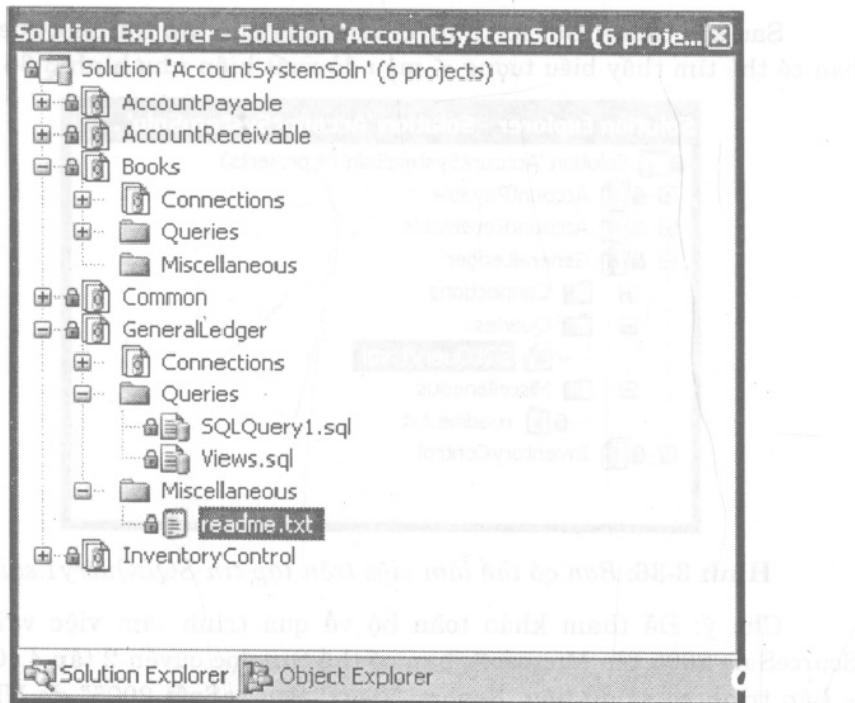
Hình 3-32: Đăng ký Solution trên Visual SourceSafe 2005.

Lập tức cửa sổ yêu cầu tạo thư mục trên Visual SourceSafe 2005 xuất hiện như hình 3-33, nhấn nút OK, bạn có thể tìm thấy danh sách Project của Solution xuất hiện biểu tượng ổ khóa như hình 3-34.

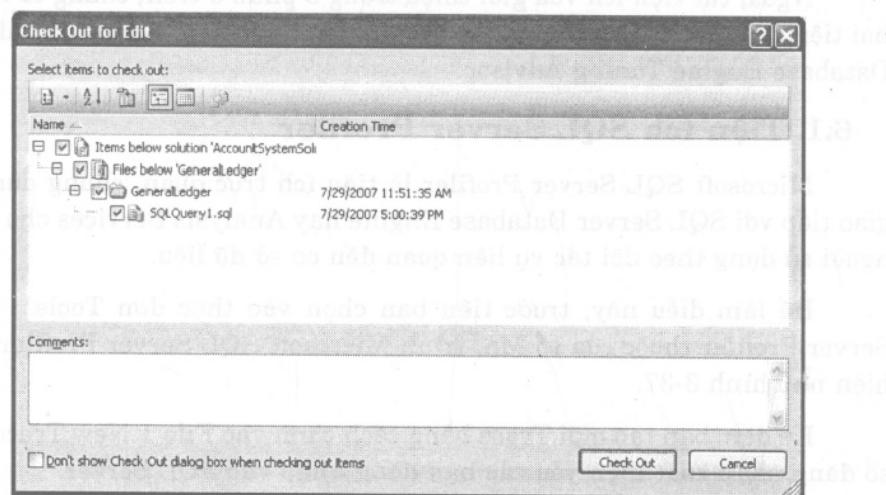


Hình 3-33: Tạo thư mục trên Visual SourceSafe 2005.

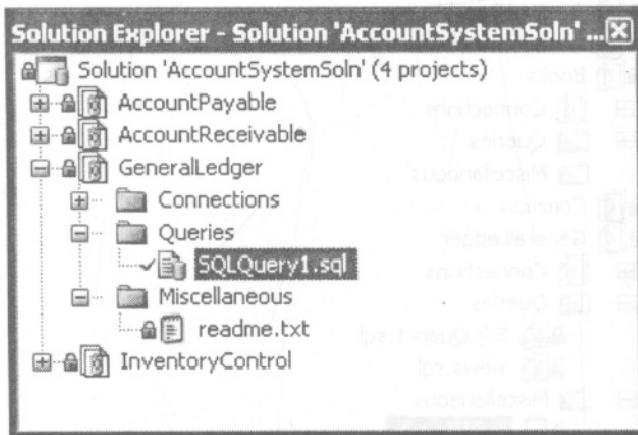
Chương 3: Làm việc với Management Studio

97 **Hình 3-34:** Đăng ký vào Visual SourceSafe 2005 thành công.

Sau khi đăng ký thành công, để có thể thay đổi tên Solution, thêm, thay đổi hay xóa Project; thêm, xóa hay thay đổi nội dung tập tin trong mỗi Project bạn đều phải Check Out for Edit.

**Hình 3-35:** Check Out for Edit.

Sau khi Check Out for Edit thành công cho tập tin SQLQuery1.sql, bạn có thể tìm thấy biểu tượng ✓ màu đỏ xuất hiện như hình 3-36.



Hình 3-36: Bạn có thể làm việc trên tập tin SQLQuery1.sql.

Chú ý: Để tham khảo toàn bộ về quá trình làm việc với Visual SourceSafe 2005 của Microsoft, bạn có thể tìm đọc quyển 2 tập 4 “C# 2005 – Lập trình cơ sở dữ liệu, Report, Visual SourceSafe 2005” do Nhà sách Minh Khai phát hành.

6. CÁC TIỆN ÍCH KHÁC

Ngoài các tiện ích vừa giới thiệu trong 5 phần ở trên, chúng ta còn có hai tiện ích khác mà bạn cũng nên tìm hiểu thêm là SQL Server Profiler và Database Engine Tuning Advisor.

6.1. Tiện ích SQL Server Profiler

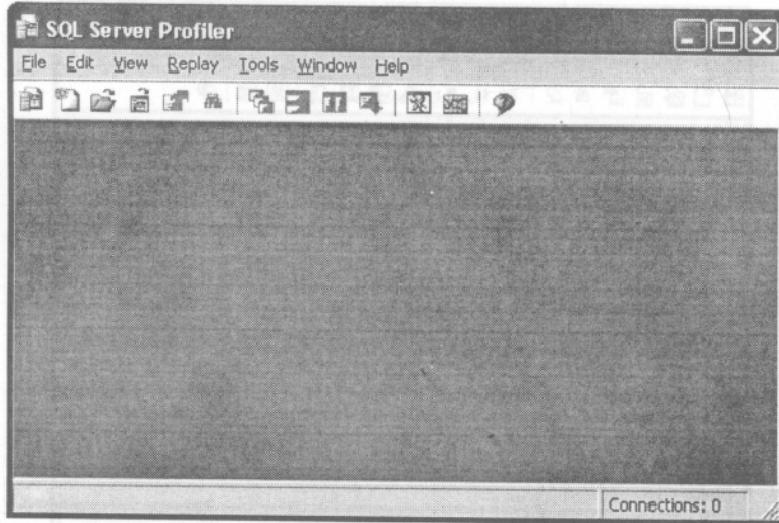
Microsoft SQL Server Profiler là tiện ích trực quan, chúng dùng để giao tiếp với SQL Server Database Engine hay Analysis Services cho phép người sử dụng theo dõi tác vụ liên quan đến cơ sở dữ liệu.

Để làm điều này, trước tiên bạn chọn vào thực đơn Tools | SQL Server Profiler thuộc cửa sổ MS, trình Microsoft SQL Server Profiler xuất hiện như hình 3-37.

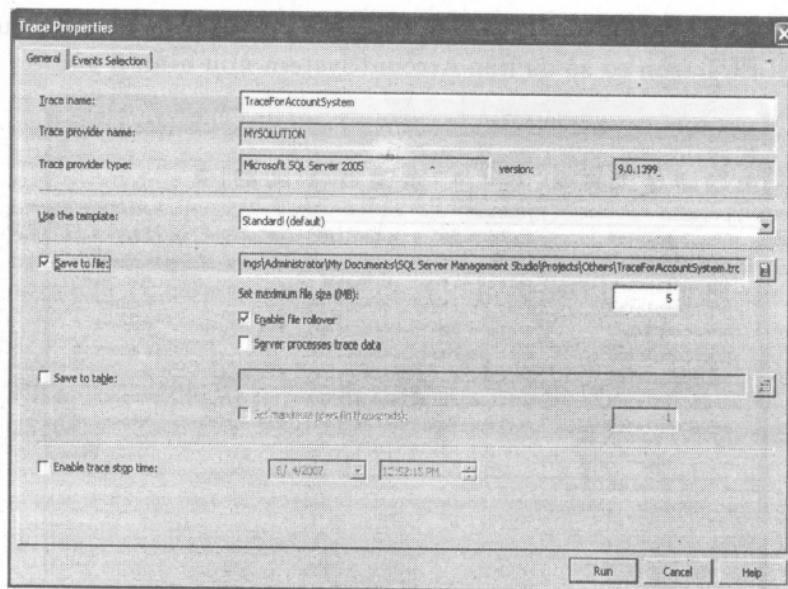
Kế đến, bạn tạo mới Trace bằng cách chọn vào File | New Trace, cửa sổ đăng nhập xuất hiện yêu cầu bạn đăng nhập vào SQL Server.

Chương 3: Làm việc với Management Studio

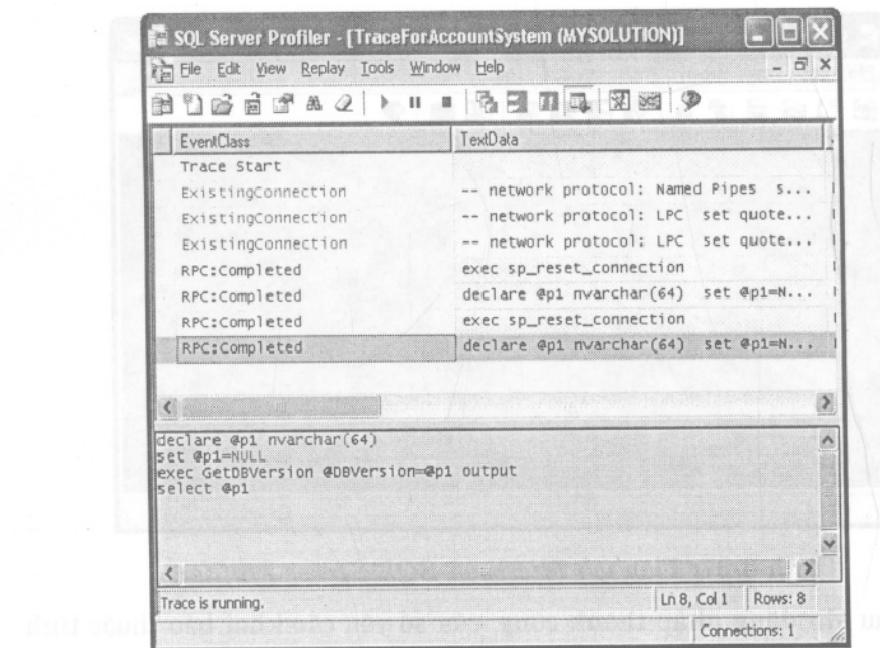
99 M®

**Hình 3-37:** Tiện ích Microsoft SQL Server Profiler.

Sau khi đăng nhập thành công, cửa sổ yêu cầu khai báo thuộc tính cho tác vụ theo dõi các tác vụ thực thi xuất hiện, bạn có thể khai báo thuộc tính cho tác vụ theo dõi này như hình 3-38.

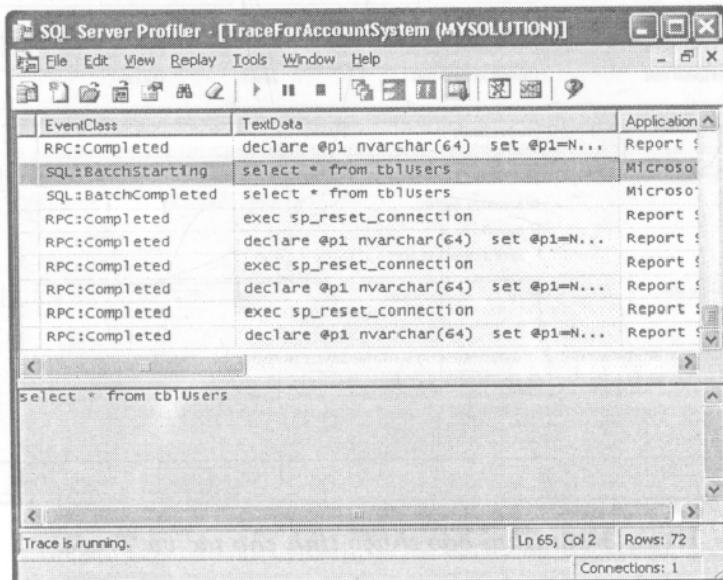
**Hình 3-38:** Khai báo thuộc tính cho tác vụ Trace.

Nhấn nút Run, cửa sổ tiếp theo xuất hiện như hình 3-39, trong cửa sổ này bạn có thể tìm thấy mọi sự kiện xảy ra trong SQL Server 2005.



Hình 3-39: Sự kiện xảy ra trong SQL Server 2005.

Chẳng hạn, bạn có thể theo dõi quá trình thực thi phát biểu SQL dạng SELECT trên cơ sở dữ liệu AccountSystem như hình 3-40.

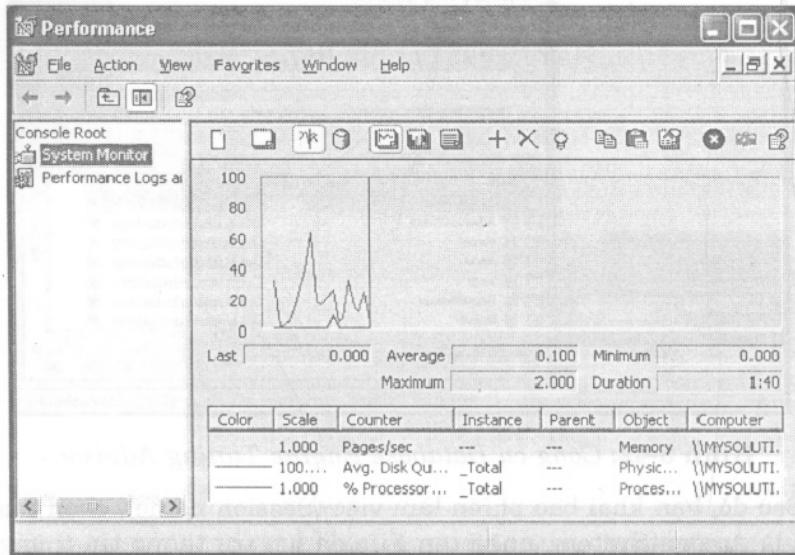


Hình 3-40: Thông tin về phát biểu SELECT.

Chương 3: Làm việc với Management Studio101 

Ngoài ra, bạn cũng có thể theo dõi hành vi thực thi trên SQL Server 2005 thông qua cửa sổ Performance Monitor.

Để sử dụng cửa sổ này, bạn chọn vào Tools | Performance Monitor, cửa sổ xuất hiện như hình 3-41.



Hình 3-41: Tiện ích Performance Monitor.

6.2. Tiện ích Database Engine Tuning Advisor

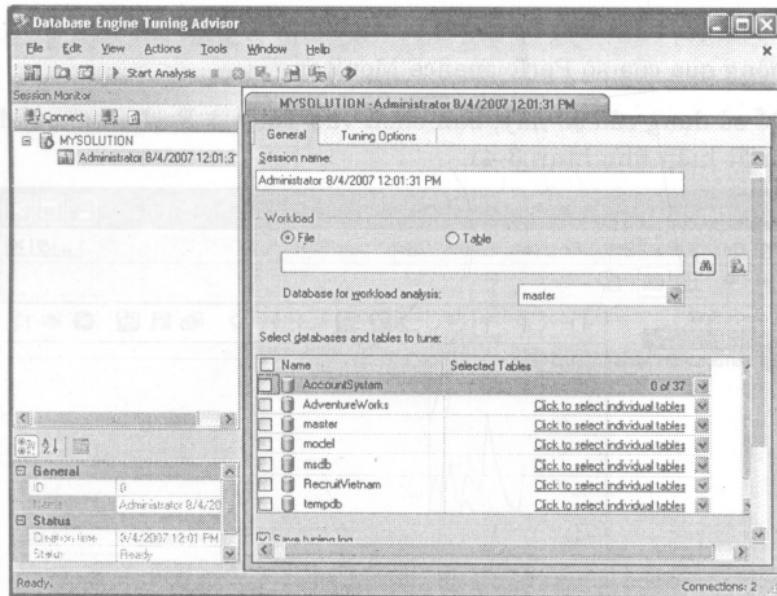
Database Engine Tuning Advisor là công cụ cho phép phân tích hiệu quả của workload khi chạy trên nhiều cơ sở dữ liệu. Trong đó, workload được định nghĩa là một tập phát biểu T-SQL thực thi trên nhiều cơ sở dữ liệu.

Công cụ này sẽ đưa ra những khuyến cáo (bao gồm clustered indexes, nonclustered indexes, indexed views và partitioning) khi người sử dụng thêm, xóa hay thay đổi về cấu trúc cơ sở dữ liệu trong Microsoft SQL Server.

Chẳng hạn, bạn chọn vào thực đơn Tools | Database Engine Tuning Advisor từ MS, cửa sổ Database Engine Tuning Advisor xuất hiện như hình 3-42.

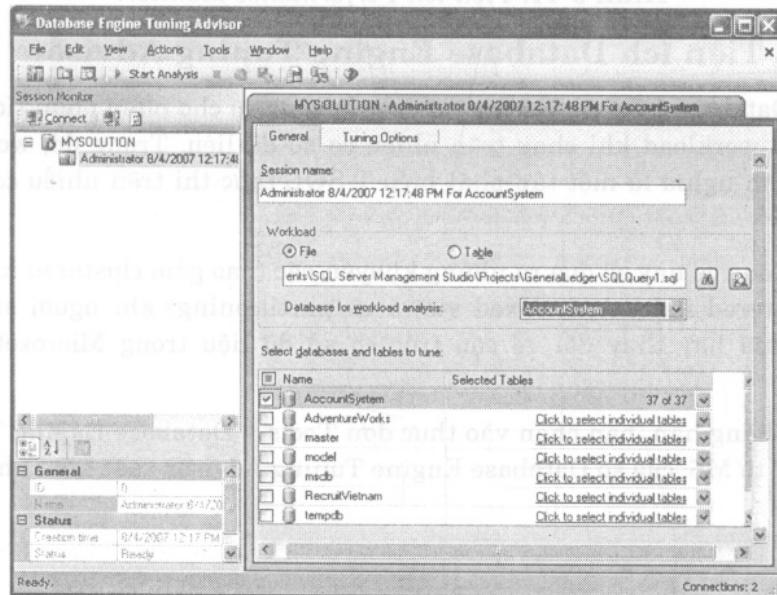


Chương 3: Làm việc với Management Studio



Hình 3-42: Công cụ Database Engine Tuning Advisor.

Sau đó, bạn khai báo phiên làm việc (Session name), chọn tên cơ sở dữ liệu là AccountSystem, nhập tên File để lưu trữ thông tin thống kê là Projects\GeneralLedger\SQLQuery1.sql như hình 3-43.



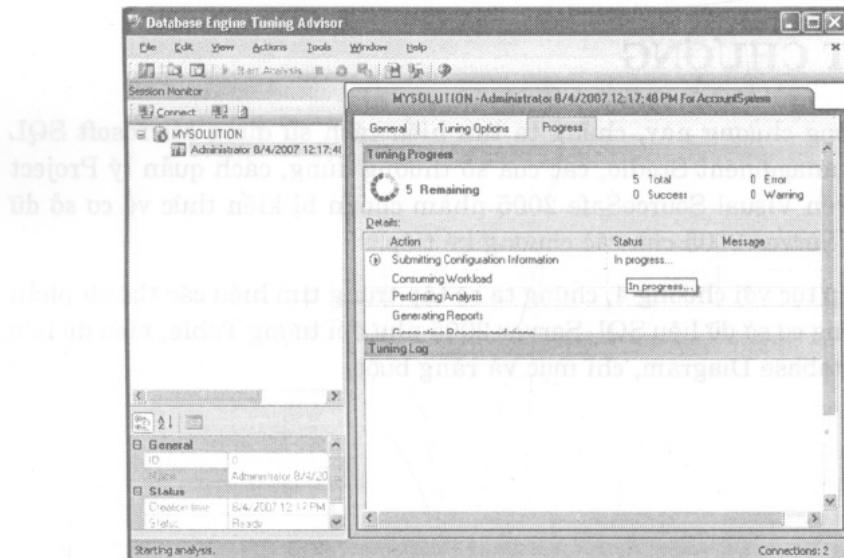
Hình 3-43: Khai báo phiên làm việc.

Chương 3: Làm việc với Management Studio

103

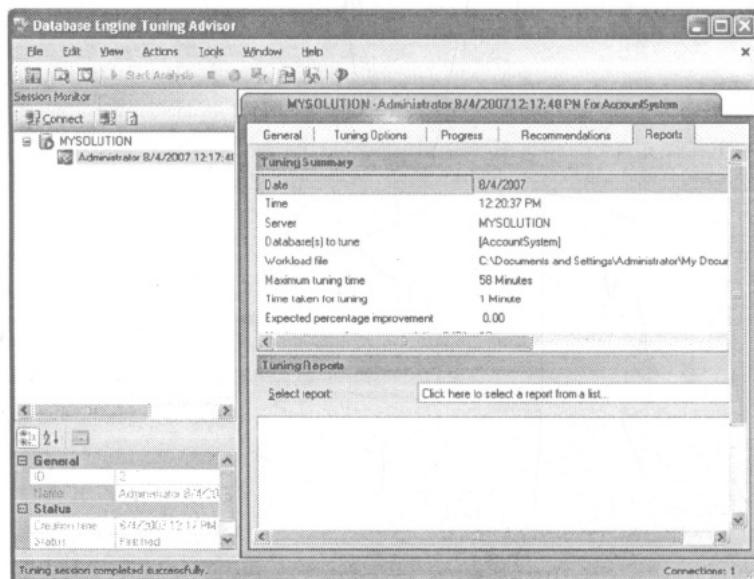


Kế đến, bạn nhấp vào nút Start Analysis trên thanh công cụ hay R-Click | Start Analysis.



Hình 3-44: Thực thi phân tích.

Bạn có thể chọn vào ngăn Report để xem thông tin thống kê sau khi phân tích thành công như hình 3-45.



Hình 3-45: Thông tin thống kê.

Chú ý: Bạn có thể tìm hiểu chi tiết về hai tiện ích này trong tập sách “Quản trị cơ sở dữ liệu SQL Server 2005” sẽ phát hành trong thời gian tới.

7. KẾT CHƯƠNG

Trong chương này, chúng ta tìm hiểu cách sử dụng Microsoft SQL Server Management Studio, các cửa sổ thường dùng, cách quản lý Project và File trên Visual SourceSafe 2005 nhằm chuẩn bị kiến thức về cơ sở dữ liệu SQL Server 2005 cho các chương kế tiếp.

Tiếp tục với chương 4, chúng ta sẽ tập trung tìm hiểu các thành phần chính trong cơ sở dữ liệu SQL Server 2005 như đối tượng Table, kiểu dữ liệu, View, Database Diagram, chỉ mục và ràng buộc.

Chương 4:

THÀNH PHẦN CHÍNH TRONG CƠ SỞ DỮ LIỆU

Tóm tắt chương 4

Chúng ta đã tìm hiểu cấu trúc của ứng dụng cơ sở dữ liệu định kèm theo sách, một trong những đối tượng chính của cơ sở dữ liệu là Table dùng để lưu trữ dữ liệu, nếu muốn đa dạng hóa phần trình bày dữ liệu thì bạn có thể sử dụng phát biểu SELECT, đối tượng View hay biểu thức bảng.

Tuy nhiên, bạn cũng có thể sử dụng thủ tục nội tại hay hàm để kết xuất dữ liệu một cách uyển chuyển hơn.

Ngoài ra, bạn cũng có thể sử dụng các đặc điểm mới như bảng dữ liệu tạm (Temporary Table) hay biểu thức bảng dữ liệu (Table Expression) sẽ giới thiệu trong chương kế tiếp để có thể tạo ra tập dữ liệu theo yêu cầu.

Để tiếp tục khai thác đặc điểm của SQL Server 2005, chúng ta tìm hiểu cách tạo và quản lý các đối tượng của cơ sở dữ liệu như: Data Type, Table, View, khai báo chỉ mục, tao quan hệ giữa các đối tượng Table bằng đối tượng Database Diagram và khai báo ràng buộc dữ liệu.

Các vấn đề chính sẽ được đề cập:

- ✓ Kiểu dữ liệu trong SQL Server 2005.
- ✓ Đối tượng cơ sở dữ liệu.
- ✓ Đối tượng Table.
- ✓ Khai báo chỉ mục.
- ✓ Đối tượng Diagram.
- ✓ Đối tượng View.
- ✓ Khai báo ràng buộc dữ liệu.

1. KIỂU DỮ LIỆU TRONG SQL SERVER 2005

Hầu hết các kiểu dữ liệu trong cơ sở dữ liệu SQL Server 2000 đều được sử dụng trong cơ sở dữ liệu SQL Server 2005. Tuy nhiên, một số kiểu dữ liệu mới được giới thiệu trong phiên bản cơ sở dữ liệu SQL Server 2005 là xml, char(max) và varchar(max).

Để tìm hiểu hết các kiểu dữ liệu trong cơ sở dữ liệu SQL Server 2005, trước tiên chúng ta hãy điểm qua những khuyến cáo khi sử dụng kiểu dữ liệu cho phù hợp với dữ liệu của thế giới thực.

Chẳng hạn, bạn có thể khai báo kiểu dữ liệu là bit hay char để lưu giới tính của nhân viên. Tùy theo quan điểm của nhà phân tích mà họ có thể chọn một trong các loại cơ sở dữ liệu trên. Ví dụ, họ sử dụng kiểu bit có hai giá trị là 0 và 1 tương ứng với Nam hay Nữ, một số người khác thì sử dụng kiểu char là ký tự M và F ứng với Nam (Male) và Nữ (Female).

Như vậy, việc quyết định chọn kiểu dữ liệu nào trong trường hợp này là tùy thuộc vào từng trường hợp cụ thể, cung cách phân tích và thiết kế hệ thống của người phát triển ứng dụng.

Một ví dụ điển giải khác về cách chọn kiểu dữ liệu ứng với dung lượng byte mà loại dữ liệu này yêu cầu cấp phát. Chẳng hạn, bạn không nên chọn kiểu dữ liệu datetime (dùng 8 byte) để lưu ngày tháng năm sinh của nhân viên, thay vào đó bạn phải chọn kiểu dữ liệu là smalldatetime (dùng 4 byte).

Trong trường hợp bạn không quan tâm đến khái niệm tối ưu hóa cơ sở dữ liệu thì bạn khai báo kiểu dữ liệu là datetime thay vì kiểu dữ liệu smalldatetime.

Tuy nhiên, chúng ta biết rằng ngày tháng năm sinh của nhân viên nằm trong khoảng 1900 trở về sau thì tại sao phải cần đến kiểu dữ liệu datetime (cho phép lưu từ 1/1/1753 đến 31/12/9999). Đối với trường hợp này chúng ta sử dụng kiểu smalldatetime (cho phép lưu từ 1/1/1900 đến 6/6/2079) là chọn lựa tốt nhất.

Chú ý: Nếu bạn xây dựng ứng dụng quản lý thông tin của dòng tộc thì bạn không thể sử dụng kiểu dữ liệu smalldatetime, thay vào đó bạn phải sử dụng kiểu dữ liệu datetime, bởi vì một số vị trí bối trong dòng tộc có thể có ngày sinh trước năm 1900.

Tóm lại, việc chọn kiểu dữ liệu phù hợp với ngữ cảnh của ứng dụng để chứa thông tin cho thế giới thực là tùy vào khả năng chọn lựa kiểu dữ liệu

Chương 4: Thành phần chính trong cơ sở dữ liệu

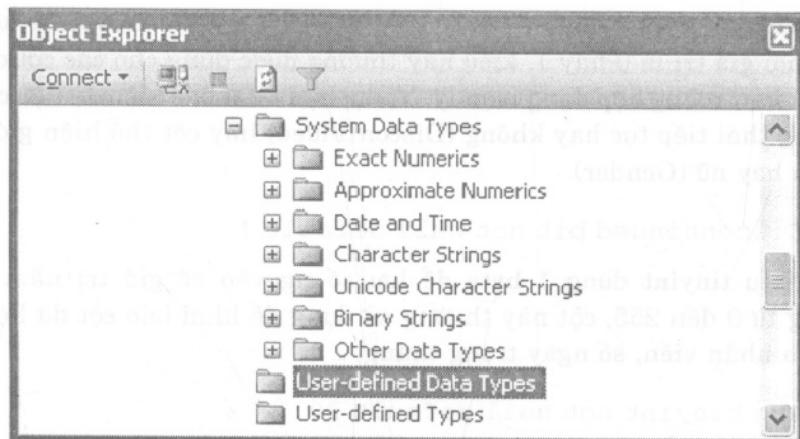
107

có sẵn của SQL Server 2005 và mục đích cuối cùng của việc chọn lựa kiểu dữ liệu đúng là nó sẽ đem lại lợi ích về lưu trữ cũng như cải thiện được khả năng truy cập dữ liệu khi thi hành.

Nhằm giới thiệu các loại kiểu dữ liệu được sử dụng trong SQL Server 2005, chúng ta hãy bắt đầu với các nhóm kiểu dữ liệu mà SQL Server 2005 đã phân loại là: System Data Types, User-Defined Data Types, User-Defined Types và Xml Schema Collections.

Trong đó, chúng ta chú trọng hai kiểu dữ liệu thường dùng là System Data Types, User-Defined Data Types.

Bạn có thể tìm thấy nhóm kiểu dữ liệu vừa trình bày ở trên trong MS như hình 4-1.



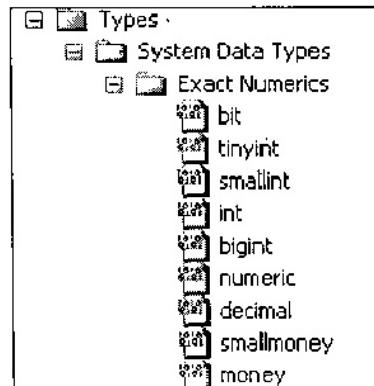
Hình 4-1: Nhóm kiểu dữ liệu.

1.1. Nhóm kiểu dữ liệu System Data Types

Nhóm kiểu dữ liệu System Data Types được chia thành nhiều loại ứng với kích thước và kiểu dữ liệu bao gồm: Exact Numerics, Approximate Numerics, Date and Time, Character Strings, Unicode Character Strings, Binary Strings và Other Data Types.

1.1.1. Loại Exact Numerics

Loại Exact Numerics bao gồm các kiểu dữ liệu là: bit, tinyint, smallint, int, bigint, numeric, decimal, smallmoney và money như hình 4-2.

**Hình 4-2: Kiểu dữ liệu Exact Numerics.**

Trong đó, kiểu dữ liệu bit là kiểu số nguyên dùng 1 byte để lưu một trong hai giá trị là 0 hay 1, kiểu này thường được dùng cho các cột dữ liệu ứng với hai trường hợp dạng luận lý. Ví dụ, bạn khai báo kiểu dữ liệu cho cột là trạng thái tiếp tục hay không (Discontinued) hay cột thể hiện giới tính là nam hay nữ (Gender).

```
Discontinued bit not null default 1
```

Kiểu tinyint dùng 1 byte để lưu số nguyên có giá trị nằm trong khoảng từ 0 đến 255, cột này thường sử dụng để khai báo cột dữ liệu như tuổi của nhân viên, số ngày trong tháng, ...

```
Age tinyint not null default 20
```

Tương tự như vậy, kiểu dữ liệu smallint dùng 2 byte để lưu trữ số nguyên trong khoảng -32,768 đến +32,767.

```
Quantity smallint not null default 0,
Price smallint not null default 0
```

Lớn hơn kiểu smallint là kiểu int, kiểu này dùng 4 byte để lưu số nguyên nằm trong khoảng -2,147,483,648 đến +2,147,483,647, bạn có thể sử dụng kiểu này cho cột số tự động, đơn giá bằng tiền VND, ...

```
Amount int not null default 0
```

Chú ý: Khi muốn khai báo cột dữ liệu chứa dữ liệu là số nguyên tự động tăng thì bạn khai báo như sau:

```
OnlineOrderNumber int identity(n,m) not null
```

Chương 4: Thành phần chính trong cơ sở dữ liệu

109

Trong đó, n là số bắt đầu, m ứng với số tăng. Ví dụ, bạn muốn cột dữ liệu tăng dần từ 1 lên bởi 1 thì khai báo:

```
OnlineOrderNumber int identity(1,1) not null
```

Kiểu dữ liệu có khoảng giá trị lớn hơn kiểu int là bigint, bigint cũng là kiểu số nguyên, kiểu này dùng 8 byte để lưu trữ số nguyên trong khoảng từ -9,223,372,036,854,775,808 đến +9,223,372,036,854,775,807.

```
TotalAmount bigint not null default 0
```

Lớn hơn kiểu bigint có kiểu numeric và decimal, hai kiểu này sử dụng 9 byte để lưu trữ số chấm động trong khoảng -10^{38+1} đến $+10^{38-1}$. Bạn có thể sử dụng kiểu decimal để lưu giá trị cho cột giá trị tiền, đơn giá, ...

```
Discount decimal not null default 0
```

Tương tự như smallint, kiểu smallmoney dùng 4 byte để lưu trữ số kiểu tiền tệ trong khoảng -214,748.3648 đến +214,748.3647. Bạn có thể sử dụng kiểu smallmoney để lưu giá trị cho cột giá trị tiền, đơn giá có dấu tiền tệ.

```
DepositAmount smallmoney not null default 0
```

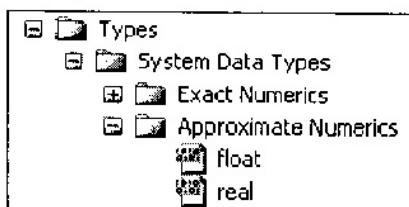
Lớn hơn kiểu smallmoney, chúng ta có kiểu money dùng 8 byte để lưu số tiền tệ trong khoảng từ -922,337,203,685,477.5808 đến +922,337,203,685,477.5807.

```
TotalDepositAmount money not null default 0
```

Chú ý: Các con số trình bày ở trên được định dạng theo kiểu Mỹ. Trong đó, dấu chấm dùng để phân cách hàng số lẻ.

1.1.2. Loại Approximate Numerics

Đại diện cho loại Approximate Numerics có hai kiểu float và real dùng để lưu số hợp lệ trong hai khoảng khác nhau như hình 4-3.



Hình 4-3: Kiểu dữ liệu Approximate Numerics.

M® 110**Chương 4: Thành phần chính trong cơ sở dữ liệu**

Trong đó, tùy số bit dùng cho phần định trị mà kiểu float dùng 4 hay 8 byte để lưu số chấm động hợp lệ trong khoảng -1.79E+308 đến -2.23E-308, giá trị 0 và từ khoảng từ 2.23E-308 đến 1.79E+308.

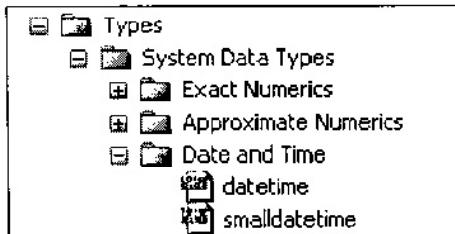
```
Price float not null default 0
```

Trong khi kiểu float dùng 4 hay 8 byte thì kiểu real dùng đúng 4 byte để lưu số chấm động hợp lệ trong khoảng -3.40E+38 đến -1.18E-38, giá trị 0 và từ khoảng 1.18E-38 đến 3.40E+38.

```
OrderNumber real not null default 0
```

1.1.3. Loại Date and Time

Nhóm Date and Time bao gồm hai kiểu dữ liệu là datetime và smalldatetime dùng để lưu dữ liệu có định dạng thời gian như hình 4-4.



Hình 4-4: Kiểu dữ liệu Date and Time.

Trong đó, kiểu datetime dùng 8 byte để lưu giá trị là thời gian xảy ra trong khoảng từ 1/1/1753 đến 31/12/9999. Bạn có thể sử dụng kiểu dữ liệu này để lưu thời gian xảy ra trước năm 1900.

```
DateOfBirth datetime not null
```

Nhỏ hơn kiểu datetime là kiểu smalldatetime, kiểu này dùng 4 byte để lưu giá trị là kiểu thời gian xảy ra trong khoảng từ 1/1/1900 đến 6/6/2079. Kiểu dữ liệu này thường dùng cho các cột dữ liệu như ngày lập hóa đơn hay ngày giao hàng.

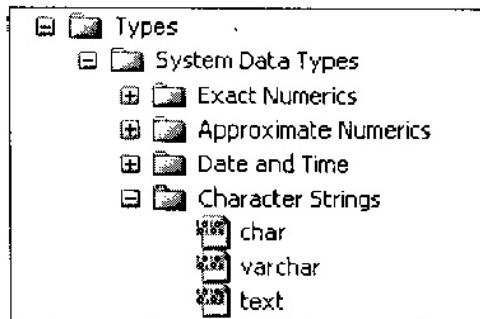
```
OrderDate smalldatetime default getdate(),
DeliveryDate smalldatetime default getdate() +10
```

1.1.4. Loại Character Strings

Đại diện cho loại Character Strings bao gồm ba kiểu char, varchar và text như hình 4-5.

Chương 4: Thành phần chính trong cơ sở dữ liệu

111

**Hình 4-5: Kiểu dữ liệu Character Strings.**

Kiểu `char` cho phép bạn lưu chuỗi có chiều dài cố định tối đa là 8000 ký tự không unicode, khi bạn sử dụng kiểu này thì đòi hỏi dữ liệu phải có chiều dài không thay đổi.

Chẳng hạn, bạn khai báo chiều dài cho cột dữ liệu chứa số chứng minh nhân dân cho công dân trên 16 tuổi của Việt Nam thì sử dụng kiểu `char` có chiều 9. Nếu khai báo chiều dài lớn hơn 9 thì khi truy cập dữ liệu bạn cần xử lý các ký tự rỗng còn lại.

```
IDCardNo char(9) not null
```

Trong khi kiểu `char` có chiều dài cố định thì kiểu `varchar` có chiều dài biến thiên và cũng cho phép khai báo chiều dài lớn nhất là 8000 ký tự không unicode.

Kiểu dữ liệu `varchar` thường được dùng để khai báo các cột dữ liệu cho địa chỉ, tên, diễn giải, ...

```
Address varchar(100) null,
FullName varchar(50) null
```

Chú ý: Nếu bạn khai báo kiểu dữ liệu `char` hay `varchar` mà không chỉ định chiều dài thì chiều mặc định là 1.

Trong trường hợp dữ liệu nhập vào có thể vượt quá 8000 ký tự, bạn có thể sử dụng từ khóa `max` (cho phép lưu dữ liệu đến $2^{31}-1$ bytes) thay vì khai báo con số cụ thể như ở trên.

```
Description varchar(max)
```

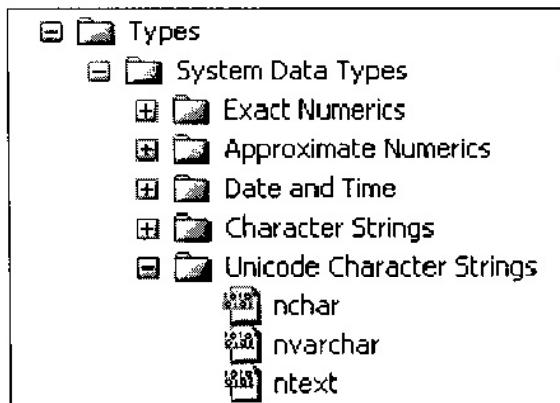
Nếu kích thước của dữ liệu lớn hơn `varchar` hay `char`, bạn có thể sử dụng kiểu dữ liệu `text`. Kiểu dữ liệu `text` cho phép lưu chuỗi có chiều dài xấp xỉ $2^{31}-1$ bytes.

Notes text

Chú ý: Trong cơ sở dữ liệu SQL Server 2000, khi kích thước kiểu dữ liệu lớn hơn 8000 ký tự, bạn phải sử dụng kiểu text. Tuy nhiên, khi làm việc với cơ sở dữ liệu SQL Server 2005, đối với trường hợp này bạn nên sử dụng kiểu dữ liệu varchar với kích thước là max thay vì dùng kiểu text.

Notes varchar (max)**1.1.5. Loại Unicode Character Strings**

Tương tự như Character Strings, Unicode Character Strings bao gồm ba kiểu nchar, nvarchar và ntext như hình 4-6.



Hình 4-6: Kiểu dữ liệu Unicode Character Strings.

Kiểu nchar cho phép bạn lưu chuỗi có chiều dài cố định tối đa là 4000 ký tự unicode. Tương tự như kiểu char, khi bạn sử dụng kiểu nchar thì độ dài dữ liệu unicode phải có chiều dài không thay đổi. Chẳng hạn, bạn khai báo chiều dài cho cột dữ liệu chứa số chứng minh nhân dân của công dân trên 16 tuổi của Trung Quốc thì sử dụng kiểu nchar có chiều dài 10.

SoCMND nchar(10) not null

Kiểu nvarchar có chiều dài biến thiên và cũng cho phép khai báo chiều dài lớn nhất là 4000 ký tự unicode.

Kiểu dữ liệu nvarchar thường được dùng để khai báo các cột dữ liệu unicode cho địa chỉ, tên, diễn giải.

```

DiaChi nvarchar (100) null,
HoVaTen nvarchar (50) null
  
```

Chương 4: Thành phần chính trong cơ sở dữ liệu

113



Chú ý: Nếu bạn khai báo kiểu dữ liệu nchar hay nvarchar mà không chỉ định chiều dài thì chiều mặc định là 1.

Đối với trường hợp dữ liệu unicode nhập vào có thể vượt quá 4000 ký tự, bạn có thể sử dụng từ khóa max (cho phép lưu dữ liệu đến $2^{31}-1$ bytes) thay vì khai báo con số cụ thể như ở trên.

DienGiai nvarchar (max)

Ngoài ra, nếu kích thước của dữ liệu lớn hơn nvarchar hay nchar, bạn có thể sử dụng kiểu dữ liệu ntext. Kiểu dữ liệu ntext cho phép lưu chuỗi có chiều dài xấp xỉ $2^{30}-1$ ký tự Unicode.

ChuThich text

Lưu ý: Trong cơ sở dữ liệu SQL Server 2000, khi kích thước kiểu dữ liệu unicode lớn hơn 4000 ký tự, bạn phải sử dụng kiểu ntext. Tuy nhiên, khi làm việc với cơ sở dữ liệu SQL Server 2005, đối với trường hợp này bạn nên sử dụng kiểu dữ liệu nvarchar với kích thước là max thay vì dùng kiểu ntext.

ChuThich nvarchar (max)

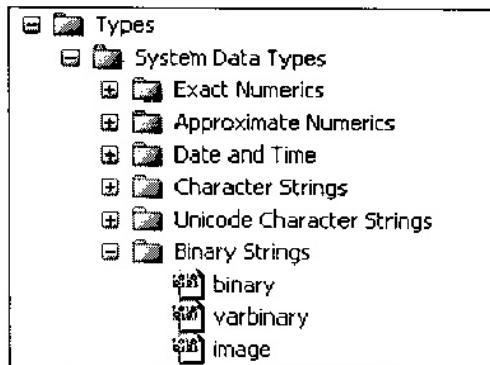
Lưu ý: Để xác định đúng chiều dài cho chuỗi, bạn cần khai báo kiểu dữ liệu tương ứng và cần phải xét đến tính thực tế theo số đông. Chẳng hạn, bạn cho rằng chiều dài của cột Address là 50 nhưng người khác cho rằng chiều dài địa chỉ có thể lớn hơn 100.

Do kích thước của kiểu dữ liệu có ảnh hưởng đến dung lượng lưu trữ và tốc độ truy cập dữ liệu, bạn cần chỉ định chiều dài thích hợp của kiểu dữ liệu để phục vụ cho số đông.

Giả sử, chúng ta đưa ra con số phù hợp là 100. Như vậy, những địa chỉ có chiều dài lớn hơn 100 ký tự thì giải quyết như thế nào. Đối với trường hợp này, chúng ta bắt buộc người sử dụng tự điều chỉnh chiều dài trong khoảng 100 ký tự khi nhập liệu.

1.1.6. Loại Binary Strings

Đại diện cho loại Binary Strings bao gồm ba kiểu binary, varbinary và image như hình 4-7.



Hình 4-7: Kiểu dữ liệu Binary Strings.

Trong đó, kiểu binary cho phép bạn lưu dữ liệu kiểu nhị phân có chiều dài cố định tối đa là 8000 bytes, khi bạn sử dụng kiểu này thì đòi hỏi dữ liệu phải có chiều dài không thay đổi.

 Password binary (15) not null

Nếu kiểu binary có chiều dài cố định thì kiểu varbinary có chiều dài biến thiên và cũng cho phép khai báo chiều dài lớn nhất là 8000 bytes.

 Description varbinary (200)

Trong trường hợp dữ liệu lưu vào có thể vượt quá 8000 bytes, bạn có thể sử dụng từ khóa max (cho phép lưu dữ liệu nhị phân đến $2^{31}-1$ bytes) thay vì khai báo con số cụ thể như ở trên.

 Description varbinary (max)

Nếu kích thước của dữ liệu lớn hơn varbinary hay binary, bạn có thể sử dụng kiểu dữ liệu image. Kiểu dữ liệu image cho phép lưu dữ liệu nhị phân có chiều dài xấp xỉ $2^{31}-1$ bytes.

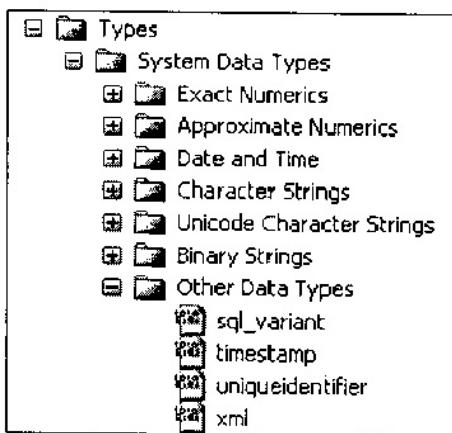
 Notes image

Chú ý: Trong cơ sở dữ liệu SQL Server 2000, khi kích thước kiểu dữ liệu nhị phân lớn hơn 8000 ký tự, bạn phải sử dụng kiểu image. Tuy nhiên, khi làm việc với cơ sở dữ liệu SQL Server 2005, đối với trường hợp này bạn nên sử dụng kiểu dữ liệu varbinary với kích thước là max thay vì dùng kiểu image.

 Notes varbinary (max)

1.1.7. Loại Other Data Types

Trong trường hợp bạn không xác định được dữ liệu sẽ lưu thuộc loại nào ở trên thì sử dụng kiểu dữ liệu `sql_variant`. Khi khai báo kiểu dữ liệu này cho cột dữ liệu, bạn có thể lưu mọi loại dữ liệu vào cột này ngoại trừ kiểu `text`, `ntext`, `image`, `timestamp` và `sql_variant` như hình 4-8.



Hình 4-8: Kiểu dữ liệu khác.

Nếu muốn kiểu dữ liệu nhị phân tự động tạo ra duy nhất trong cơ sở dữ liệu thì bạn hãy sử dụng kiểu `timestamp`. Kiểu `timestamp` dùng kích thước 8 bytes để lưu dữ liệu nhị phân.

`OrderNumber timestamp`

Tầm vực rộng hơn kiểu `timestamp` là kiểu `uniqueidentifier`, kiểu `uniqueidentifier` dùng kiểu dữ liệu nhị phân với kích thước 16 bytes, giá trị do kiểu này tự động tạo ra sẽ là duy nhất trong cơ sở dữ liệu.

Chẳng hạn, bạn khai báo cột dữ liệu có tên `IDCongDanToanCau` như sau:

`IDCongDanToanCau uniqueidentifier`

Một trong kiểu dữ liệu mới được giới thiệu trong SQL Server 2005 là `Xml`. Kiểu dữ liệu này cho phép chúng ta lưu trữ dữ liệu định dạng `Xml`.

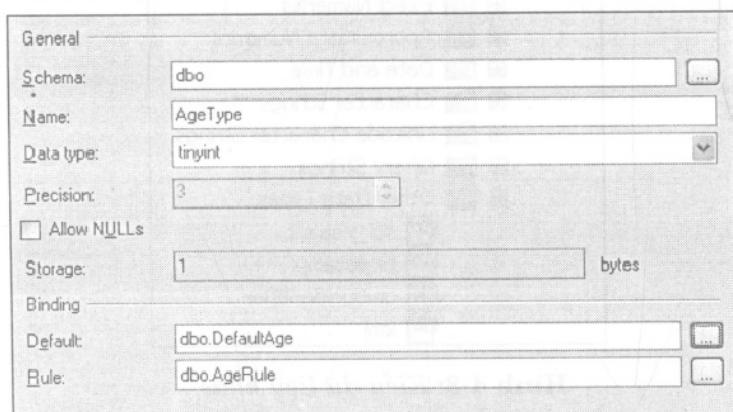
Như vậy, nếu bạn có dữ liệu là chuỗi định dạng `Xml` thì sử dụng kiểu dữ liệu `Xml` như sau:

`SpecificateOfProduct Xml`

1.2. Nhóm kiểu dữ liệu User-Defined Data Types

Dựa trên kiểu dữ liệu của hệ thống, bạn có thể định nghĩa lại một số thuộc tính cho kiểu dữ liệu này như giá trị mặc định, giới hạn giá trị trong khoảng cho phép.

Chẳng hạn, bạn muốn giới hạn kiểu dữ liệu AgeType dựa trên kiểu tinyint với giá trị mặc định là 20 và giá trị nằm trong khoảng từ 1 đến 100 thì khai báo như hình 4-9.



Hình 4-9: Khai báo kiểu AgeType.

Chú ý: Để tạo mới kiểu dữ liệu User-Defined Data Types, chọn ngăn Programming | Types | User-Defined Data Types | R-Click | New User-Defined Data Type, cửa sổ xuất hiện như hình trên.

Chúng ta sẽ tham khảo cách khai báo tạo đối tượng Default trong chương kế tiếp. Tuy nhiên, trong trường hợp này bạn cần tìm hiểu sơ lược đối tượng DefaultAge ứng với giá trị mặc định là 20 được khai báo bằng phát biểu CREATE với cú pháp như ví dụ 4-1.

Ví dụ 4-1: Khai báo đối tượng Default

```
CREATE DEFAULT DefaultAge AS 20
```

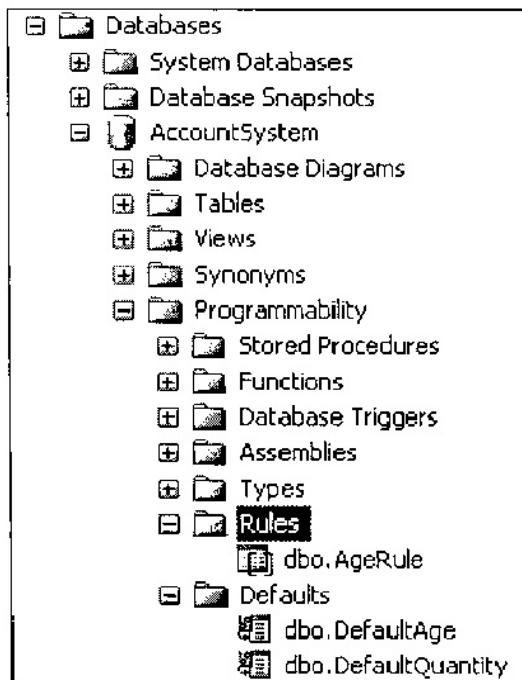
Tương tự như vậy, bạn khai báo cú pháp để tạo đối tượng Rule ứng với điều kiện tuổi nằm trong khoảng từ 1 đến 100 như ví dụ 4-2.

Ví dụ 4-2: Khai báo đối tượng Rule

```
CREATE RULE AgeRule
AS
@AgeRule BETWEEN 1 AND 100;
```

Chương 4: Thành phần chính trong cơ sở dữ liệu**117**

Lưu ý: Sau khi khai báo đối tượng Rule và Default, bạn có thể tìm thấy hai đối tượng này trong cơ sở dữ liệu như hình 4-10.



Hình 4-10: Đối tượng Rule và Default.

Chú ý: Từ khóa dbo (database owner) nằm trước tên đối tượng là chủ nhân của cơ sở dữ liệu.

Như vậy, khi thiết kế bảng dữ liệu trong SQL Server 2005, thay vì bạn khai báo cột Age có kiểu tinyint

Age tinyint not null default 20

thì bạn có thể khai báo cột dữ liệu Age là AgeType như sau:

Age AgeType

Bạn có thể tìm thấy kiểu dữ liệu AgeType vừa với khai báo trong khi thiết kế bảng dữ liệu trong MS như hình 4-11.

Column Name	Data Type	Allow Nulls
BitType	bit	<input checked="" type="checkbox"/>
TinyIntType	tinyint	<input checked="" type="checkbox"/>
SmallIntType	smallint	<input checked="" type="checkbox"/>
IntType	int	<input checked="" type="checkbox"/>
BigInt	bigint	<input checked="" type="checkbox"/>
NumericType	numeric(18, 0)	<input checked="" type="checkbox"/>
DecimalType	decimal(18, 0)	<input checked="" type="checkbox"/>
SmallMoneyType	smallmoney	<input checked="" type="checkbox"/>
MoneyType	money	<input checked="" type="checkbox"/>
FloatingPointType	float	<input checked="" type="checkbox"/>
RealType	real	<input checked="" type="checkbox"/>
DateTimeType	datetime	<input checked="" type="checkbox"/>
SmallDateTime	smalldatetime	<input checked="" type="checkbox"/>
IDCard	char(9)	<input checked="" type="checkbox"/>
Tel	varchar(15)	<input checked="" type="checkbox"/>
FullName	nvarchar(50)	<input checked="" type="checkbox"/>
Password	binary(50)	<input checked="" type="checkbox"/>
UniversalId	uniqueidentifier	<input checked="" type="checkbox"/>
SpecificationOfProduct	xml	<input checked="" type="checkbox"/>
Age	AgeType:tinyint	<input checked="" type="checkbox"/>

Hình 4-11: Kiểu dữ liệu AgeType.

2. ĐỐI TƯỢNG TABLE

Sau khi khám phá kiểu dữ liệu được sử dụng trong cơ sở dữ liệu, chúng ta tiếp tục tìm hiểu các đối tượng của cơ sở dữ liệu SQL Server 2005.

Các đối tượng chính trong cơ sở dữ liệu được kể đến là Table, Diagram, View, Stored Procedure, Function và người sử dụng (Users).

Mỗi đối tượng có nhiệm vụ riêng của chúng, chẳng hạn đối tượng Table dùng để lưu dữ liệu, bạn có thể khai báo một số thuộc tính cho đối tượng như: Index, Trigger, Constraint.

Chương 4: Thành phần chính trong cơ sở dữ liệu

119

Để tạo đối tượng Table, bạn có thể sử dụng MS hay phát biểu SQL dạng CREATE TABLE. Tuy nhiên, bạn có thể tìm hiểu cách tạo đối tượng Table bằng phát biểu CREATE TABLE trong chương kế tiếp.

Nếu bạn tạo đối tượng Table bằng MS thì chọn vào tên cơ sở dữ liệu | Tables | R-Click | New Table | cửa sổ xuất hiện cho phép bạn khai báo cột dữ liệu như hình 4-12.

Column Name	Data Type	Allow Nulls
ReceiptBatchNo	varchar(50)	<input type="checkbox"/>
ReceiptBatchDate	smalldatetime	<input checked="" type="checkbox"/>
UserName	varchar(10)	<input checked="" type="checkbox"/>
Discontinued	bit	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

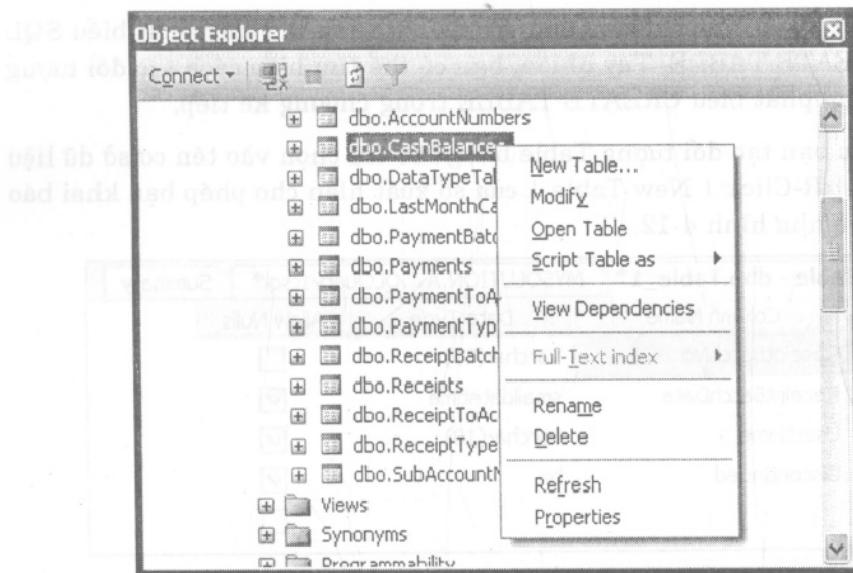
Hình 4-12: Tạo đối tượng Table.

Chú ý: Để tạo khóa chính cho cột trong bảng dữ liệu, bạn chỉ cần chọn tên cột và nhấn nút có biểu tượng chìa khóa trên thanh công cụ.

Column Name	Data Type	Allow Nulls
ReceiptBatchNo	varchar(10)	<input type="checkbox"/>
ReceiptBatchDate	smalldatetime	<input checked="" type="checkbox"/>
UserName	varchar(10)	<input type="checkbox"/>
Discontinued	bit	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

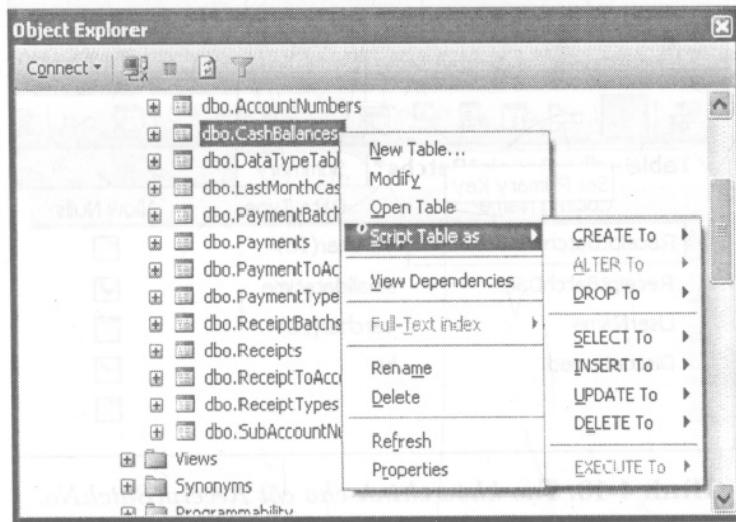
Hình 4-13: Tạo khóa chính cho cột ReceiptBatchNo.

Chú ý: Bạn có thể thực hiện các thao tác cho đối tượng Table như: Xóa (Delete), đổi tên Table (Rename), xem dữ liệu (Open Table), định nghĩa lại cấu trúc (Modify) hay tạo Script (Script Table As) trong MS bằng cách R-Click trên tên bảng dữ liệu như hình 4-14.



Hình 4-14: Thực hiện các thao tác khác cho đối tượng Table.

Đối với trường hợp thao tác đến dữ liệu trên đối tượng Table, chúng ta có thể sử dụng phát biểu SQL dạng DML (Data Manipulation Language) như: SELECT, DELETE, UPDATE, INSERT bằng MS như hình 4-15.



Hình 4-15: Tạo các phát biểu SQL dạng DML.

Lưu ý: Bạn có thể tìm hiểu chi tiết về phát biểu SQL dạng CREATE TABLE trong những chương kế tiếp.

Chương 4: Thành phần chính trong cơ sở dữ liệu

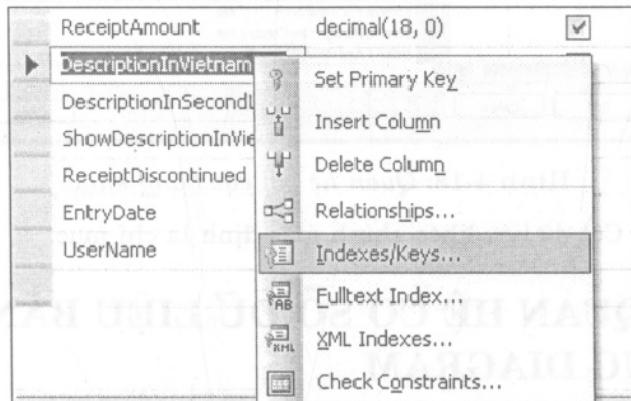
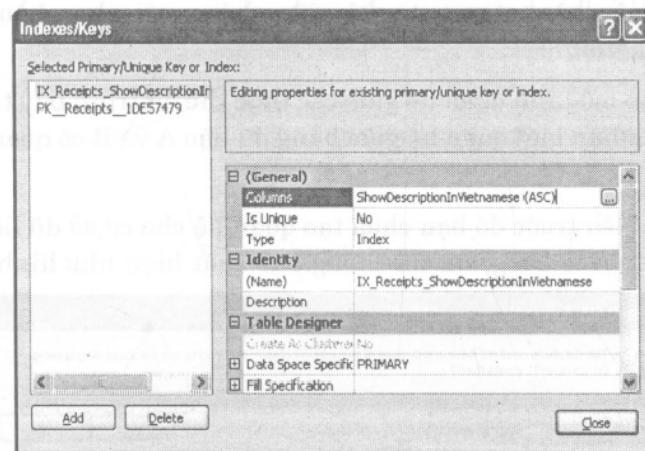
121 M®

3. KHAI BÁO CHỈ MỤC

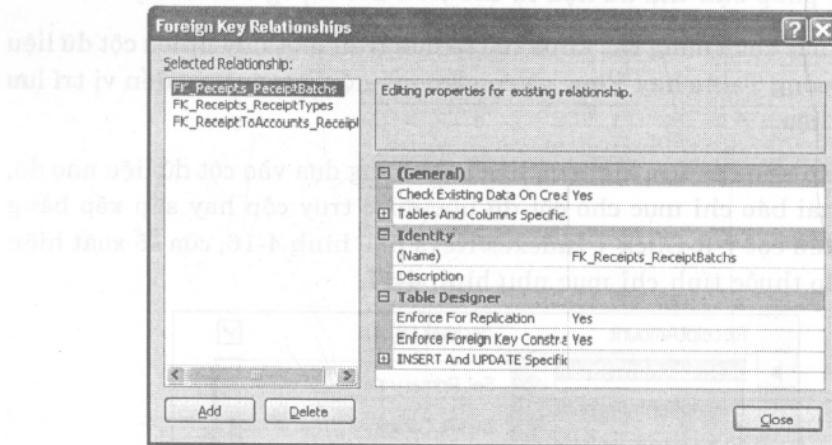
Tương tự như chỉ mục trên cuốn sách, chỉ mục (Indexes) trong cơ sở dữ liệu cho phép bạn tìm dữ liệu cụ thể trên đối tượng Table hay View.

Chỉ mục chứa đựng các khóa tạo ra dựa trên một hay nhiều cột dữ liệu trong đối tượng Table hay View và con trỏ mà nó được ánh xạ đến vị trí lưu trữ của dữ liệu.

Nếu có nhu cầu tìm kiếm dữ liệu trên bảng dựa vào cột dữ liệu nào đó, bạn cần khai báo chỉ mục cho cột dữ liệu được truy cập hay sắp xếp bằng cách chọn tên cột | R-Click | Indexes/Keys như hình 4-16, cửa sổ xuất hiện rồi khai báo thuộc tính chỉ mục như hình 4-17.

**Hình 4-16: Khai báo chỉ mục.****Hình 4-17: Khai báo chỉ mục.**

Trong trường hợp bạn muốn xem quan hệ của bảng này với các bảng dữ liệu khác thì chọn vào Relationships. Chẳng hạn, trong trường hợp này, bảng Receipts có 3 quan hệ đến các bảng tương ứng là ReceiptBatchs, ReceiptTypes và ReceiptToAccounts như hình 4-18.



Hình 4-18: Quan hệ với các bảng khác.

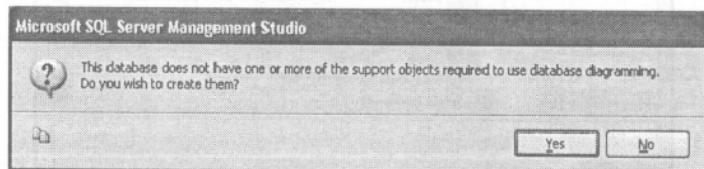
Chú ý: Cột dữ liệu khóa chính mặc định là chỉ mục.

4. TẠO QUAN HỆ CƠ SỞ DỮ LIỆU BẰNG ĐỐI TƯỢNG DIAGRAM

Sau khi chúng ta hoàn tất việc thiết kế các bảng dữ liệu liên quan, bạn có thể tiến hành tạo quan hệ giữa chúng với nhau bằng đối tượng Database Diagram.

Dựa vào mô hình quan hệ giữa các thực thể đã trình bày trong chương 2, bạn có thể nhận biết quan hệ giữa bảng dữ liệu A và B có quan hệ 1-1, 1-n hay n-n.

Chú ý: Nếu trước đó bạn chưa tạo quan hệ cho cơ sở dữ liệu, khi chọn vào ngăn Database Diagram thì thông báo xuất hiện như hình 4-19.

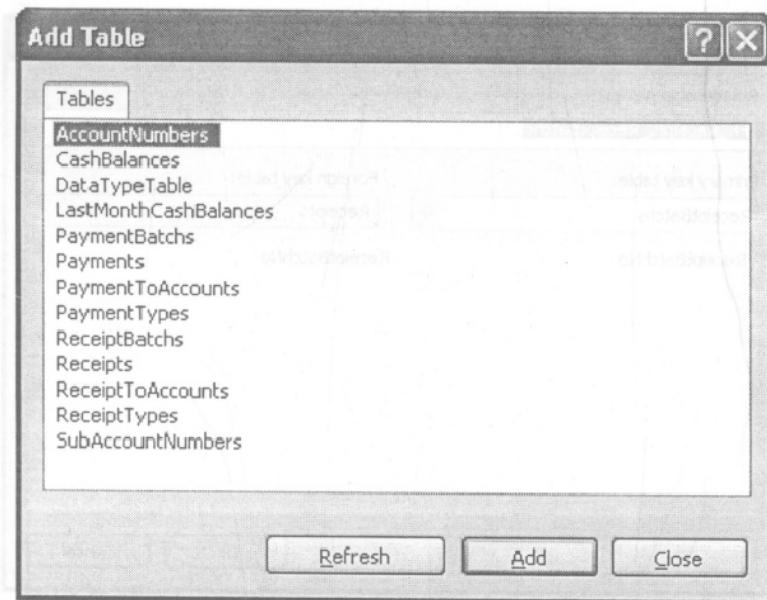


Hình 4-19: Thông báo tạo đối tượng hỗ trợ cho quan hệ.

Chương 4: Thành phần chính trong cơ sở dữ liệu

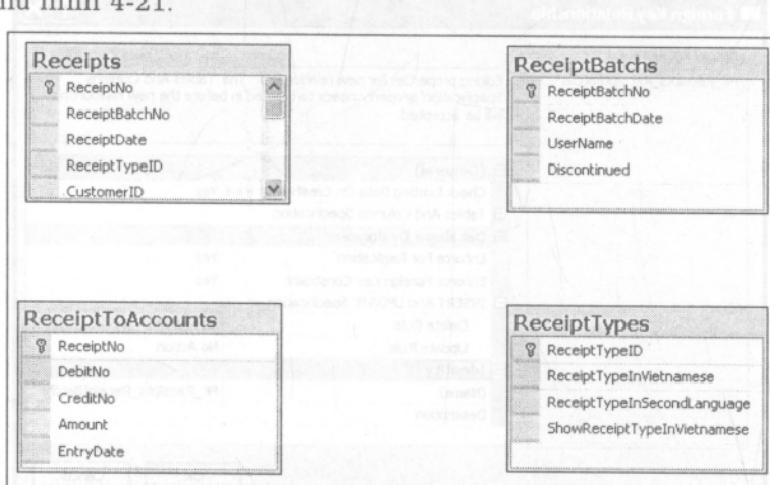
123

Sau khi chọn Yes, bạn có thể tạo quan hệ cơ sở dữ liệu bằng cách chọn vào ngăn Database Diagrams | New Database Diagram, cửa sổ yêu cầu chọn tên bảng dữ liệu xuất hiện như hình 4-20.



Hình 4-20: Danh sách bảng dữ liệu đang có.

Chọn những bảng dữ liệu có nhu cầu tạo quan hệ rồi nhấn nút Add, kể đến bạn nhấn nút Close để kết thúc, cửa sổ Database Diagram bây giờ xuất hiện như hình 4-21.

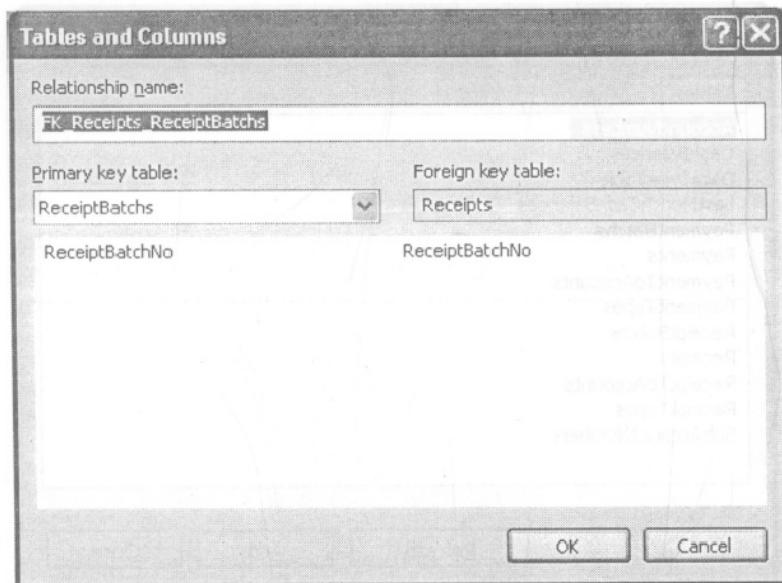


Hình 4-21: Bảng dữ liệu đã chọn trong Database Diagram.

M® 124

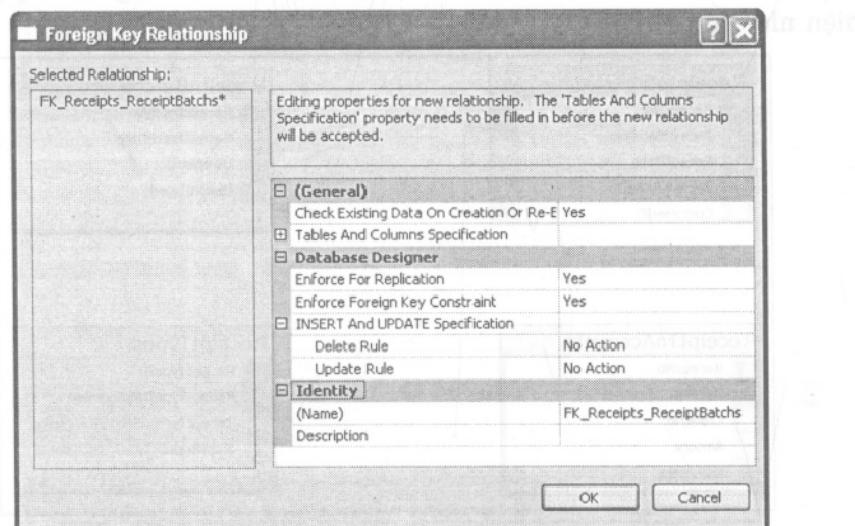
Chương 4: Thành phần chính trong cơ sở dữ liệu

Bằng cách chọn cột dữ liệu là khóa chính từ bảng thứ nhất rồi kéo và thả vào cột dữ liệu ứng với khóa ngoại của bảng thứ hai, quan hệ sẽ thiết lập tương tự như hình 4-22.



Hình 4-22: Thiết lập quan hệ giữa hai bảng.

Sau khi xác nhận cột dữ liệu cho khóa chính và khóa ngoại, nhấn nút OK thì cửa sổ thuộc tính của quan hệ xuất hiện như hình 4-23.



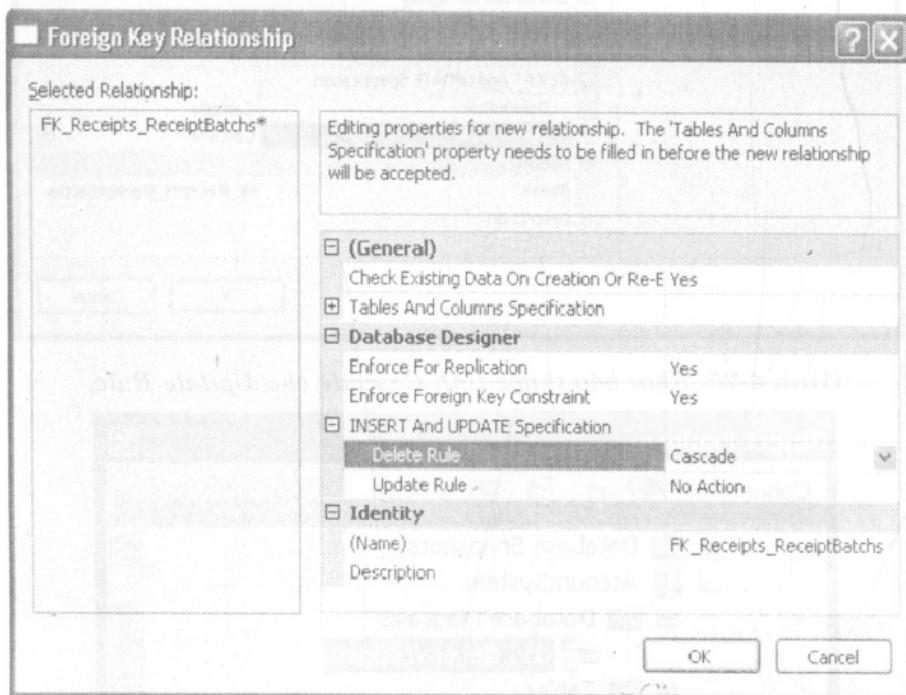
Hình 4-23: Thuộc tính của quan hệ.

Chương 4: Thành phần chính trong cơ sở dữ liệu

125

Chú ý: Bạn nên thiết lập quan hệ giữa các bảng với nhau trước khi thêm, cập nhật, xóa hay truy vấn dữ liệu. Trong trường hợp dữ liệu đã tồn tại, để thiết lập quan hệ bạn nên kiểm tra tính thống nhất của dữ liệu trong các bảng dữ liệu mà bạn sẽ thiết lập quan hệ.

Để cho phép xóa mẫu tin trong bảng con có quan hệ và các mẫu tin tương ứng quan hệ sẽ bị xóa trong bảng cha, bạn có thể chọn giá trị Cascade trong thuộc tính INSERT and UPDATE Specification | Delete Rule như hình 4-24.

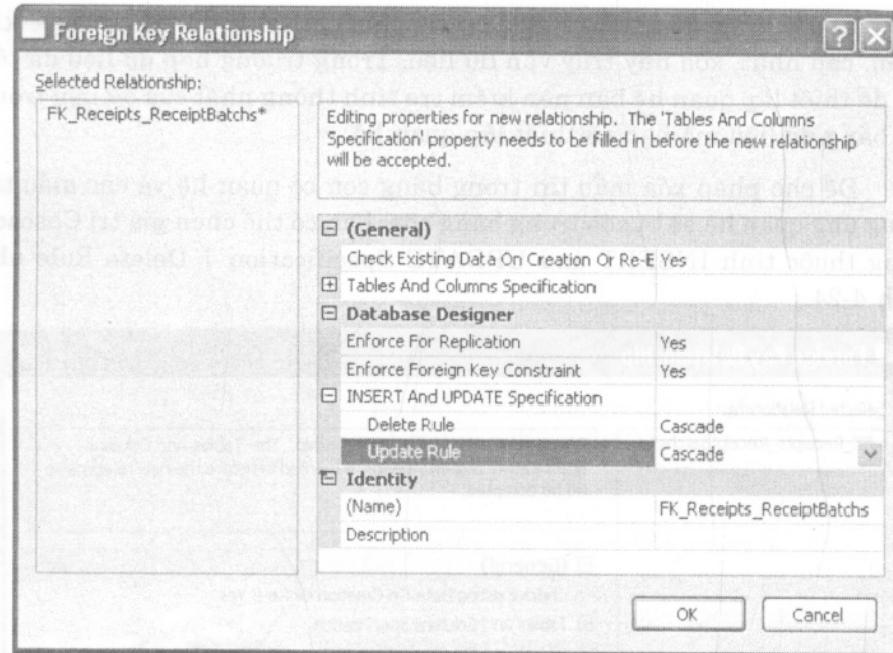
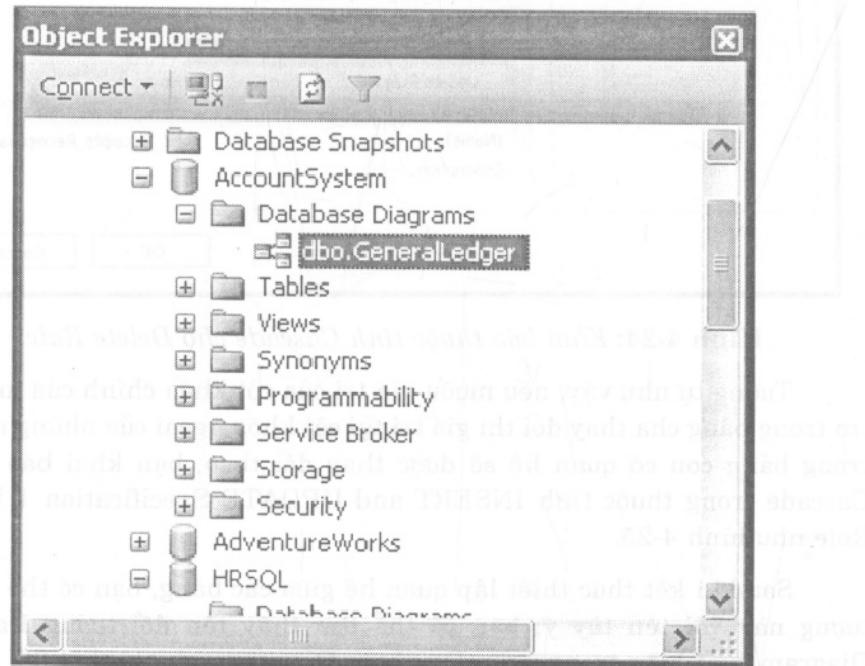


Hình 4-24: Khai báo thuộc tính Cascade cho Delete Rule.

Tương tự như vậy, nếu muốn giá trị của cột khóa chính của một mẫu tin trong bảng cha thay đổi thì giá trị tại cột khóa ngoại của những mẫu tin trong bảng con có quan hệ sẽ được thay đổi theo, bạn khai báo giá trị Cascade trong thuộc tính INSERT and UPDATE Specification | Update Rule như hình 4-25.

Sau khi kết thúc thiết lập quan hệ giữa các bảng, bạn có thể lưu đối tượng này với tên tùy ý; bạn có thể tìm thấy tên đối tượng Database Diagram xuất hiện trong ngăn Database Diagrams như hình 4-26.

M® 126

Chương 4: Thành phần chính trong cơ sở dữ liệu**Hình 4-25:** Khai báo thuộc tính Cascade cho Update Rule.**Hình 4-26:** Thiết lập quan hệ cơ sở dữ liệu.

Chương 4: Thành phần chính trong cơ sở dữ liệu

127

Chú ý: Chúng ta có thể tạo quan hệ giữa các bảng dữ liệu thông qua các cột khóa chính và khóa ngoại khi thực thi phát biểu CREATE TABLE bằng mệnh đề REFERENCES sẽ được trình bày trong những chương kế tiếp.

5. ĐỐI TƯỢNG VIEW

Chúng ta đã tìm hiểu cấu trúc của cơ sở dữ liệu được kèm theo sách dùng để tham khảo; để có được cấu trúc đó chúng ta phải thực hiện qua nhiều công đoạn từ quá trình thu thập thông tin đến phân tích hệ thống và chuẩn hóa (trình bày trong chương 1) để kết quả sau cùng là cơ sở dữ liệu bao gồm các bảng dữ liệu như đã trình bày trong chương 2.

Nhằm thực hiện ba bước chuẩn hóa, chúng ta đã phân rã thông tin của một đối tượng thành nhiều thực thể khác nhau. Khi cần kết xuất dữ liệu với định dạng mà dữ liệu của chúng được lấy ra từ nhiều bảng, bạn có thể sử dụng phát biểu SQL dạng SELECT với các mệnh đề thường dùng như: JOIN, WHERE, GROUP BY và ORDER BY.

View là đối tượng cơ sở dữ liệu cho phép bạn kết hợp, sắp xếp, tính toán, trích lọc dữ liệu bằng cách sử dụng phát biểu SQL dạng SELECT với các mệnh đề JOIN, WHERE, GROUP BY và ORDER BY.

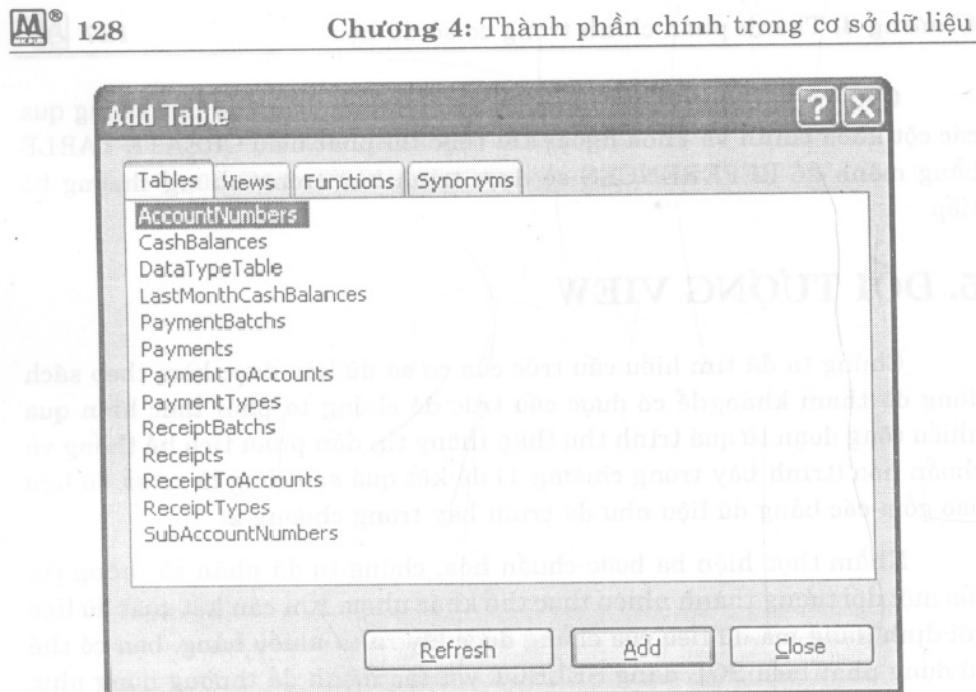
Như vậy, View được biết đến như bảng ảo, khi gọi nó sẽ trả về dữ liệu theo tiêu chí truy vấn mà bạn đã định nghĩa bằng phát biểu SQL.

Nếu bạn sử dụng ngôn ngữ lập trình để kết nối và truy vấn dữ liệu với phát biểu SQL như trên, khi truyền vào SQL Server chúng phải được kiểm tra cú pháp rồi biên dịch trước khi thực thi để trả về kết quả. Thay vì sử dụng phát biểu SQL như trên, bạn có thể sử dụng đối tượng View.

Để khai báo View, trước tiên bạn phải xác định bảng dữ liệu cần kết hợp, xem xét các cột dữ liệu cần trình bày, sau đó bạn sử dụng MS hay phát biểu CREATE VIEW.

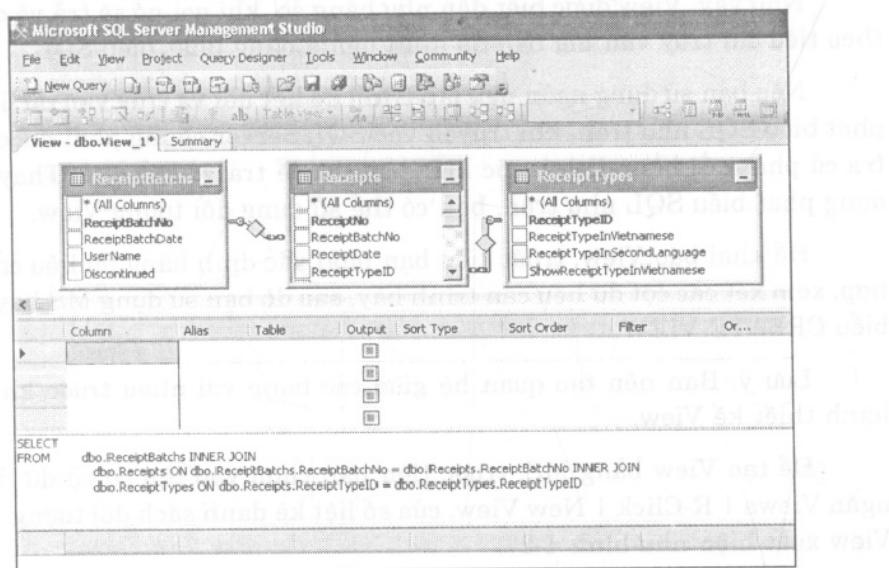
Lưu ý: Bạn nên tạo quan hệ giữa các bảng với nhau trước khi tiến hành thiết kế View.

Để tạo View bằng MS, trước tiên bạn chọn vào tên cơ sở dữ liệu | ngăn Views | R-Click | New View, cửa sổ liệt kê danh sách đối tượng Table, View xuất hiện như hình 4-27.



Hình 4-27: Danh sách bảng dữ liệu.

Chọn tên bảng dữ liệu hay View, nhấn nút Add để thêm chúng vào màn hình thiết kế, nhấn nút Close, bạn có thể tìm thấy danh sách bảng vừa chọn xuất hiện gồm 4 phần như hình 4-28.

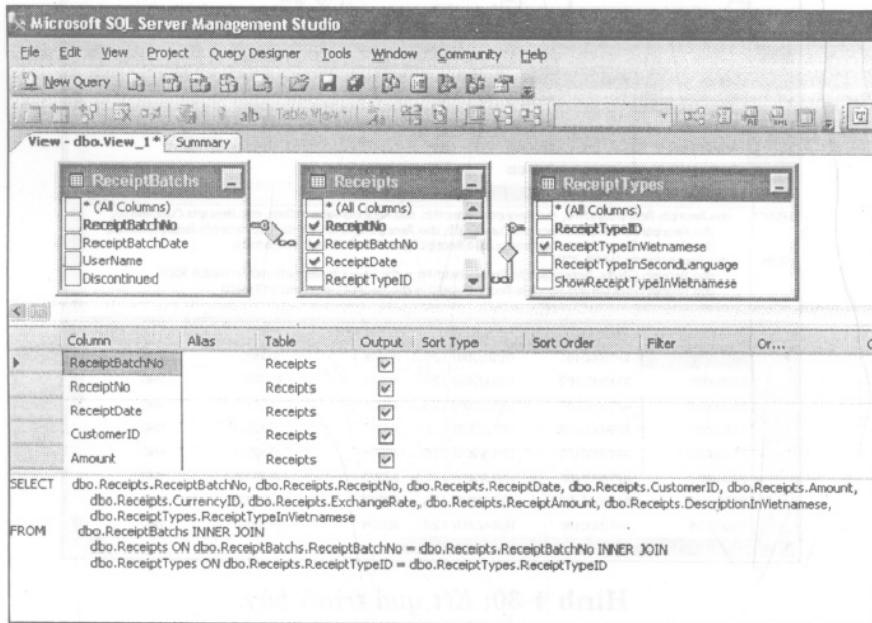


Hình 4-28: Danh sách bảng dữ liệu trong View.

Chương 4: Thành phần chính trong cơ sở dữ liệu

129 M®

Để liệt kê cột dữ liệu cần trình bày, bạn check vào cột dữ liệu tương ứng trên bảng, lập tức tên cột dữ liệu xuất hiện trong phần thứ 3, phần phát biểu SQL tự phát sinh như hình 4-29.

**Hình 4-29: Thiết kế View.**

Trong trường hợp cần kiểm tra tính chính xác của dữ liệu, bạn có thể nhấn vào biểu tượng ! trên thanh công cụ, kết quả sẽ trình bày trong phần thứ 4 như hình 4-30.

Sau khi khai báo kết thúc, bạn cần lưu View với tên tùy ý. Tuy nhiên, để tránh trùng lặp với tên bảng dữ liệu, bạn nên thêm ký tự v hay vw trước tên của View. Chẳng hạn, bạn lưu View trên với tên vwReceipts.

Bạn có thể tìm thấy đối tượng View vừa tạo xuất hiện trong ngăn Views như hình 4-31.

Bây giờ, bạn có thể thay đổi cấu trúc của View bằng cách chọn vào tên View | R-Click | Modify, mở View rồi chọn Open View.

Tương tự như Table, bạn có thể tạo các kịch bản SQL bằng cách chọn vào thực đơn “Script View as”.

M® 130

Chương 4: Thành phần chính trong cơ sở dữ liệu

Object Explorer Details

ReceiptBatchs

- * (All Columns)
- ReceiptBatchID
- ReceiptBatchDate
- UserName
- Discontinued

Receipts

- * (All Columns)
- ReceiptID
- ReceiptBatchID
- ReceiptDate
- ReceiptTypeID

ReceiptTypes

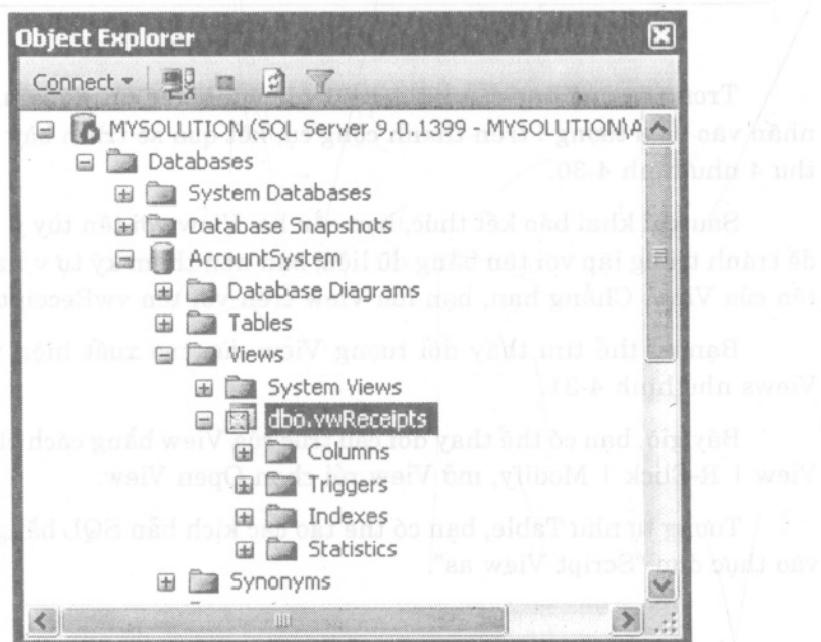
- * (All Columns)
- ReceiptTypeID
- ReceiptTypeInVietnamese
- ReceiptTypeInSecondLanguage
- ShowReceiptTypeInVietnamese

Column	Alias	Table	Output	Sort Type	Sort Order	Filter
ReceiptBatchID		Receipts	<input checked="" type="checkbox"/>			
ReceiptID		Receipts	<input checked="" type="checkbox"/>			
ReceiptDate		Receipts	<input checked="" type="checkbox"/>			
CustomerID		Receipts	<input checked="" type="checkbox"/>			

```

SELECT dbo.Receipts.ReceiptBatchID, dbo.Receipts.ReceiptID, dbo.Receipts.ReceiptDate, dbo.Receipts.CustomerID,
       dbo.Receipts.Amount, dbo.Receipts.CurrencyID, dbo.Receipts.ExchangeRate, dbo.Receipts.ReceiptAmount,
       dbo.Receipts.DescriptionInVietnamese, dbo.ReceiptTypes.ReceiptTypeInVietnamese
FROM dbo.ReceiptBatchs INNER JOIN
      dbo.Receipts ON dbo.ReceiptBatchs.ReceiptBatchID = dbo.Receipts.ReceiptBatchID INNER JOIN
      dbo.ReceiptTypes ON dbo.Receipts.ReceiptTypeID = dbo.ReceiptTypes.ReceiptTypeID
  
```

ReceiptBatchID	ReceiptID	ReceiptDate	CustomerID	Amount	CurrencyID
BR000001	RPT0000101	10/10/2007 12:0...	A0001	500000	VND
BR000001	RPT0000102	10/11/2007 12:0...	A0002	450000	VND
BR000002	RPT0000103	10/1/2007 12:0...	A0001	105000	VND
BR000002	RPT0000104	10/1/2007 12:0...	A0003	1500000	VND
BR000003	RPT0000105	10/1/3/2007 12:0...	A0004	3000000	VND
BR000003	RPT0000106	10/1/3/2007 12:0...	A0003	2000000	VND
BR000003	RPT0000107	10/1/3/2007 12:0...	A0002	1200000	VND
BR000004	RPT0000108	10/1/4/2007 12:0...	A0004	1450000	VND

Hình 4-30: Kết quả trình bày.**Hình 4-31: Tạo View thành công.**

Chương 4: Thành phần chính trong cơ sở dữ liệu

131

Ngoài cách tạo View bằng trình MS, bạn có thể tạo đối tượng View bằng cách sử dụng phát biểu CREATE VIEW sẽ được trình bày trong chương kế tiếp.

Bạn có thể tạo kịch bản View bằng cách chọn vào tên View | R-Click | Script View As | Create To | New Query Editor Windows, cửa sổ Query của SQL Server xuất hiện như hình 4-32.

```

MYSOLUTION.A...QLQuery3.sql* MYSOLUTION.Ad...QLQuery1.sql*
USE [AccountSystem]
GO
/******** Object: View [dbo].[vwCustomers]
Script Date: 08/28/2007 19:43:21 *****/
|
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE VIEW [dbo].[vwCustomers]
AS
SELECT * FROM Customers
WHERE ProvinceID='HCM'

```

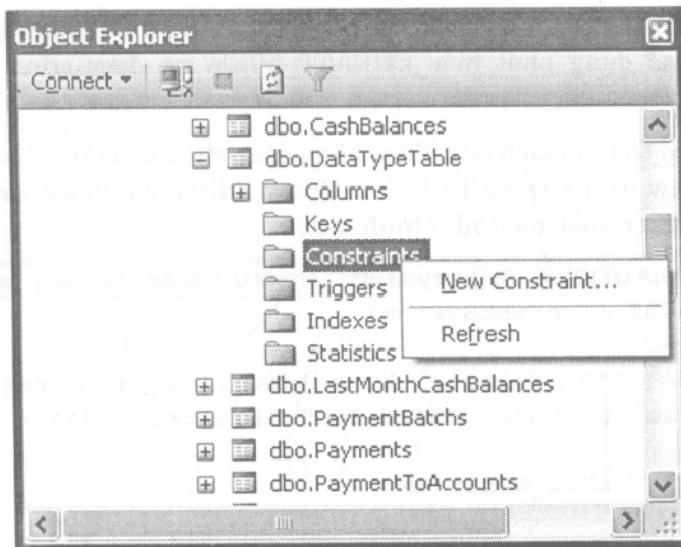
Hình 4-32: Cấu trúc View bằng phát biểu SQL.

Chú ý: Bạn có thể tìm hiểu chi tiết về phát biểu CREATE VIEW trong những chương kế tiếp.

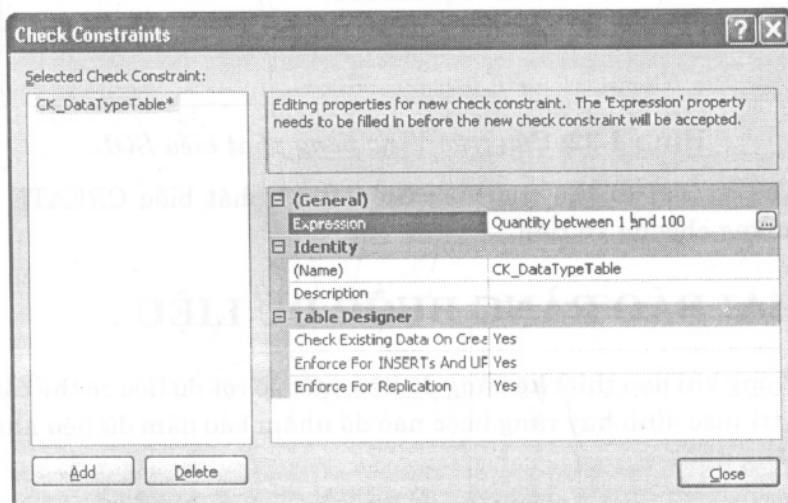
6. KHAI BÁO RÀNG BUỘC DỮ LIỆU

Trong khi bạn thiết kế bảng dữ liệu, một số cột dữ liệu có thể cần khai báo giá trị mặc định hay ràng buộc nào đó nhằm bảo đảm dữ liệu nhập vào được thống nhất.

Ví dụ, bạn có thể khai báo ràng buộc trong khi thiết kế cột dữ liệu bằng cách chọn vào ngăn Constraints (hoặc chọn vào cột dữ liệu rồi R-Click | Check Constraints) rồi R-Click như hình 4-33.

**Hình 4-33: Tao Constraints.**

Chọn New Constraint, cửa sổ xuất hiện và bạn có thể khai báo tương tự như hình 4-34.

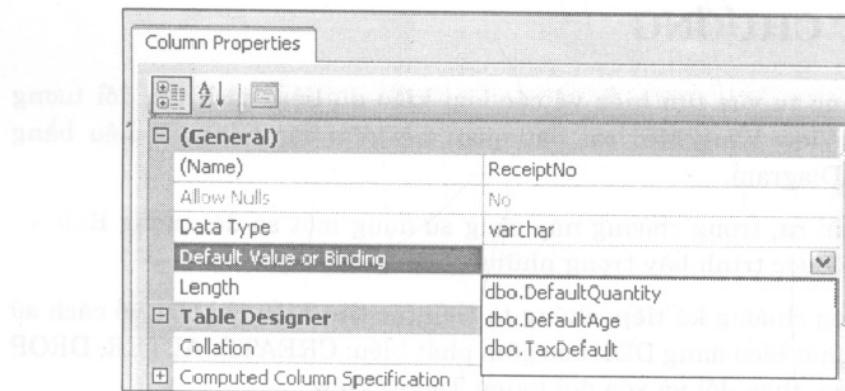
**Hình 4-34: Khai báo Constraint.**

Tuy nhiên, ngoài việc khai báo giá trị mặc định và ràng buộc trong lúc thêm cột dữ liệu vào bảng, bạn có thể định nghĩa giá trị mặc định có thể dùng chung cho nhiều trường hợp.

Chương 4: Thành phần chính trong cơ sở dữ liệu

133

Giả sử, chúng ta đã tạo đối tượng Default với tên TaxDefault có giá trị mặc định là 10%. Khi khai báo cột dữ liệu, bạn có thể chỉ định giá trị mặc định bằng cách chọn vào thuộc tính như hình 4-35.



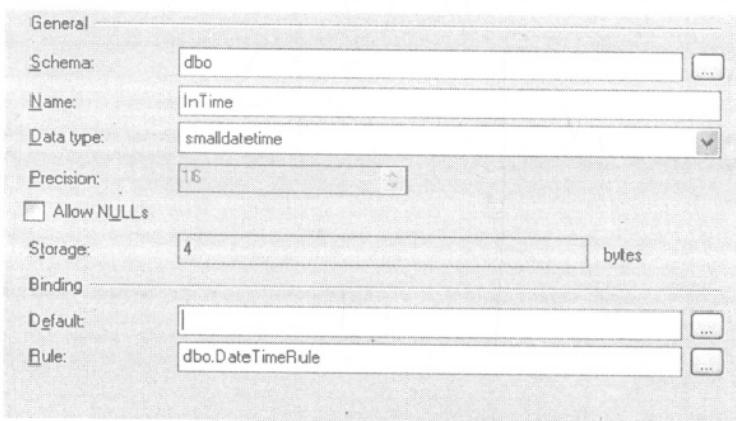
Hình 4-35: Khai báo giá trị mặc định.

Ngoài ra, khi khai báo kiểu dữ liệu do người sử dụng định nghĩa, bạn cũng có thể chọn đối tượng Default này trong phần Default.

Như giới thiệu ở phần User-Defined Data Types, khi khai báo cột dữ liệu, nếu giá trị của chúng được giới hạn trong khoảng cho phép, bạn có thể chỉ định chúng trong khi thiết kế.

Tuy nhiên, để có thể sử dụng chung cho nhiều trường hợp khác, bạn nên tạo đối tượng Rule.

Sau đó, mỗi khi kích hoạt báo kiểu dữ liệu người dùng trong phần User-Defined Data Types thì thông tin trình bày như hình 4-36.



Hình 4-36: Sử dụng Rule.

Lưu ý: Chúng ta sẽ tìm hiểu đối tượng Default và Rule trong những chương kế tiếp.

7. KẾT CHƯƠNG

Chúng ta vừa tìm hiểu về các loại kiểu dữ liệu, cách tạo đối tượng Table và View bằng MS, cài đặt quan hệ giữa các bảng dữ liệu bằng Database Diagram.

Ngoài ra, trong chương này cũng sử dụng một số đối tượng Rule và Default sẽ được trình bày trong những chương kế tiếp.

Trong chương kế tiếp, chúng ta tiếp tục tìm hiểu chi tiết về cách sử dụng các phát biểu dạng DDL bao gồm phát biểu: CREATE, ALTER, DROP dùng để tạo, thay đổi và xóa đối tượng Table, View.

Chương 5:

PHÁT BIỂU T-SQL DẠNG ĐỊNH NGHĨA DỮ LIỆU

Tóm tắt chương 5

Trong chương trước, chúng ta đã tìm hiểu kiểu dữ liệu và các đối tượng trong cơ sở dữ liệu, thiết kế các đối tượng chính của cơ sở dữ liệu là Table, Database Diagram, View bảng MS.

Bạn sẽ tiếp tục tìm hiểu phát biểu T-SQL dạng định nghĩa dữ liệu (DDL) với cú pháp mới của SQL Server 2005 như CREATE, ALTER, DROP dùng cho cơ sở dữ liệu, Table và View rồi thực thi chúng trong cửa sổ Query.

Cũng trong chương này, bạn có thể tìm hiểu cách tạo bảng dữ liệu tạm thời và nơi lưu trữ thông tin đối tượng Database, Table và View.

Các vấn đề chính sẽ được đề cập:

- ✓ Phát biểu CREATE.
- ✓ Phát biểu ALTER.
- ✓ Phát biểu DROP.

1. PHÁT BIỂU CREATE

SQL Server 2005 giới thiệu 45 phát biểu CREATE cho phép bạn thực hiện mọi thao tác liên quan đến cơ sở dữ liệu thay vì sử dụng trên MS. Tuy nhiên, chúng ta sẽ tham khảo từng loại phát biểu CREATE ứng với các đối tượng cơ sở dữ liệu xuyên suốt trong bộ sách SQL Server 2005.

Một số phát biểu SQL dạng CREATE thường được sử dụng khi làm việc với cơ sở dữ liệu sẽ được trình bày như: CREATE DATABASE, CREATE TABLE, CREATE VIEW.

Mặc dù một số phát biểu trong phần này đã trình bày sơ lược trong các chương trước, nhưng trong phần này chúng ta sẽ tìm hiểu chi tiết về cú pháp của chúng, nhằm sử dụng chúng đúng trong ngữ cảnh đang thiết kế.

1.1. Phát biểu CREATE DATABASE

Phát biểu CREATE DATABASE cho phép bạn tạo cơ sở dữ liệu và chỉ định vị trí lưu trữ tập tin bằng phát biểu SQL thay vì sử dụng MS. Để sử dụng phát biểu này, bạn có thể tham khảo cú pháp như sau:

```
CREATE DATABASE database_name
```

```
[ ON
  [ PRIMARY ] [ <filespec> [ ,....n ]
  [ , <filegroup> [ ,....n ] ]
[ LOG ON { <filespec> [ ,....n ] } ]
]
[ COLLATE collation_name ]
[ WITH <external_access_option> ]
]
[ ; ]
```

Trong đó, database_name là tên cơ sở dữ liệu duy nhất trong cơ sở dữ liệu. Từ khóa ON cho phép chỉ định các thuộc tính của cơ sở dữ liệu sẽ tạo.

4.1.1. Tạo mới cơ sở dữ liệu

Bạn tạo cơ sở dữ liệu có tên AccountSystem thì khai báo như ví dụ

Ví dụ 5-1: Khai báo tao cơ sở dữ liệu

```
CREATE DATABASE AccountingSystem
ON
PRIMARY
    (NAME = AccountingSystem_Data,
    FILENAME = 'C:\AccountingSystem.mdf',
    SIZE = 10MB,
    MAXSIZE = 100,
    FILEGROWTH = 5)
LOG ON
    (NAME = AccountingSystem_Log,
    FILENAME = 'C:\AccountingSystem.ldf',
    SIZE = 10MB,
    MAXSIZE = 50,
    FILEGROWTH = 2)
```

Chú ý: Thuộc tính NAME chính là tên cơ sở dữ liệu dùng để tham chiếu, tên cơ sở dữ liệu sẽ xuất hiện trong MS; FILENAME chỉ định đường dẫn và tên tập tin .mdf hay .ldf; SIZE là dung lượng khởi đầu của tập tin .mdf hay .ldf; MAXSIZE là dung lượng tối đa cấp phát cho tập tin .mdf hay .ldf; FILEGROWTH là dung lượng được tự động tăng.

Tuy nhiên, nếu bạn muốn chỉ định vị trí lưu trữ tập tin .mdf và .ldf cùng thư mục với cơ sở dữ liệu master thì khai báo ví dụ trên như sau.

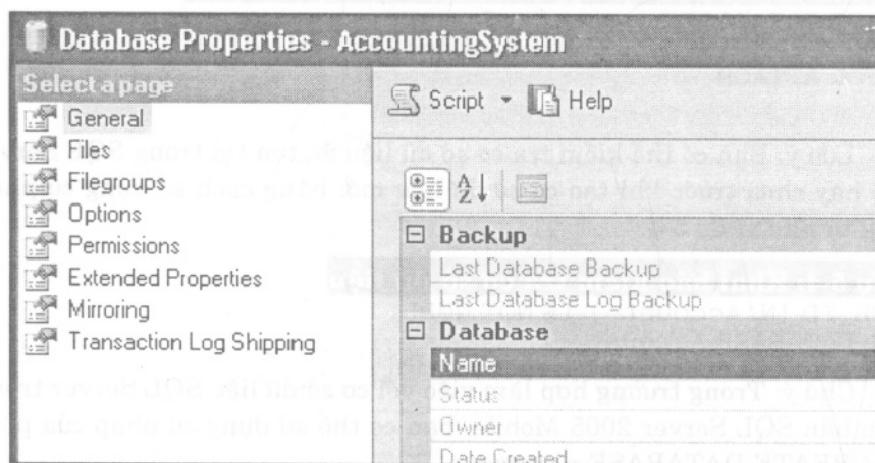
Chương 5: Phát biểu T-SQL định nghĩa dữ liệu

137

Ví dụ 5-2: Khai báo tao cơ sở dữ liệu

--Khai báo lấy ra đường dẫn của cơ sở dữ liệu
 DECLARE @database_path nvarchar(256);
 SET @database_path =
 (SELECT SUBSTRING(physical_name,
 1, CHARINDEX(N'master.mdf',
 LOWER(physical_name)) - 1)
 FROM master.sys.master_files
 WHERE database_id = 1 AND file_id = 1);
--Sử dụng lệnh EXEC
EXEC ('CREATE DATABASE AccountingSystem
ON
PRIMARY
(NAME = AccountingSystem_Data,
FILENAME = ''' + @database_path
+ 'AccountingSystem.mdf'',
SIZE = 10MB,
MAXSIZE = 100,
FILEGROWTH = 5)
LOG ON
(NAME = AccountingSystem_Log,
FILENAME = ''' + @database_path
+ 'AccountingSystem.ldf'',
SIZE = 10MB,
MAXSIZE = 50,
FILEGROWTH = 2)')
--In ra đường dẫn của cơ sở dữ liệu
print @database_path

Chú ý: Bạn có thể kiểm tra các thuộc tính của cơ sở dữ liệu vừa tạo bằng cách sử dụng MS như hình 5-1.



Hình 5-1: Thuộc tính cơ sở dữ liệu.

1.1.2. Tạo cơ sở dữ liệu từ tập tin cơ sở dữ liệu DETACH

Trong trường hợp muốn tạo cơ sở dữ liệu từ cơ sở dữ liệu đã được DETACH, bạn có thể sử dụng cú pháp như sau:

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

Chú ý: Thực hiện phát biểu CREATE với mệnh đề FOR ATTACH như trên chính là ATTACH cơ sở dữ liệu dạng DETACH.

Chẳng hạn, tạo cơ sở dữ liệu DETACH từ cơ sở dữ liệu AccountingSystem, bạn có thể khai báo để ATTACH cơ sở dữ liệu này như ví dụ 5-3.

Ví dụ 5-3: Khai báo tạo cơ sở dữ liệu từ tập tin .mdf

```
CREATE DATABASE AccountingSystem
    ON (FILENAME = 'C:\AccountingSystem.mdf')
    FOR ATTACH
GO
```

Lưu ý: Bạn có thể kiểm tra cơ sở dữ liệu đã tồn tại trong SQL Server 2005 hay chưa trước khi tạo cơ sở dữ liệu mới bằng cách sử dụng cú pháp tương tự như ví dụ 5-4.

Ví dụ 5-4: Khai báo kiểm tra cơ sở dữ liệu

```
IF DB_ID ('Account') IS NOT NULL
DROP DATABASE Account;
```

Chú ý: Trong trường hợp làm việc với cơ sở dữ liệu SQL Server trong phiên bản SQL Server 2005 Mobile, bạn có thể sử dụng cú pháp của phát biểu CREATE DATABASE như sau:

```
CREATE DATABASE database_name
    [DATABASEPASSWORD 'database_password'
```

Chương 5: Phát biểu T-SQL định nghĩa dữ liệu

139



```

[ENCRYPTION {ON|OFF}]
]
[COLLATE collation_name comparison_style]
database password ::= identifier

```

1.1.3. Tạo cơ sở dữ liệu với tùy chọn COLLATION

Bạn có thể tạo cơ sở dữ liệu trong SQL Server 2005 với tùy chọn Collation là Vietnamese_CI_AI bằng cú pháp như ví dụ 5-5.

Ví dụ 5-5: Khai báo sử dụng mệnh đề COLLATION

```

CREATE DATABASE VietnameseDatabase
COLLATE Vietnamese_CI_AI
WITH TRUSTWORTHY ON,
DB_CHAINING ON;
GO

```

Khi thực thi phát biểu CREATE DATABASE trên, bạn có thể tìm thấy kết xuất trình bày như hình 5-2.

The screenshot shows a SQL query window titled 'MYSOLUTION.m...Statement.sql' with the following code:

```

IF DB_ID ('N'VietnameseDatabase') IS NOT NULL
DROP DATABASE VietnameseDatabase;
GO

CREATE DATABASE VietnameseDatabase
COLLATE Vietnamese_CI_AI
WITH TRUSTWORTHY ON,
DB_CHAINING ON;
GO

```

Below the query window is a 'Messages' pane showing the output:

```

Command(s) completed successfully.

```

Hình 5-2: Tạo cơ sở dữ liệu với mệnh đề COLLATION.

Chú ý: Bạn có thể tìm hiểu danh sách COLLATION của SQL Server 2005 bằng cách sử dụng phát biểu như ví dụ 5-6.

Ví dụ 5-6: Khai báo liệt kê danh sách COLLATION

```

SELECT * FROM fn_helpcollations()
WHERE name like 'Vietnamese%'
GO

```

Khi thực thi phát biểu trên, bạn có thể tìm thấy danh sách COLLATION trong SQL Server 2005 như hình 5-3.

	name	description
1	Vietnamese_BIN	Vietnamese, binary sort
2	Vietnamese_BIN2	Vietnamese, binary code point comparison sort
3	Vietnamese_CI_AI	Vietnamese, case-insensitive, accent-insensitive, kanatype-insensitive, width-insensitive
4	Vietnamese_CI_AI_WS	Vietnamese, case-insensitive, accent-insensitive, kanatype-insensitive, width-sensitive
5	Vietnamese_CI_AI_KS	Vietnamese, case-insensitive, accent-insensitive, kanatype-sensitive, width-insensitive
6	Vietnamese_CI_AI_KS_WS	Vietnamese, case-insensitive, accent-insensitive, kanatype-sensitive, width-sensitive
7	Vietnamese_CI_AS	Vietnamese, case-insensitive, accent-sensitive, kanatype-insensitive, width-insensitive
8	Vietnamese_CI_AS_WS	Vietnamese, case-insensitive, accent-sensitive, kanatype-insensitive, width-sensitive
9	Vietnamese_CI_AS_KS	Vietnamese, case-insensitive, accent-sensitive, kanatype-sensitive, width-insensitive
10	Vietnamese_CI_AS_KS_WS	Vietnamese, case-insensitive, accent-sensitive, kanatype-sensitive, width-sensitive
11	Vietnamese_CS_AI	Vietnamese, case-sensitive, accent-insensitive, kanatype-insensitive, width-insensitive
12	Vietnamese_CS_AI_WS	Vietnamese, case-sensitive, accent-insensitive, kanatype-insensitive, width-sensitive
13	Vietnamese_CS_AI_KS	Vietnamese, case-sensitive, accent-insensitive, kanatype-sensitive, width-insensitive
14	Vietnamese_CS_AI_KS_WS	Vietnamese, case-sensitive, accent-insensitive, kanatype-sensitive, width-sensitive
15	Vietnamese_CS_AS	Vietnamese, case-sensitive, accent-sensitive, kanatype-insensitive, width-insensitive
16	Vietnamese_CS_AS_WS	Vietnamese, case-sensitive, accent-sensitive, kanatype-insensitive, width-sensitive
17	Vietnamese_CS_AS_KS	Vietnamese, case-sensitive, accent-sensitive, kanatype-sensitive, width-insensitive
18	Vietnamese_CS_AS_KS_WS	Vietnamese, case-sensitive, accent-sensitive, kanatype-sensitive, width-sensitive

Hình 5-3: Danh sách COLLATION.

1.1.4. Thông tin cơ sở dữ liệu

Với SQL Server 2000, thông tin cơ sở dữ liệu chứa trong bảng sysdatabases thuộc cơ sở dữ liệu master bằng phát biểu:

```
select * from master.dbo.sysdatabases
```

Tuy nhiên, khi làm việc với cơ sở dữ liệu SQL Server 2005, bạn cũng có thể tìm thấy danh sách cơ sở dữ liệu trong bảng sys.databases thay vì sysdatabases.

```
select * from sys.databases
```

Nếu bạn thực thi phát biểu SELECT thứ hai, bạn có thể tìm thấy danh sách cơ sở dữ liệu như hình 5-3-1.

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

141

	name	database_id	source_database_id
1	master	1	NULL
2	tempdb	2	NULL
3	model	3	NULL
4	msdb	4	NULL
5	AdventureWorks	5	NULL
6	AccountSystem	6	NULL
7	HRSQl	7	NULL
8	RecruitVietnam	8	NULL
9	AccountingSystem	9	NULL
10	KHANGDB	10	NULL
11	VietnameseDatabase	11	NULL
12	HumanResourceAndPayroll	12	NULL

Hình 5-3-1: Danh sách cơ sở dữ liệu.

Tuy nhiên, khi bạn thực thi phát biểu SELECT thứ nhất thì kết quả trình bày tương tự như hình 5-3-2.

	name	dbid	sid
1	master	1	0x01
2	tempdb	2	0x01
3	model	3	0x01
4	msdb	4	0x01
5	AdventureWorks	5	0x010500000000000515000000B4
6	AccountSystem	6	0x010500000000000515000000B4
7	HRSQl	7	0x010500000000000515000000B4
8	RecruitVietnam	8	0x010100000000000512000000
9	AccountingSystem	9	0x010500000000000515000000B4
10	KHANGDB	10	0x0984D426483848A462529E82
11	VietnameseDatabase	11	0x010500000000000515000000B4
12	HumanResourceAndPayroll	12	0x010500000000000515000000B4

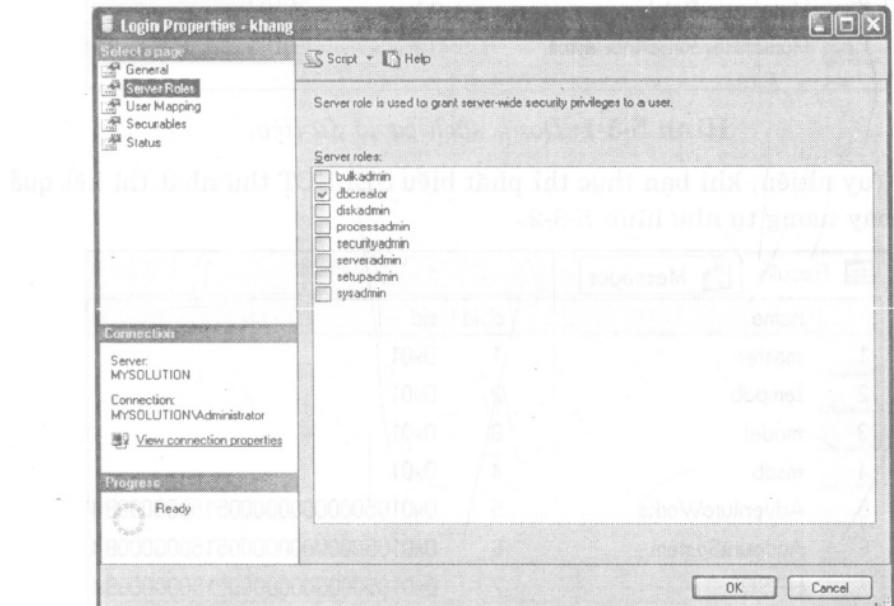
Hình 5-3-2: Danh sách cơ sở dữ liệu.

1.1.5. Quyền cơ sở dữ liệu

Nhằm giới thiệu nhanh quyền hạn của người sử dụng trong cơ sở dữ liệu có thể tạo cơ sở dữ liệu, bạn chỉ cần quan tâm đến quyền tạo cơ sở dữ liệu của người sử dụng đã có sẵn.

Để tạo được cơ sở dữ liệu trong SQL Server 2005, tài khoản đăng nhập phải là sysadmin, dbcreator hoặc có một trong các quyền CREATE DATABASE, CREATE ANY DATABASE, ALTER ANY DATABASE.

Để kiểm tra quyền tạo cơ sở dữ liệu, bạn có thể bắt đầu ngăn Security | Logins | tên tài khoản | R-Click | Properties | Server Roles như hình 5-4.



Hình 5-4: Quyền tạo cơ sở dữ liệu.

Khi đăng nhập vào quyền Administrator hay tài khoản thuộc sysadmin, bạn có thể tạo tài khoản cho người sử dụng bằng cách chọn vào ngăn Security | Logins | R-Click | New login.

Khi khai báo tên tài khoản xong, bạn chọn vào ngăn Server Roles để chọn quyền tạo cơ sở dữ liệu là dbcreator.

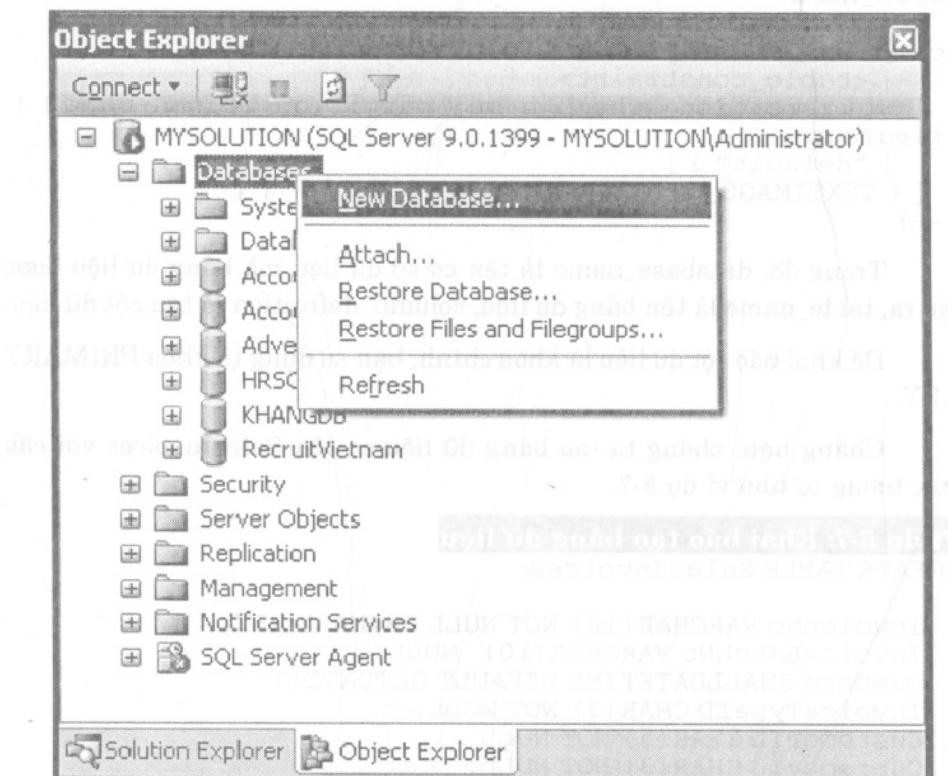
Chú ý: Nếu bạn check vào sysadmin thì tài khoản này có quyền như Administrator của hệ điều hành.

Chương 5: Phát biểu T-SQL dang định nghĩa dữ liệu

143 M®

Sau khi tạo thành công tài khoản trong cơ sở dữ liệu SQL Server 2005, nếu người sử dụng có quyền tạo cơ sở dữ liệu thì khi họ đăng nhập vào SQL Server sẽ có thể sử dụng phát biểu CREATE DATABASE.

Trong trường hợp làm việc với MS, họ có thể sử dụng được menu có tên New Database ... khi R-Click trên ngăn Databases như hình 5-5.



Hình 5-5: Tài khoản được tạo cơ sở dữ liệu.

Chú ý: Để tìm hiểu chi tiết về quyền của người sử dụng trong SQL Server 2005, bạn có thể tìm đọc cuốn sách “Quản trị SQL Server 2005 - Tập 5” thuộc bộ sách “SQL Server 2005” do Nhà sách Minh Khai phát hành trong thời gian tới.

1.2. Phát biểu CREATE TABLE

Mặc dù chúng ta đã làm quen với phát biểu **CREATE TABLE** qua ví dụ trong chương trước, chương này chúng ta tiếp tục tìm hiểu chi tiết về phát biểu **CREATE TABLE**.

1.2.1. Cú pháp phát biểu **CREATE TABLE**

Để tạo bảng dữ liệu trong cơ sở dữ liệu, bạn sử dụng cú pháp như sau:

```
CREATE TABLE
    [ database_name . [ schema_name ] . | schema_name . ]
    table_name
        ( { <column_definition> |
<computed_column_definition> }
        [ <table_constraint> ] [ ,...n ] )
        [ ON { partition_scheme_name ( partition_column_name ) | |
filegroup
        | "default" } ]
        [ { TEXTIMAGE_ON { filegroup | "default" } } ]
    [ ; ]
```

Trong đó, database_name là tên cơ sở dữ liệu mà bảng dữ liệu được tạo ra, table_name là tên bảng dữ liệu, column_definition là tên cột dữ liệu.

Để khai báo cột dữ liệu là khóa chính, bạn sử dụng từ khóa PRIMARY KEY.

Chẳng hạn, chúng ta tạo bảng dữ liệu có tên SalesInvoices với cấu trúc tương tự như ví dụ 5-7.

Ví dụ 5-7: Khai báo tạo bảng dữ liệu

```
CREATE TABLE SalesInvoices
(
    InvoiceNo VARCHAR(10) NOT NULL PRIMARY KEY,
    InvoiceBatchNo VARCHAR(10) NULL,
    DueDate SMALLDATETIME DEFAULT GETDATE(),
    InvoiceTypeID CHAR(3) NOT NULL,
    CustomerID CHAR(5) NOT NULL ,
    CurrencyID CHAR(3) NOT NULL,
    ExchangeRate DECIMAL DEFAULT 0,
    DescriptionInVietnamese NVARCHAR(150)
        NOT NULL,
    DescriptionInSecondLanguage VARCHAR(150)
        NOT NULL,
    ShowDescriptionInVietnamese BIT DEFAULT 1,
    InvoiceDiscontinued BIT DEFAULT 0,
    EntryDate SMALLDATETIME DEFAULT GETDATE(),
    UserName VARCHAR(10) NOT NULL
)
GO
```

Bạn có thể tìm thấy cột khóa chính trong MS như hình 5-6.

Chương 5: Phát biểu T-SQL định nghĩa dữ liệu145 

Column Name	Data Type	Allow Nulls
InvoiceNo	varchar(10)	<input type="checkbox"/>
InvoiceBatchNo	varchar(10)	<input checked="" type="checkbox"/>
DueDate	smalldatetime	<input checked="" type="checkbox"/>
InvoiceTypeID	char(3)	<input type="checkbox"/>
CustomerID	char(5)	<input type="checkbox"/>
CurrencyID	char(3)	<input type="checkbox"/>
ExchangeRate	decimal(18, 0)	<input checked="" type="checkbox"/>
DescriptionInVietnamese	nvarchar(150)	<input type="checkbox"/>
DescriptionInSecondLang	varchar(150)	<input type="checkbox"/>
ShowDescriptionInVietnam	bit	<input checked="" type="checkbox"/>
InvoiceDiscontinued	bit	<input checked="" type="checkbox"/>
EntryDate	smalldatetime	<input checked="" type="checkbox"/>
UserName	varchar(10)	<input type="checkbox"/>

Hình 5-6: Khai báo khóa chính.**1.2.2. Phát biểu CREATE TABLE với từ khóa chính**

Ngoài ra, bạn cũng có thể sử dụng cú pháp để tạo bảng dữ liệu với khóa chính được cấu trúc như ví dụ 5-8.

Ví dụ 5-8: Khai báo tạo bảng dữ liệu

```
CREATE TABLE SalesInvoiceBatchs
(
    InvoiceBatchNo VARCHAR(10) NOT NULL,
    InvoiceBatchDate SMALLDATETIME
        DEFAULT GETDATE(),
    BatchDiscontinued BIT DEFAULT 0,
    EntryDate SMALLDATETIME DEFAULT GETDATE(),
    UserName VARCHAR(10) NOT NULL ,
    PRIMARY KEY CLUSTERED
    (
        InvoiceBatchNo ASC
    )
)
GO
```

Nếu bảng có tổ hợp cột dữ liệu là khóa chính thì bạn có thể khai báo từ khóa PRIMARY KEY như sau:

```
PRIMARY KEY (OrdinalNumber,
    InvoiceNo, ProductID)
```

Chẳng hạn, chúng ta khai báo phát biểu CREATE TABLE để tạo bảng dữ liệu có tên SalesInvoiceDetails như ví dụ 5-9.

Ví dụ 5-9: Khai báo tạo bảng dữ liệu

```
CREATE TABLE SalesInvoiceDetails
(
    OrdinalNumber tinyint NOT NULL,
    InvoiceNo VARCHAR(10) NOT NULL,
    ProductID VARCHAR(10) NOT NULL,
    Quantity DECIMAL DEFAULT 0,
    Price DECIMAL DEFAULT 0,
    Amount DECIMAL DEFAULT 0,
    Discount DECIMAL DEFAULT 0,
    VATTaxRate DECIMAL DEFAULT 0,
    VATTaxAmount DECIMAL DEFAULT 0,
    InvoiceAmount DECIMAL DEFAULT 0,
    PRIMARY KEY (OrdinalNumber,
    InvoiceNo, ProductID)
)
GO
```

Lưu ý: Đối với SQL Server 2000, bạn có thể tìm thấy thông tin của cột dữ liệu trong bảng syscolumns như sau:

```
select * from syscolumns
where id=(select id from sysobjects
where name = 'Customers')
```

Nếu làm việc với cơ sở dữ liệu SQL Server 2005, để biết thông tin của cột dữ liệu, bạn có thể khai báo phát biểu như sau:

```
select * from sys.columns
where object_id=(select object_id
from sys.tables where name = 'Customers')
```

Sau khi tạo thành công, bạn có thể tìm thấy bảng dữ liệu sẽ có tổ hợp khóa trong MS tương tự như hình 5-7.

Column Name	Data Type	Allow Nulls
OrdinalNumber	tinyint	<input type="checkbox"/>
InvoiceNo	varchar(10)	<input type="checkbox"/>
ProductID	varchar(10)	<input type="checkbox"/>
Quantity	decimal(18, 0)	<input checked="" type="checkbox"/>
Price	decimal(18, 0)	<input checked="" type="checkbox"/>
Amount	decimal(18, 0)	<input checked="" type="checkbox"/>
Discount	decimal(18, 0)	<input checked="" type="checkbox"/>
VATTaxRate	decimal(18, 0)	<input checked="" type="checkbox"/>
VATTaxAmount	decimal(18, 0)	<input checked="" type="checkbox"/>
InvoiceAmount	decimal(18, 0)	<input checked="" type="checkbox"/>

Hình 5-7: Tổ hợp khóa.

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

147

1.2.3. Phát biểu CREATE TABLE với hàm IDENTITY

Trong trường hợp bạn muốn khai báo cột số tự động thì sử dụng thuộc tính identity với hai tham số là seed (số bắt đầu) và increment (số tăng).

```
IDENTITY(seed, increment)
```

Chẳng hạn, bạn khai báo phát biểu CREATE TABLE để tạo bảng dữ liệu có tên Categories với cột CategoryID là số tự động tăng kiểu int như ví dụ 5-10.

Ví dụ 5-10: Khai báo cột số tự động tăng

```
CREATE TABLE Categories
(
    CategoryID INT IDENTITY(1,1)
        NOT NULL PRIMARY KEY,
    CategoryName NVARCHAR(50) NOT NULL,
    Discontinued BIT DEFAULT 0
)
GO
```

1.2.4. Phát biểu CREATE TABLE với mệnh đề REFERENCES

Khi khai báo cột dữ liệu khóa ngoại trong bảng dữ liệu quan hệ N, bạn có thể sử dụng mệnh đề REFERENCES để tạo ràng buộc khóa ngoại với khóa chính trong bảng quan hệ 1.

Giả sử, chúng ta có bảng dữ liệu Customers có ba cột khóa ngoại là CustomerTypeID (quan hệ với cột CustomerTypeID khóa chính trong bảng CustomerTypes), ProvinceID (quan hệ với cột ProvinceID khóa chính trong bảng Provinces), BankID (quan hệ với cột BankID khóa chính trong bảng Banks) thì khai báo như ví dụ 5-11.

Ví dụ 5-11: Khai báo mệnh đề References

```
CREATE TABLE Customers
(
    CustomerID CHAR(5) NOT NULL PRIMARY KEY,
    CompanyNameInVietnamese NVARCHAR(50) NOT NULL,
    CompanyNameInSecondLanguage VARCHAR(50) NOT NULL,
    ShowCompanyNameInVietnamese BIT DEFAULT 1,
    ContactName NVARCHAR(50) NOT NULL,
    ContactTitle NVARCHAR(50) NOT NULL,
    Address NVARCHAR(100) NULL,
    ProvinceID CHAR(3) NOT NULL
    REFERENCES Provinces(ProvinceID),
    Telephone VARCHAR(20) NULL,
    FaxNumber VARCHAR(10) NULL,
```

M® 148

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

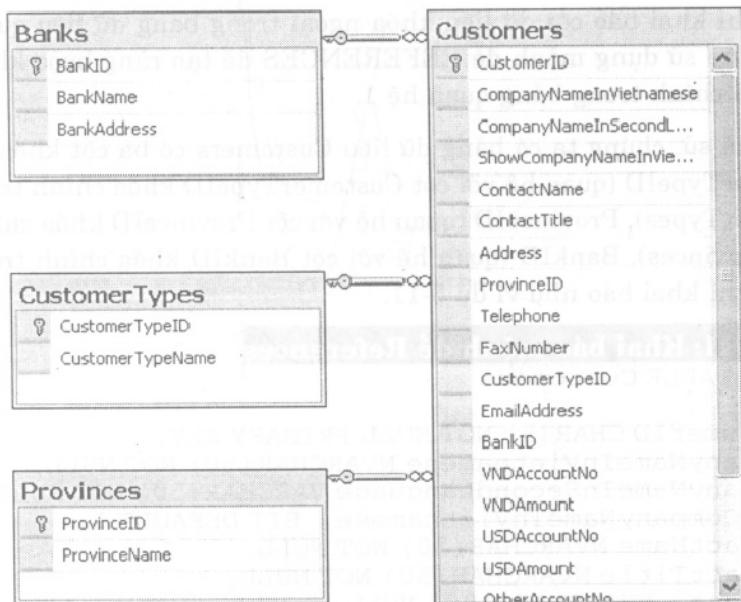
```

CustomerTypeID CHAR (3)
    REFERENCES CustomerTypes (CustomerTypeID),
EmailAddress VARCHAR (20) NULL,
BankID CHAR (3) NOT NULL
    REFERENCES Banks (BankID),
VNDAccountNo VARCHAR (20) NULL,
VNDAmount INT DEFAULT 0,
USDAccountNo VARCHAR (20) NOT NULL,
USDAmount REAL DEFAULT 0,
OtherAccountNo VARCHAR (20) NULL,
OtherAmount REAL DEFAULT 0,
DueDate   SMALLDATETIME DEFAULT GETDATE(),
Discontinued BIT DEFAULT 0
)
GO

```

Sau khi bạn thực thi phát biểu trên, bốn bảng dữ liệu Banks, Provinces, CustomerTypes và Customers sẽ có quan hệ với nhau theo cột dữ liệu đã khai báo.

Để kiểm tra ràng buộc dữ liệu vừa tạo, bạn thêm các bảng dữ liệu Banks, Provinces, CustomerTypes và Customers vào Database Diagram thì quan hệ tự động được thiết lập như hình 5-8.



Hình 5-8: Thiết lập quan hệ bằng mệnh đề REFERENCES.

1.2.5. Phát biểu CREATE TABLE với bảng tạm

Bạn có thể kiểm tra bảng dữ liệu CustomerTypes đã tồn tại trong cơ sở dữ liệu hay chưa bằng cách sử dụng phát biểu như sau:

```
TF EXISTS (SELECT * FROM sys.objects
WHERE object_id = OBJECT_ID(N'[dbo].[CustomerTypes]')
AND type in (N'U'))
```

Khi tạo bảng dữ liệu bằng phát biểu CREATE TABLE, bảng dữ liệu sẽ tồn tại thường trú trong cơ sở dữ liệu.

Tuy nhiên, nếu bạn cần tạo bảng dữ liệu tạm thời có hiệu lực trong phiên làm việc thì sử dụng phát biểu CREATE TABLE với tên bảng dữ liệu có tiền tố # như ví dụ 5-12.

Ví dụ 5-12: Tạo bảng tạm trong một phiên làm việc

```
CREATE TABLE #SalesStatistic
(
    CustomerID CHAR(5) NOT NULL PRIMARY KEY,
    TotalInvoices smallint NOT NULL default 1,
    TotalAmount DECIMAL NOT NULL DEFAULT 0
)
GO
```

Trong trường hợp bảng dữ liệu tạm có hiệu lực trong mọi phiên làm việc, bạn khai báo tiền tố là hai dấu # như ví dụ 5-13.

Ví dụ 5-13: Tạo bảng tạm trong mọi phiên làm việc

```
CREATE TABLE ##SalesStatistic
(
    CustomerID CHAR(5) NOT NULL PRIMARY KEY,
    TotalInvoices smallint NOT NULL default 1,
    TotalAmount DECIMAL NOT NULL DEFAULT 0
)
GO
```

Chú ý: Bảng dữ liệu tạm được tạo ra sẽ lưu trong cơ sở dữ liệu tempdb, bạn có thể tìm hiểu thêm cách làm việc với bảng dữ liệu tạm trong những chương kế tiếp.

1.2.6. Thông tin của bảng dữ liệu

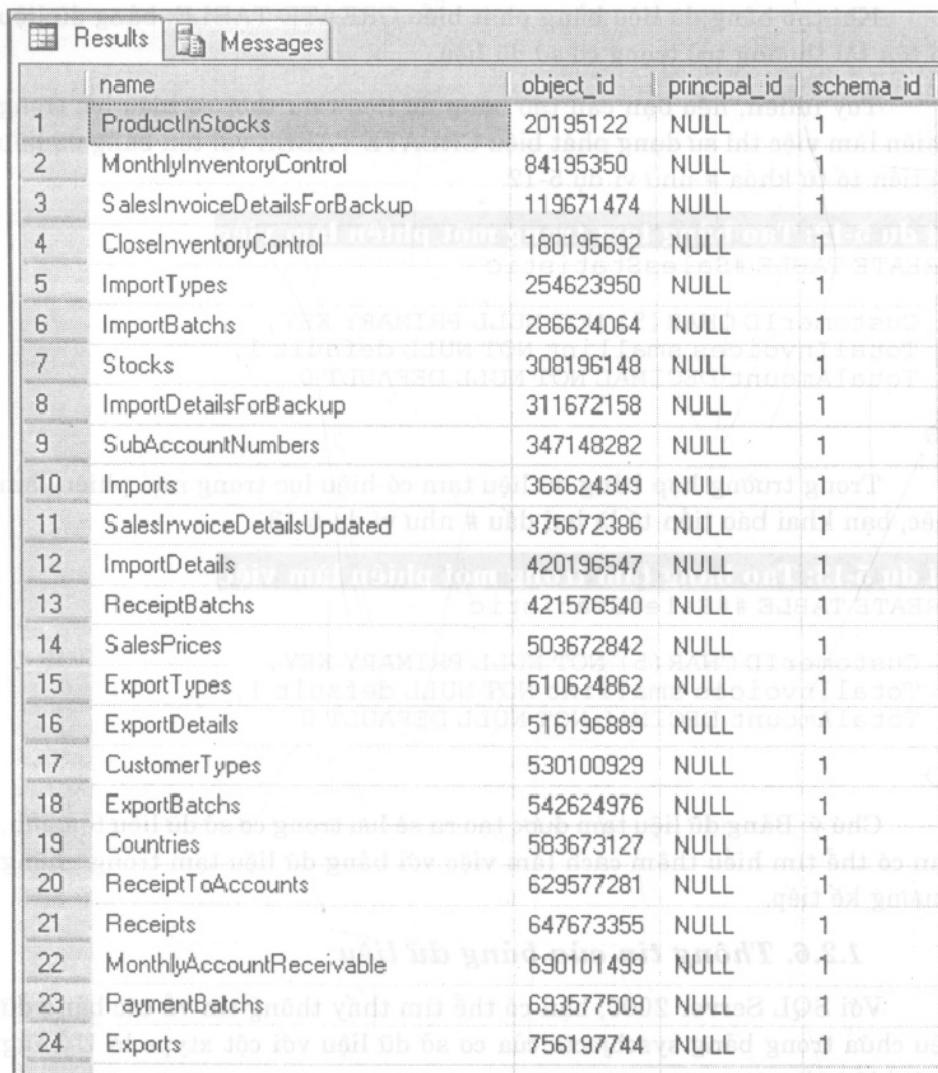
Với SQL Server 2000, bạn có thể tìm thấy thông tin về các bảng dữ liệu chứa trong bảng sysobjects của cơ sở dữ liệu với cột xtype là U bằng phát biểu như sau:

```
select * from sysobjects where xtype='U'
```

Tuy nhiên, khi làm việc với cơ sở dữ liệu SQL Server 2005, bạn có thể tìm thấy danh sách bảng dữ liệu trong bảng sys.tables thay vì sysobjects bảng phát biểu như sau:

```
select * from sys.tables where type='U'
```

Nếu thực thi phát biểu trên, bạn có thể tìm thấy danh sách bảng dữ liệu trong cơ sở dữ liệu AccountSystem tương tự như hình 5-9.



	name	object_id	principal_id	schema_id
1	ProductInStocks	20195122	NULL	1
2	MonthlyInventoryControl	84195350	NULL	1
3	SalesInvoiceDetailsForBackup	119671474	NULL	1
4	CloselInventoryControl	180195692	NULL	1
5	ImportTypes	254623950	NULL	1
6	ImportBatchs	286624064	NULL	1
7	Stocks	308196148	NULL	1
8	ImportDetailsForBackup	311672158	NULL	1
9	SubAccountNumbers	347148282	NULL	1
10	Imports	366624349	NULL	1
11	SalesInvoiceDetailsUpdated	375672386	NULL	1
12	ImportDetails	420196547	NULL	1
13	ReceiptBatchs	421576540	NULL	1
14	SalesPrices	503672842	NULL	1
15	ExportTypes	510624862	NULL	1
16	ExportDetails	516196889	NULL	1
17	CustomerTypes	530100929	NULL	1
18	ExportBatchs	542624976	NULL	1
19	Countries	583673127	NULL	1
20	ReceiptToAccounts	629577281	NULL	1
21	Receipts	647673355	NULL	1
22	MonthlyAccountReceivable	690101499	NULL	1
23	PaymentBatchs	693577509	NULL	1
24	Exports	756197744	NULL	1

Hình 5-9: Danh sách bảng dữ liệu.

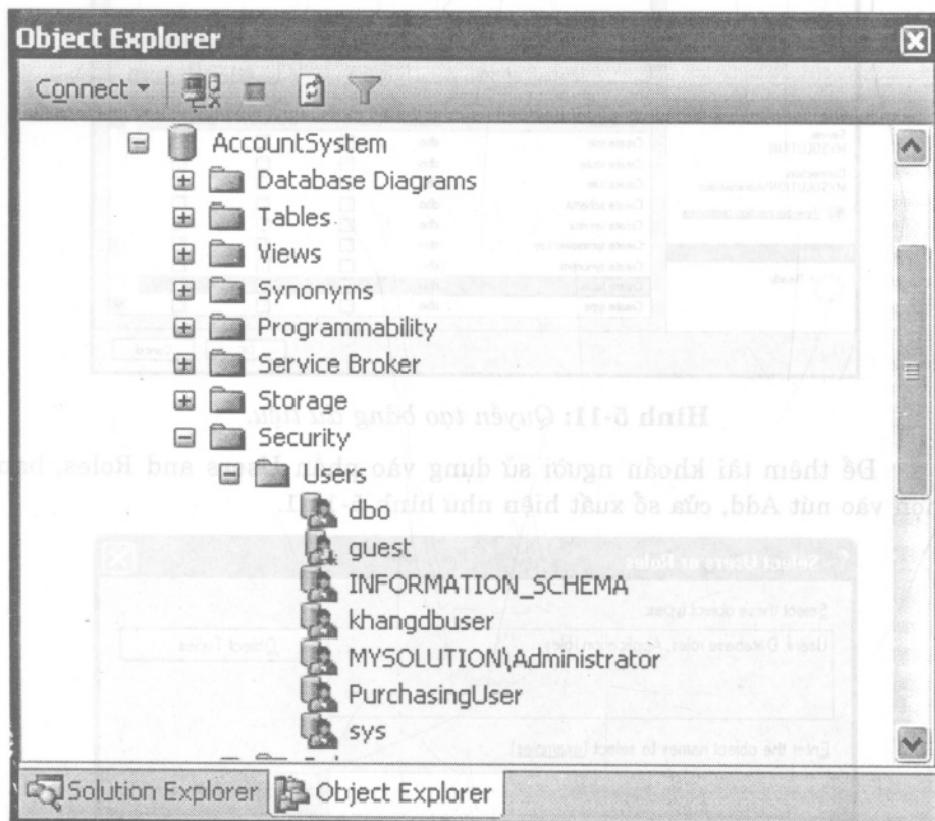
Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

151

**1.2.7. Quyền tạo bảng dữ liệu**

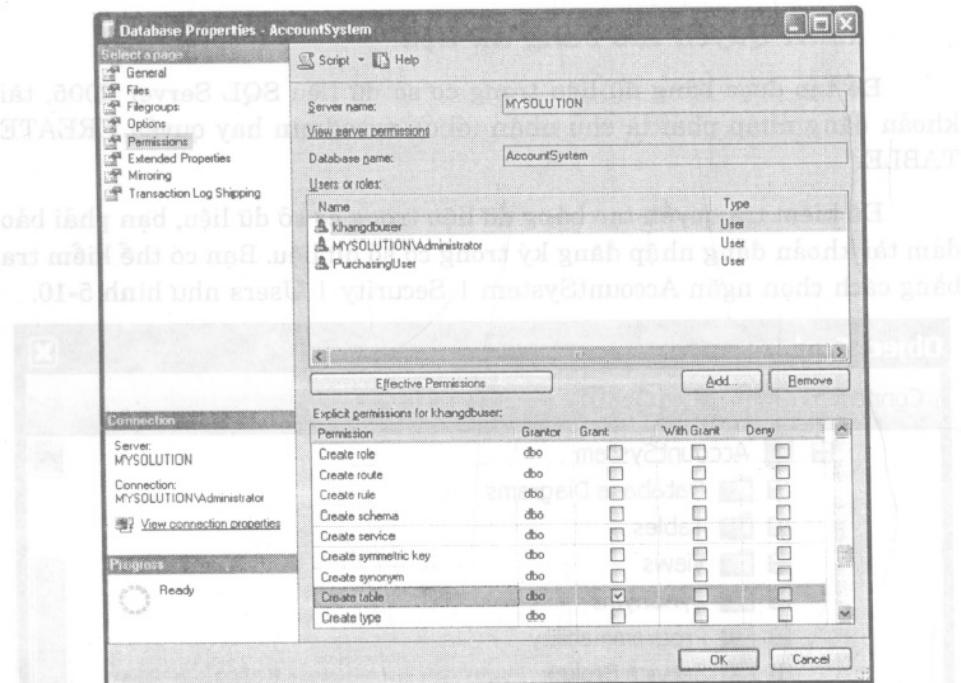
Để tạo được bảng dữ liệu trong cơ sở dữ liệu SQL Server 2005, tài khoản đăng nhập phải là chủ nhân (dbo), sysadmin hay quyền CREATE TABLE.

Để kiểm tra quyền tạo bảng dữ liệu trong cơ sở dữ liệu, bạn phải bảo đảm tài khoản đăng nhập đăng ký trong cơ sở dữ liệu. Bạn có thể kiểm tra bằng cách chọn ngăn AccountSystem | Security | Users như hình 5-10.

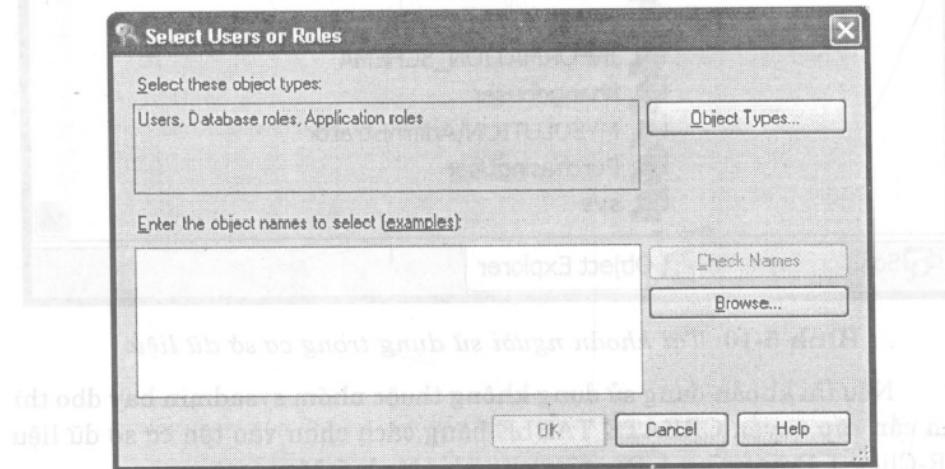


Hình 5-10: Tài khoản người sử dụng trong cơ sở dữ liệu.

Nếu tài khoản đang sử dụng không thuộc nhóm sysadmin hay dbo thì bạn cần cấp quyền CREATE TABLE bằng cách chọn vào tên cơ sở dữ liệu | R-Click | Properties | Permissions như hình 5-11.

Chương 5: Phát biểu T-SQL đang định nghĩa dữ liệu**Hình 5-11: Quyền tạo bảng dữ liệu.**

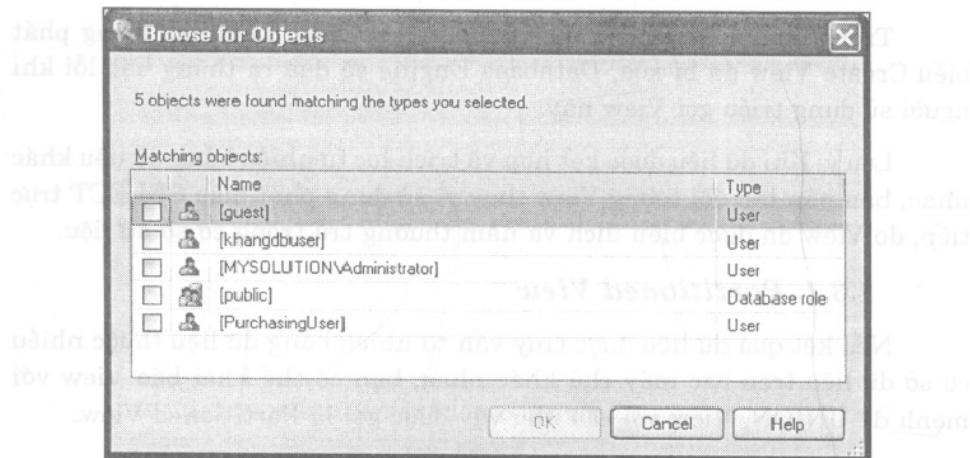
Để thêm tài khoản người sử dụng vào phần Users and Roles, bạn chọn vào nút Add, cửa sổ xuất hiện như hình 5-11-1.

**Hình 5-11-1: Thêm tài khoản người sử dụng.**

Bằng cách chọn vào nút Browse, bạn có thể tìm thấy danh sách các tài khoản trong phần Security | Users như hình 5-11-2.

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

153

**Hình 5-11-2: Danh sách người sử dụng.**

Chú ý: Để tìm hiểu chi tiết về quyền của người sử dụng trên đối tượng cơ sở dữ liệu trong SQL Server 2005, bạn có thể tìm đọc cuốn sách “Quản trị SQL Server 2005” thuộc bộ sách “SQL Server 2005” do Nhà sách Minh Khai phát hành.

1.3. Phát biểu CREATE VIEW

Phát biểu CREATE VIEW cho phép chúng ta tạo VIEW nhằm kết hợp, sắp xếp, trích lọc và tính toán dữ liệu trên một hay nhiều bảng dữ liệu với cấu trúc như sau:

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

Trong đó, view_name là tên View, column là danh sách cột dữ liệu, select_statement là phát biểu SELECT. Chẳng hạn, bạn có thể khai báo View với cấu trúc như ví dụ 5-14.

Ví dụ 5-14: Khai báo đối tượng View

```
CREATE VIEW vwCustomers
AS
SELECT * FROM Customers
WHERE ProvinceID='HCM'
```

Khi bạn triệu gọi View, Database Engine sẽ kiểm tra những bảng dữ liệu hay View đã khai báo trong View có tồn tại hay không. Nếu phát biểu trong View hợp lệ và ràng buộc đúng thì tập dữ liệu trả về.

M® 154**Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu**

Trong trường hợp bảng dữ liệu hay View được khai báo trong phát biểu Create View đã bị xóa, Database Engine sẽ đưa ra thông báo lỗi khi người sử dụng triệu gọi View này.

Lưu ý: Khi dữ liệu được kết hợp và trích lọc từ nhiều bảng dữ liệu khác nhau, bạn nên tạo đối tượng View thay vì sử dụng phát biểu SELECT trực tiếp, do View đã được biên dịch và nằm thường trú trong cơ sở dữ liệu.

1.3.1. Partitioned View

Nếu kết quả dữ liệu được truy vấn từ nhiều bảng dữ liệu thuộc nhiều cơ sở dữ liệu trên các máy chủ khác nhau, bạn có thể khai báo View với mệnh đề UNION, View với cấu trúc như vậy được gọi là Partitioned View.

```
CREATE VIEW Customers
AS
--Bảng dữ liệu Customers trên máy hiện hành.
SELECT *
FROM Company.dbo.Customers
UNION ALL
--Bảng dữ liệu Customers trên máy Server2.
SELECT *
FROM Server2.Company.dbo.Customers
UNION ALL
--Bảng dữ liệu Customers trên máy Server3.
SELECT *
FROM Server3.Company.dbo.Customers
```

Chẳng hạn, bạn khai báo phát biểu SELECT để truy vấn dữ liệu trên bảng Customers của cơ sở dữ liệu AccountSystem và AccountingSystem thì khai báo phát biểu View với cấu trúc như ví dụ 5-15.

Ví dụ 5-15: Khai báo đối tượng View

```
CREATE VIEW vwCustomers
AS
    SELECT CustomerID, CompanyNameInVietnamese,
    ContactName, ContactTitle, Address
    FROM Customers
    WHERE ProvinceID='HCM'
    UNION
    SELECT CustomerID, CompanyNameInVietnamese,
    ContactName, ContactTitle, Address
    FROM AccountingSystem.dbo.Customers
GO
```

Bạn cũng có thể chỉ định tên cột dữ liệu trước từ khóa AS để làm giao tiếp khi dữ liệu trả về nếu triệu gọi View bằng cách khai báo tên cột tương ứng ngay sau tên View.

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

155

Chẳng hạn, bạn khai báo View để truy vấn danh sách nhà cung cấp trong bảng Suppliers như ví dụ 5-16.

Ví dụ 5-16: Khai báo View

```
CREATE VIEW vwSuppliers
AS
    SELECT SupplierID, CompanyNameInVietnamese
    FROM Suppliers
```

Khi thực thi phát biểu SELECT (select * from vwSuppliers) với View có tên vwSuppliers, bạn có thể nhận kết quả như hình 5-12.

SupplierID	CompanyNameInVietnamese
1	Công ty Trách Nhiệm Hữu Hạn Ajinomoto Việt Nam
2	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam
3	Công ty Cổ phần Yamaka Việt Nam
4	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam

Hình 5-12: Thực thi phát biểu SELECT với đối tượng View.

1.3.2. Khai báo tên cột dữ liệu

Nếu bạn muốn kết quả trả về trình bày với tên cột dữ liệu theo ý mình thì bạn có thể khai báo tên cột sau tên View như sau:

```
CREATE VIEW vwSuppliers (columnName [ , ])
AS
    SELECT STATEMENTS
```

Chẳng hạn, bạn khai báo View để truy vấn danh sách nhà cung cấp trong bảng Suppliers với tên gọi hai cột dữ liệu MaNCC, TenNCC ứng với hai cột SupplierID, CompanyNameInVietnamese thì khai báo như ví dụ 5-17.

Ví dụ 5-17: Khai báo tên cột dữ liệu tham chiếu

```
CREATE VIEW vwSuppliers
(MaNCC, TenNCC)
AS
    SELECT SupplierID, CompanyNameInVietnamese
    FROM Suppliers
```

Nếu triệu gọi View có tên vwSuppliers bằng phát biểu SELECT (select * from vwSuppliers) thì kết quả trả về như hình 5-13.

```
select * from vwSuppliers
GO
```

The screenshot shows the SQL Server Management Studio (SSMS) interface. The query window contains the code above. Below it, the results pane is visible, showing a table with four rows of supplier information. The columns are labeled 'ManCC' and 'TenNCC'. The data is as follows:

	ManCC	TenNCC
1	S0001	Công ty Trách Nhiệm Hữu Hạn Ajinomoto Việt Nam
2	S0002	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam
3	S0003	Công ty Cổ phần Yamaka Việt Nam
4	S0004	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam

Hình 5-13: Thay đổi tên cột dữ liệu khi trình bày.

Trong ví dụ trên, thay vì sử dụng từ khóa AS để đổi tên cột dữ liệu khi kết xuất kết quả của View, bạn có thể sử dụng cấu trúc View như ví dụ 5-18.

Ví dụ 5-18: Khai báo từ khóa AS

```
CREATE VIEW vwSuppliers
AS
    SELECT SupplierID As MaNCC,
           CompanyNameInVietnamese As TenNCC
      FROM Suppliers
```

Chú ý: Cấu trúc của View có tối đa 1,024 cột và phụ thuộc vào phát biểu SQL dạng SELECT với mệnh đề JOIN, WHERE, GROUP BY, ORDER BY, ... bạn có thể tham khảo phát biểu SELECT trong chương kế tiếp.

133 Mā hōa View

Nếu bạn có nhu cầu mã hóa View sau khi tạo thì sử dụng phát biểu WITH ENCRYPTION trước từ khóa AS. Khi quyết định mã hóa View, bạn nên lưu trữ cấu trúc của View, nếu có thay đổi cấu trúc thì bạn có thể tạo lại View.

Ví dụ, bạn muốn mã hóa View có tên vwAccounts được mã hóa với cấu trúc như ví dụ 5-19

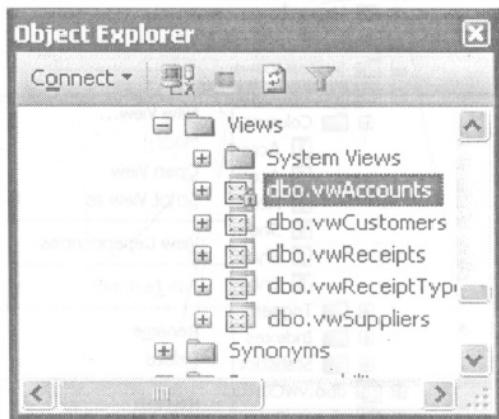
Ví dụ 5-19: Khai báo View có tên vwAccounts

```
CREATE VIEW vwAccounts
    WITH ENCRYPTION
AS
    SELECT * FROM Accounts
GO
```

Chương 5: Phát biểu T-SQL định nghĩa dữ liệu

157

Khi thực thi phát biểu trên, View có tên vwAccounts được tạo ra, bạn có thể tìm thấy biểu tượng chìa khóa trước tên View như hình 5-14.



Hình 5-14: View đã mã hóa.

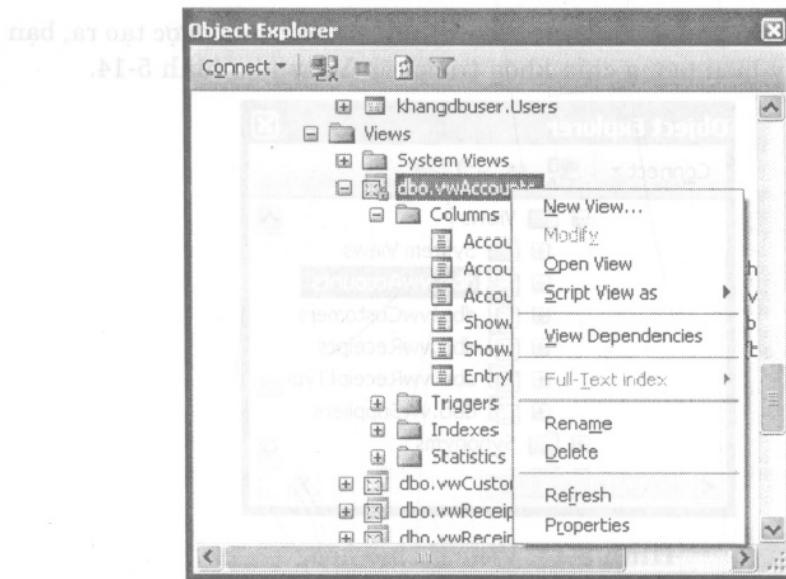
Mặc dù View đã mã hóa, nhưng bạn cũng có thể tìm thấy danh sách cột dữ liệu trong View bằng cách chọn vào ngăn Columns như hình 5-15.



Hình 5-15: Cột dữ liệu trong View đã mã hóa.

Chú ý: Bạn không thể thay đổi View đã khai báo mã hóa bằng cách chọn R-Click | Modify như hình 5-16.

M® 158

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu**Hình 5-16: Không cho phép thay đổi View bằng MS.**

Lưu ý: Bạn cũng không thể sử dụng thủ tục hệ thống sp_helptext để xem nội dung của View như ví dụ 5-20.

Ví dụ 5-20: Khai báo xem nội dung View

```
sp_helptext vwAccounts
GO
```

Khi thực thi phát biểu trên, bạn sẽ nhận được kết quả trả về là "The text for object 'vwAccounts' is encrypted" như hình 5-17.

```
MYSOLUTION.A...tatement.sql* [Summary]
CREATE VIEW vwAccounts
WITH ENCRYPTION
AS
SELECT *
FROM AccountNumbers
GO

sp_helptext vwAccounts
Go

Messages
The text for object 'vwAccounts' is encrypted.
```

Hình 5-17: Mã hóa đối tượng View.

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

159

1.3.4. Nghiêm cấm thay đổi cấu trúc bảng dữ liệu

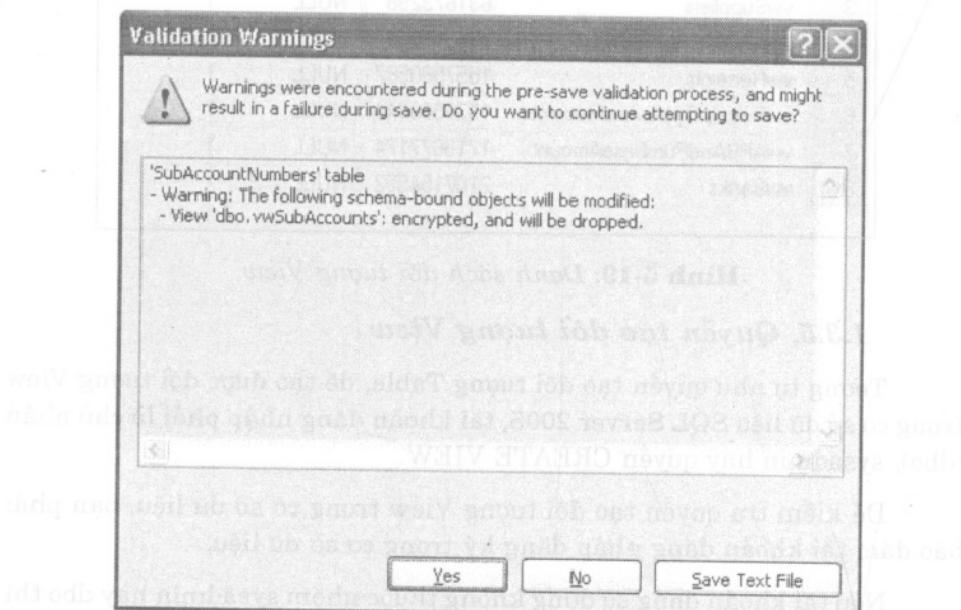
Nếu muốn nghiêm cấm người sử dụng thay đổi tên cột dữ liệu trong bảng dữ liệu được khai báo trong View thì bạn sử dụng phát biểu WITH với từ khóa SCHEMABINDING.

Chẳng hạn, bạn khai báo View có tên vwSubAccounts để liệt kê danh sách cột dữ liệu trong bảng SubAccounts như ví dụ 5-21.

Ví dụ 5-21: Khai báo View có tên vwSubAccounts

```
CREATE VIEW vwSubAccounts
    WITH ENCRYPTION, SCHEMABINDING
AS
    SELECT SubAccountNo, AccountNo,
           ForwardAccountNo,
           SubAccountNameInVietnamese,
           SubAccountNameInSecondLanguage,
           ShowSubAccountNameInVietnamese, EntryDate
    FROM dbo.SubAccountNumbers
GO
```

Sau khi bạn thực thi phát biểu trong ví dụ 5-19, View có tên vwSubAccounts sẽ được tạo ra, nếu bạn sử dụng phát biểu ALTER VIEW hay MS để thay đổi tên cột dữ liệu trong bảng SubAccounts, lỗi sẽ phát sinh tương tự như hình 5-18.



Hình 5-18: Lỗi phát sinh do thay đổi tên cột dữ liệu.

Chú ý: Bạn có thể tạo ra các đối tượng View với cấu trúc đa dạng sau khi tham khảo cách khai báo phát biểu SELECT cùng với các mệnh đề WHERE, TOP, GROUP BY, HAVING, DISTINCT, UNION, UNION ALL, CROSSJOIN, EXCEPT và INTERSECT trong chương kế tiếp.

Ngoài ra, với SQL Server 2000, bảng dữ liệu chứa trong bảng sysobjects trong cơ sở dữ liệu với cột xtype là V bằng phát biểu như sau:

```
select * from sysobjects where xtype= 'V'
```

Tuy nhiên, khi làm việc với cơ sở dữ liệu SQL Server 2005, bạn có thể tìm thấy danh sách bảng dữ liệu trong bảng sys.views thay vì sysobjects bằng phát biểu như sau:

```
select * from sys.views
```

Nếu thực thi phát biểu trên, bạn có thể tìm thấy danh sách đối tượng View trình bày tương tự như hình 5-19.

	name	object_id	principal_id	schema_id
1	vwCustomers	91147370	NULL	1
2	vwAccounts	171147655	NULL	1
3	vwSuppliers	631673298	NULL	1
4	vwARBAAndSalesAmount	1543676547	NULL	1
5	vwReceipts	1557580587	NULL	1
6	vwReceiptTypesAndReceipts	1573580644	NULL	1
7	vwAPBAndPurchaseAmount	1719677174	NULL	1
8	vwBanks	2107154552	NULL	1

Hình 5-19: Danh sách đối tượng View.

1.3.5. Quyền tạo đối tượng View

Tương tự như quyền tạo đối tượng Table, để tạo được đối tượng View trong cơ sở dữ liệu SQL Server 2005, tài khoản đăng nhập phải là chủ nhân (dbo), sysadmin hay quyền CREATE VIEW.

Để kiểm tra quyền tạo đối tượng View trong cơ sở dữ liệu, bạn phải bảo đảm tài khoản đăng nhập đăng ký trong cơ sở dữ liệu.

Nếu tài khoản đang sử dụng không thuộc nhóm sysadmin hay dbo thì bạn cần cấp quyền CREATE VIEW bằng cách chọn vào tên cơ sở dữ liệu | R-Click | Properties | Permissions như hình 5-20.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu 177
Ví dụ 6-4: Khai báo sắp xếp giảm dần theo CustomerID

```
SELECT CustomerID, CompanyNameInVietnamese, Address
FROM Customers
ORDER BY CustomerID DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-4.

CustomerID	CompanyNameInVietnamese	Address
1	A0009 Trung tâm giáo dục Vietnam	1 An Huy
2	A0008 Công ty Cổ phần Reruit Vietnam	1 An Trung
3	A0007 Công ty Đa quốc gia UFCA	1 Hoà An
4	A0006 Tập đoàn UCIA USA	98A Bình Điền, Tp Biên Hòa
5	A0005 Công ty Cổ phần Suzumi Vietnam	101 Hoàng Văn Thụ
6	A0004 Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	1001 Nguyễn Vệ
7	A0003 Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	101 Nguyễn Tuệ
8	A0002 Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	1 Nguyễn Huệ
9	A0001 Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	1 Lê Duẩn

Hình 6-4: Sắp xếp giảm dần.

Trong trường hợp bạn muốn sắp xếp theo tổ hợp cột dữ liệu theo dạng biểu thức thì khai báo tương tự như ví dụ 6-5.

Ví dụ 6-5: Khai báo sắp xếp theo biểu thức

```
SELECT InvoiceNo, ProductID, Quantity,
Price, Discount, Quantity*Price As Amount,
Quantity*Price*VATRate/100 AS VATAmount
FROM SalesInvoiceDetails
ORDER BY Quantity*Price-Discount DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-5.

Chú ý: Nếu bạn không chỉ định từ khóa ASC hay DESC sau cột dữ liệu thì mặc định sắp xếp theo chiều tăng dần.

M®

178

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

The screenshot shows a Microsoft SQL Server Management Studio (SSMS) window. At the top, there are tabs for 'Results' and 'Messages'. The 'Results' tab is selected, displaying a grid of data from a query. The 'Messages' tab shows a single message: 'Query executed successfully.'.

	InvoiceNo	ProductID	Quantity	Price	Discount	Amount	VATAmount
1	SI00000005	P00001	250	15000	50000	3750000	375000.000000
2	SI00000003	P00003	300	10000	50000	3000000	300000.000000
3	SI00000004	P00001	150	15000	0	2250000	225000.000000
4	SI00000006	P00002	165	12500	0	2062500	206250.000000
5	SI00000002	P00002	200	10000	0	2000000	200000.000000
6	SI00000001	P00002	200	10000	50000	2000000	200000.000000
7	SI00000007	P00005	125	13500	60000	1687500	168750.000000
8	SI00000005	P00003	120	12500	55000	1500000	150000.000000
9	SI00000004	P00004	120	12000	0	1440000	144000.000000
10	SI00000003	P00002	100	10000	50000	1000000	100000.000000
11	SI00000001	P00001	100	10000	50000	1000000	100000.000000
12	SI00000002	P00003	100	10000	100000	1000000	100000.000000
13	SI00000008	P00006	35	14500	30000	507500	50750.000000
14	SI00000009	P00003	35	12500	15000	437500	43750.000000
15	SI00000010	P00003	35	12500	30000	437500	43750.000000
16	SI00000008	P00003	35	12500	30000	437500	43750.000000
17	SI00000010	P00004	35	12500	30000	437500	43750.000000
18	SI00000010	P00002	35	12500	30000	437500	43750.000000
19	SI00000007	P00006	25	14500	10000	362500	36250.000000
20	SI00000011	P00006	25	12500	0	312500	31250.000000
21	SI00000011	P00002	25	10500	0	262500	26250.000000
22	SI00000012	P00002	25	10500	30000	262500	26250.000000
23	SI00000009	P00001	15	12500	0	187500	18750.000000
24	SI00000010	P00004	5	12000	0	60000	6000.000000
25	SI00000009	P00004	5	12500	15000	62500	6250.000000
26	SI00000012	P00004	5	12500	30000	62500	6250.000000
27	SI00000013	P00005	5	12500	30000	62500	6250.000000
28	SI00000017	P00006	5	12500	30000	62500	6250.000000

Query executed successfully.

Hình 6-5: Sắp xếp dữ liệu theo biểu thức.

3. TỪ KHÓA TOP

Trong bảng dữ liệu thường có nhiều mẫu tin, nếu bạn có nhu cầu lấy ra một số mẫu tin nào đó thì dùng từ khóa TOP trong mệnh đề SELECT như sau:

```
SELECT TOP [NUMBER | PERCENT] [WITH TIES]
{ * | Col1, Col2, Col3 }
FROM TABLENAME
```

Chương 6: Phát biểu T-SQL cơ bản dang truy vấn dữ liệu

179 MITSUBISHI

3.1. Từ khóa TOP với số cu thể

Để liệt kê danh sách gồm 5 khách hàng trong bảng Customers thì khai báo như ví dụ 6-6

Ví dụ 6-6: Khai báo từ khóa TOP

```
SELECT TOP 5 CompanyNameInVietnamese, Address  
FROM Customers  
GO
```

Thay vì trình bày danh sách 7 mẫu tin như hình 6-3, trong trường hợp này chỉ có 5 khách hàng trình bày như hình 6-6.

	CompanyNameInVietnamese	Address
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	1 Lê Duẩn
2	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	1 Nguyễn Huệ
3	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	101 Nguyễn Tuệ
4	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	1001 Nguyễn Văn
5	Công ty Cổ phần Suzumi Vietnam	101 Hoàng Văn Thu

Hình 6-6: Lấy ra 5 mẫu tin.

3.2. Từ khóa TOP với số phần trăm

Ngoài ra, bạn có thể chỉ định phần trăm mẫu tin cần trình bày thay vì con số cụ thể như trên bằng cách sử dụng từ khóa PERCENT như sau:

```
SELECT TOP PERCENT { * | Col1, Col2, Col3 }  
FROM TABLENAME
```

Giả sử, trong bảng Customers đang có 9 mẫu tin ứng với 7 khách hàng, bạn có thể trình bày 80% danh sách khách hàng bằng cách khai báo như ví dụ 6-7.

Ví dụ 6-7: Khai báo từ khóa PERCENT

```
SELECT TOP 80 PERCENT CompanyNameInVietnamese, Address  
FROM Customers  
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-7.

Lưu ý: Bạn có thể khai báo số thập phân trước từ khóa PERCENT để lấy ra phần trăm mẩu tin. Chẳng hạn, bạn khai báo phát biểu trong ví dụ trên như ví dụ 6-7-1

M® 180

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

	Company Name Vietnamese	Address
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	1 Lê Duẩn
2	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	1 Nguyễn Huệ
3	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	101 Nguyễn Tuệ
4	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	1001 Nguyễn Vệ
5	Công ty Cổ phần Suzumi Vietnam	101 Hoàng Văn Thụ
6	Tập đoàn UCL USA	98A Bình Điền, Tp Biên Hòa
7	Công ty Đa quốc gia UFCA	1 Hòa An
8	Công ty Cổ phần Reruit Vietnam	1 An Trung

Hình 6-7: Từ khóa PERCENT.**Ví dụ 6-7-1: Khai báo từ khóa PERCENT với số lẻ**

```
SELECT TOP 25.5 PERCENT
InvoiceNo, ProductID, Quantity, Price, Discount,
Quantity*Price As Amount,
Quantity*Price*VATRate/100 AS VATAmount
FROM SalesInvoiceDetails
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-7-1.

	InvoiceNo	ProductID	Quantity	Price	Discount	Amount	VATAmount
1	S100000001	P00001	100	10000	50000	1000000	100000.000000
2	S100000002	P00003	100	10000	100000	1000000	100000.000000
3	S100000003	P00002	100	10000	50000	1000000	100000.000000
4	S100000004	P00001	150	15000	0	2250000	225000.000000
5	S100000005	P00001	250	15000	50000	3750000	375000.000000
6	S100000006	P00002	165	12500	0	2062500	206250.000000
7	S100000007	P00005	125	13500	60000	1687500	168750.000000
8	S100000008	P00006	35	14500	30000	507500	50750.000000

Hình 6-7-1: Từ khóa PERCENT với số thập phân.

Thông thường chúng ta sử dụng từ khóa TOP để lấy ra số lượng ứng với yêu cầu nào đó như: N sản phẩm bán chạy nhất, N khách hàng có nợ phải thu lớn nhất, N sản phẩm bán chạy nhất ở tỉnh thành A.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

181

Như vậy, nếu số lượng mẫu tin thỏa mãn yêu cầu của bạn là M trong khi đó phát biểu SQL dạng SELECT với từ khóa TOP chỉ lấy ra là N và N lại nhỏ hơn M.

Điều này có nghĩa là số lượng thực tế lớn hơn số mà bạn sử dụng phát biểu SELECT với từ khóa TOP trả về, vấn đề ở chỗ là chúng ta có muốn lấy ra số mẫu tin đúng theo yêu cầu hay không.

Giả sử, trong bảng SalesInvoiceDetails chúng ta có 7 sản phẩm ứng với 13 hóa đơn bán hàng như hình 6-8

	InvoiceNo	ProductID	Quantity	Price	Discount	Amount	InvoiceAmount
1	S100000005	P00001	250	15000	50000	3750000	4075000.000000
2	S100000003	P00003	300	10000	50000	3000000	3250000.000000
3	S100000004	P00001	150	15000	0	2250000	2475000.000000
4	S100000006	P00002	165	12500	0	2062500	2268750.000000
5	S100000001	P00002	200	10000	50000	2000000	2150000.000000
6	S100000002	P00002	200	10000	0	2000000	2200000.000000
7	S100000007	P00005	125	13500	60000	1687500	1796250.000000
8	S100000005	P00003	120	12500	55000	1500000	1595000.000000
9	S100000004	P00004	120	12000	0	1440000	1584000.000000
10	S100000001	P00001	100	10000	50000	1000000	1050000.000000
11	S100000002	P00003	100	10000	100000	1000000	1000000.000000
12	S100000003	P00002	100	10000	50000	1000000	1050000.000000
13	S100000008	P00006	35	14500	30000	507500	528250.000000
14	S100000010	P00002	35	12500	30000	437500	451250.000000
15	S100000008	P00003	35	12500	30000	437500	451250.000000
16	S100000009	P00003	35	12500	15000	437500	466250.000000
17	S100000010	P00003	35	12500	30000	437500	451250.000000
18	S100000010	P00004	35	12500	30000	437500	451250.000000
19	S100000007	P00006	25	14500	10000	362500	388750.000000
20	S100000011	P00006	25	12500	0	312500	343750.000000
21	S100000011	P00002	25	10500	0	262500	288750.000000
22	S100000012	P00002	25	10500	30000	262500	258750.000000
23	S100000009	P00001	15	12500	0	187500	206250.000000
24	S100000013	P00004	5	12500	30000	62500	38750.000000
25	S100000012	P00004	5	12500	30000	62500	38750.000000
26	S100000013	P00005	5	12500	30000	62500	38750.000000
27	S100000009	P00004	5	12500	15000	62500	53750.000000
28	S100000012	P00006	5	12500	30000	62500	30750.000000

Hình 6-8: Danh sách sản phẩm bán.

Chú ý: Bạn có thể kiểm tra bằng cách thực thi phát biểu SELECT như ví dụ 6-8.

Ví dụ 6-8: Khai báo liệt kê danh sách sản phẩm bán

```
SELECT InvoiceNo, ProductID, Quantity,
Price, Discount, Quantity*Price AS Amount,
Quantity*Price*(1+VATRate/100)-Discount AS InvoiceAmount
FROM SalesInvoiceDetails
ORDER BY Quantity*Price DESC
GO
```

Nếu bạn sử dụng phát biểu SELECT với từ khóa TOP và chỉ định lấy ra 10 sản phẩm có số tiền bán là lớn nhất thì khai báo như ví dụ 6-9.

Ví dụ 6-9: Khai báo lấy ra 10 sản phẩm có số tiền lớn nhất

```
SELECT TOP 10 InvoiceNo, ProductID,
Quantity, Price, Discount,
Quantity*Price AS Amount,
Quantity*Price*(1+VATRate/100)-Discount AS InvoiceAmount
FROM SalesInvoiceDetails
ORDER BY Quantity*Price*(1+VATRate/100)-Discount DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày 10 sản phẩm bán như hình 6-9.

	InvoiceNo	ProductID	Quantity	Price	Discount	Amount	InvoiceAmount
1	SI00000005	P00001	250	15000	50000	3750000	4075000.000000
2	SI00000003	P00003	300	10000	50000	3000000	3250000.000000
3	SI00000004	P00001	150	15000	0	2250000	2475000.000000
4	SI00000006	P00002	165	12500	0	2062500	2268750.000000
5	SI00000002	P00002	200	10000	0	2000000	2200000.000000
6	SI00000001	P00002	200	10000	50000	2000000	2150000.000000
7	SI00000007	P00005	125	13500	60000	1687500	1796250.000000
8	SI00000005	P00003	120	12500	55000	1500000	1595000.000000
9	SI00000004	P00004	120	12000	0	1440000	1584000.000000
10	SI00000001	P00001	100	10000	50000	1000000	1050000.000000

Hình 6-9: 10 sản phẩm có số tiền bán lớn nhất.

Tuy nhiên, theo số liệu trình bày trong hình 6-8 thì bảng SalesInvoiceDetails đang có 1 sản phẩm P00001 ứng với số hóa đơn SI00000001 có số tiền là 1050000, nếu xét về mặt thống kê thì chúng ta đã bỏ sót sản phẩm P00002 của số hóa đơn là SI00000003 có số tiền bán cũng bằng 1050000.

3.3. Từ khóa TOP với WITH TIES

Như vậy, nếu có nhu cầu lấy thêm tất cả mẫu tin thỏa mãn yêu cầu thì bạn sử dụng từ khóa WITH TIES như ví dụ 6-10.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

183

Ví dụ 6-10: Khai báo từ khóa WITH TIES

```
SELECT TOP 10 WITH TIES
InvoiceNo, ProductID,
Quantity, Price, Discount,
Quantity*Price AS Amount,
Quantity*Price*(1+VATRate/100)-Discount AS InvoiceAmount
FROM SalesInvoiceDetails
ORDER BY Quantity*Price*(1+VATRate/100)-Discount DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả trình bày sẽ là 11 mẫu tin thay vì 10 mẫu tin như hình 6-10.

	InvoiceNo	ProductID	Quantity	Price	Discount	Amount	InvoiceAmount
1	SI00000005	P00001	250	15000	50000	3750000	4075000.000000
2	SI00000003	P00003	300	10000	50000	3000000	3250000.000000
3	SI00000004	P00001	150	15000	0	2250000	2475000.000000
4	SI00000006	P00002	165	12500	0	2062500	2268750.000000
5	SI00000002	P00002	200	10000	0	2000000	2200000.000000
6	SI00000001	P00002	200	10000	50000	2000000	2150000.000000
7	SI00000007	P00005	125	13500	60000	1687500	1796250.000000
8	SI00000005	P00003	120	12500	55000	1500000	1595000.000000
9	SI00000004	P00004	120	12000	0	1440000	1584000.000000
10	SI00000003	P00002	100	10000	50000	1000000	1050000.000000
11	SI00000001	P00001	100	10000	50000	1000000	1050000.000000

Hình 6-10: Sử dụng từ khóa WITH TIES.

Chú ý: Chúng ta chỉ khai báo TOP 10 nhưng số lượng mẫu tin trả về vẫn là 11.

4. TỪ KHÓA DISTINCT

Từ khóa DISTINCT cho phép bạn loại bỏ những mẫu tin trùng lặp (được tính trên những cột dữ liệu khai báo trong phát biểu SELECT) trước khi lấy ra.

Chẳng hạn, nếu bạn khai báo phát biểu SELECT để lấy ra ba cột dữ liệu ProductID, Quantity, Price trong bảng SalesInvoiceDetails như ví dụ 6-11.

Ví dụ 6-11: Khai báo liệt kê danh sách sản phẩm

```
SELECT ProductID, Quantity, Price
FROM SalesInvoiceDetails
ORDER BY ProductID ASC
GO
```

M® 184**Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu**

Khi thực thi phát biểu trong ví dụ trên, kết quả trình bày sẽ là 30 mẫu tin như hình 6-11.

	ProductID	Quantity	Price
1	P00001	150	15000
2	P00001	250	15000
3	P00001	15	12500
4	P00001	100	10000
5	P00002	200	10000
6	P00002	200	10000
7	P00002	35	12500
8	P00002	25	10500
9	P00002	25	10500
10	P00002	165	12500
11	P00002	100	10000
12	P00003	35	12500
13	P00003	35	12500
14	P00003	35	12500
15	P00003	300	10000
16	P00003	100	10000
17	P00003	120	12500
18	P00004	120	12000
19	P00004	5	12500
20	P00004	5	12500
21	P00004	5	12500
22	P00004	35	12500
23	P00004	5	12000
24	P00004	1	11500
25	P00005	5	12500
26	P00005	125	13500
27	P00006	35	14500
28	P00006	25	14500

Query executed successfully.

Hình 6-11: Danh sách gồm 30 sản phẩm.

Trong hình trên, chúng ta tìm thấy có các nhóm mẫu tin số (5, 6), (8, 9), (12, 13, 14), (19, 20, 21) có dữ liệu trên ba cột ProductID, Quantity, Price đều giống nhau. Nếu chỉ lấy một mẫu tin trong những mẫu tin trùng lặp này, bạn sử dụng từ khóa DISTINCT như ví dụ 6-12.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

185

Ví dụ 6-12: Khai báo sử dụng từ khóa DISTINCT

```
SELECT DISTINCT ProductID, Quantity, Price
FROM SalesInvoiceDetails
ORDER BY ProductID ASC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày sẽ là 24 mẫu tin như hình 6-12.

	ProductID	Quantity	Price
1	P00001	15	12500
2	P00001	100	10000
3	P00001	150	15000
4	P00001	250	15000
5	P00002	25	10500
6	P00002	35	12500
7	P00002	100	10000
8	P00002	165	12500
9	P00002	200	10000
10	P00003	35	12500
11	P00003	100	10000
12	P00003	120	12500
13	P00003	300	10000
14	P00004	1	11500
15	P00004	5	12000
16	P00004	5	12500
17	P00004	35	12500
18	P00004	120	12000
19	P00005	5	12500
20	P00005	125	13500
21	P00006	5	12500
22	P00006	25	12500
23	P00006	25	14500
24	P00006	35	14500

Hình 6-12: Sử dụng từ khóa DISTINCT.

M[®] 186**Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu**

5. PHÁT BIỂU SELECT VÀ MỆNH ĐỀ INTO

Mệnh đề INTO cho phép bạn thêm danh sách mẫu tin lấy ra từ phát biểu SELECT vào bảng dữ liệu tự tạo ra với tên khai báo sau mệnh đề INTO.

5.1. Mệnh đề INTO và bảng dữ liệu tạm

Nếu bạn muốn tạo bảng dữ liệu tạm, bạn có thể khai báo như ví dụ 6-13.

Ví dụ 6-13: Khai báo mệnh đề INTO

```
SELECT DISTINCT ProductID, Quantity, Price, VATRate, Discount
INTO #InvoiceDetails
FROM SalesInvoiceDetails
ORDER BY ProductID ASC
GO
SELECT * FROM #InvoiceDetails
GO
DROP TABLE #InvoiceDetails
GO
```

Trong đó, bảng dữ liệu tạm có tên InvoiceDetails được tạo ra sau khi kết thúc phát biểu SELECT và sẽ được truy vấn trong phát biểu SELECT thứ hai, rồi bị xóa trong phát biểu DROP TABLE.

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-13.

5.2. Mệnh đề INTO và bảng trong cơ sở dữ liệu

Trong trường hợp tạo bảng dữ liệu tồn tại trong cơ sở dữ liệu thì sử dụng mệnh đề INTO như ví dụ 6-14.

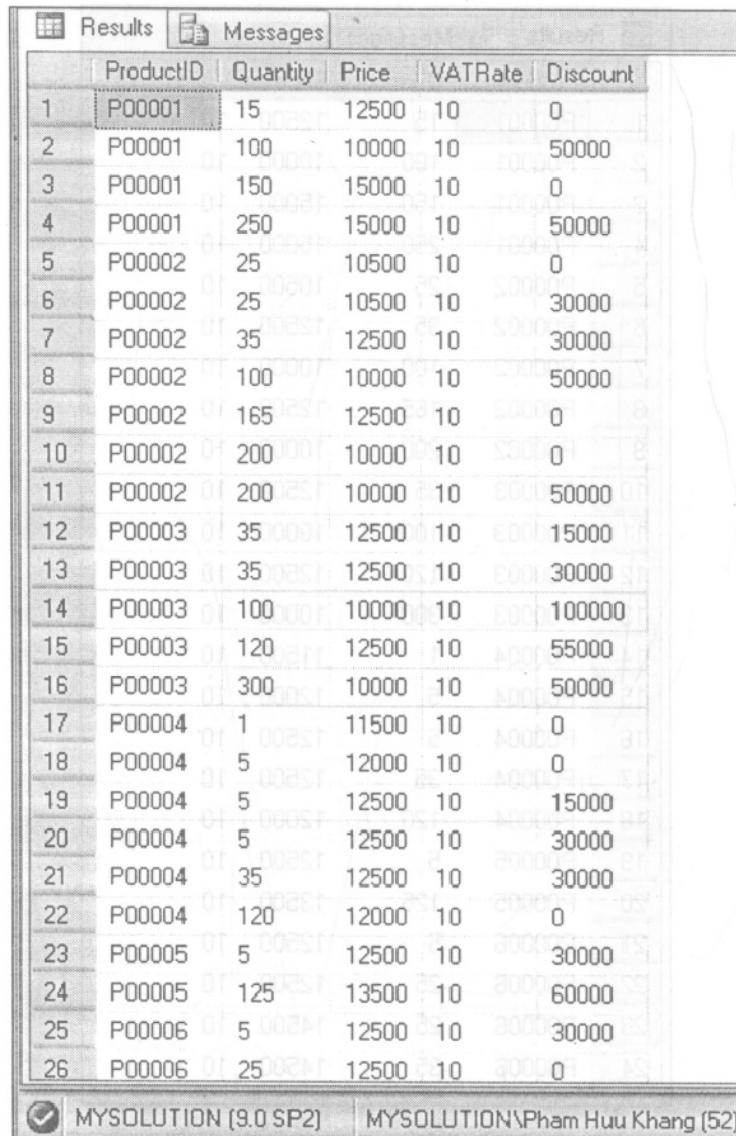
Ví dụ 6-14: Khai báo mệnh đề INTO

```
SELECT DISTINCT ProductID, Quantity, Price, VATRate
INTO InvoiceDetails
FROM SalesInvoiceDetails
ORDER BY ProductID ASC
GO
SELECT * FROM InvoiceDetails
GO
```

-- Xóa bảng tạm trước khi thực thi lại hai phát biểu trên
DROP TABLE InvoiceDetails
GO

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

187



The screenshot shows the SQL Server Management Studio interface with the 'Results' tab selected. The results grid displays data from a table with columns: ProductID, Quantity, Price, VATRate, and Discount. The data consists of 26 rows, each containing a unique ProductID (from 1 to 26), a quantity value, a price value, a VATRate of 10, and a discount value. The data is as follows:

	ProductID	Quantity	Price	VATRate	Discount
1	P00001	15	12500	10	0
2	P00001	100	10000	10	50000
3	P00001	150	15000	10	0
4	P00001	250	15000	10	50000
5	P00002	25	10500	10	0
6	P00002	25	10500	10	30000
7	P00002	35	12500	10	30000
8	P00002	100	10000	10	50000
9	P00002	165	12500	10	0
10	P00002	200	10000	10	0
11	P00002	200	10000	10	50000
12	P00003	35	12500	10	15000
13	P00003	35	12500	10	30000
14	P00003	100	10000	10	100000
15	P00003	120	12500	10	55000
16	P00003	300	10000	10	50000
17	P00004	1	11500	10	0
18	P00004	5	12000	10	0
19	P00004	5	12500	10	15000
20	P00004	5	12500	10	30000
21	P00004	35	12500	10	30000
22	P00004	120	12000	10	0
23	P00005	5	12500	10	30000
24	P00005	125	13500	10	60000
25	P00006	5	12500	10	30000
26	P00006	25	12500	10	0

MYSOLUTION [9.0 SP2] MYSOLUTION\Pham Huu Khang (52)

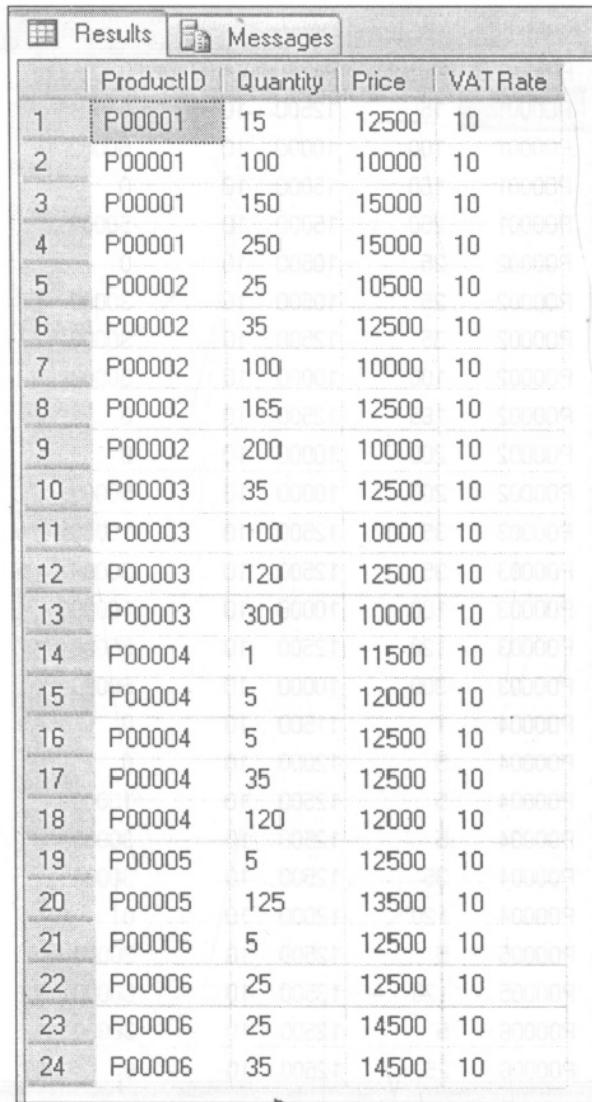
Hình 6-13: Sử dụng mệnh đề INTO.

Trong đó, bảng dữ liệu có tên InvoiceDetails được tạo ra trong cơ sở dữ liệu sau khi kết thúc phát biểu SELECT và sẽ được truy vấn trong phát biểu SELECT thứ hai, rồi bị xóa trong phát biểu DROP TABLE.

Khi thực thi phát biểu trong ví dụ trên, kết quả trình bày sẽ là mẫu tin trong bảng InvoiceDetails như hình 6-14.

M® 188

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu



	ProductID	Quantity	Price	VAT Rate
1	P00001	15	12500	10
2	P00001	100	10000	10
3	P00001	150	15000	10
4	P00001	250	15000	10
5	P00002	25	10500	10
6	P00002	35	12500	10
7	P00002	100	10000	10
8	P00002	165	12500	10
9	P00002	200	10000	10
10	P00003	35	12500	10
11	P00003	100	10000	10
12	P00003	120	12500	10
13	P00003	300	10000	10
14	P00004	1	11500	10
15	P00004	5	12000	10
16	P00004	5	12500	10
17	P00004	35	12500	10
18	P00004	120	12000	10
19	P00005	5	12500	10
20	P00005	125	13500	10
21	P00006	5	12500	10
22	P00006	25	12500	10
23	P00006	25	14500	10
24	P00006	35	14500	10

Hình 6-14: Sử dụng mệnh đề INTO.

6. PHÁT BIỂU SELECT VÀ TỪ KHÓA AS

Từ khóa AS cho phép bạn ánh xạ tên cột dữ liệu, giá trị cụ thể hay biểu thức thành cột dữ liệu khi kết xuất kết quả.

Giả sử, khi bạn tạo biểu thức tính toán trong phát biểu SELECT và không sử dụng từ khóa AS để đặt tên cho biểu thức này như ví dụ 6-15.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

189

Ví dụ 6-15: Khai báo biến thức Quantity*Price

```
SELECT ProductID, Quantity, Price,
Discount, Quantity*Price -Discount
FROM SalesInvoiceDetails
WHERE Quantity >100
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-15.

	ProductID	Quantity	Price	Discount	VATRate	(No column name)
1	P00001	150	15000	0	10	2475000.000000
2	P00001	250	15000	50000	10	4075000.000000
3	P00002	165	12500	0	10	2268750.000000
4	P00005	125	13500	60000	10	1796250.000000
5	P00002	200	10000	50000	10	2150000.000000
6	P00002	200	10000	0	10	2200000.000000
7	P00003	300	10000	50000	10	3250000.000000
8	P00004	120	12000	0	10	1584000.000000
9	P00003	120	12500	55000	10	1595000.000000

Hình 6-15: Khai báo biến thức.

Đối với trường hợp này, khi bạn truy cập vào tập dữ liệu này bằng ngôn ngữ lập trình như: C#, Visual Basic .NET, Visual Basic 6.0, ... thì chỉ có thể lấy dữ liệu của cột 5 ứng với chỉ mục cột thay vì tên.

Trong trường hợp, bạn muốn đặt tên cột dữ liệu ứng với biến thức Quantity*Price thì sử dụng từ khóa AS như ví dụ 6-16.

Ví dụ 6-16: Khai báo từ khóa AS

```
SELECT ProductID As ItemID,
Quantity, Price, Discount,
Quantity*Price As Amount,
Quantity*Price*(1+VATRate/100)-Discount As SalesAmount
FROM SalesInvoiceDetails
WHERE Quantity > 100
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-16.

M® 190**Chương 6: Phát biểu T-SQL cơ bản: dạng truy vấn dữ liệu**

	ItemID	Quantity	Price	Discount	Amount	SalesAmount
1	P00001	150	15000	0	2250000	2475000.000000
2	P00001	250	15000	50000	3750000	4075000.000000
3	P00002	165	12500	0	2062500	2268750.000000
4	P00005	125	13500	60000	1687500	1796250.000000
5	P00002	200	10000	50000	2000000	2150000.000000
6	P00002	200	10000	0	2000000	2200000.000000
7	P00003	300	10000	50000	3000000	3250000.000000
8	P00004	120	12000	0	1440000	1584000.000000
9	P00003	120	12500	55000	1500000	1595000.000000

Hình 6-16: Sử dụng từ khóa AS.

Chú ý: Bạn cũng có thể không sử dụng từ khóa AS bằng cách khai báo như ví dụ 6-17.

Ví dụ 6-17: Khai báo phát biểu SELECT không sử dụng từ khóa AS

```
SELECT ProductID, Quantity, Price, Discount,
Quantity* Price Amount,
Quantity*Price*(1+VATRate/100)-Discount InvoiceAmount
FROM SalesInvoiceDetails
WHERE Quantity > 100
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-17.

	ProductID	Quantity	Price	Discount	Amount	InvoiceAmount
1	P00001	150	15000	0	2250000	2475000.000000
2	P00001	250	15000	50000	3750000	4075000.000000
3	P00002	165	12500	0	2062500	2268750.000000
4	P00005	125	13500	60000	1687500	1796250.000000
5	P00002	200	10000	50000	2000000	2150000.000000
6	P00002	200	10000	0	2000000	2200000.000000
7	P00003	300	10000	50000	3000000	3250000.000000
8	P00004	120	12000	0	1440000	1584000.000000
9	P00003	120	12500	55000	1500000	1595000.000000

Hình 6-17: Không sử dụng từ khóa AS.

7. PHÁT BIỂU SELECT VÀ MỆNH ĐỀ WHERE

Mệnh đề WHERE cho phép bạn trích lọc dữ liệu trong một hay nhiều bảng dữ liệu dựa vào biểu thức sau mệnh đề WHERE.

[WHERE <search_condition>]

Một số phép toán thường được sử dụng sau mệnh đề WHERE như: LIKE, AND, OR, NOT, BETWEEN, IS NULL, IS NOT NULL, IN và một số phép toán so sánh như: =, >, <, <>, !=, >=, <=.

7.1. Mệnh đề WHERE và phép toán LIKE

Phép toán LIKE cho phép bạn sử dụng trong mệnh đề WHERE để liệt kê danh sách mẫu tin có giá trị tại cột nào đó so sánh với giá trị, biểu thức hay cột dữ liệu.

Thường thì phép toán LIKE được sử dụng với ký tự % ứng với giá trị tùy chọn. Chẳng hạn, bạn liệt kê danh sách khách hàng có tên chứa chữ Vietnam như ví dụ 6-18.

Ví dụ 6-18: Khai báo phép toán LIKE

```
SELECT CompanyNameInVietnamese,
       ContactName, ContactTitle, Address
  FROM Customers
 WHERE CompanyNameInVietnamese LIKE '%Vietnam%'
 ORDER BY CustomerID DESC
 GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-18.

	CompanyNameInVietnamese	ContactName	ContactTitle	Address
1	Trung tâm giáo dục Vietnam	Mr. Lee Bob	Country Manager	1 An Huy
2	Công ty Cổ phần ReruitVietnam	Mr. John Bob	Country Manager	1 An Trung
3	Công ty Cổ phần Suzumi Vietnam	Mr. Kazaminui	Director	101 Hoàng Văn Thụ
4	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	Mr. Li Chan Heo	Director	1001 Nguyễn Vé
5	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	Mr. Leo Chan Hy	Director	101 Nguyễn Tuệ
6	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam	Mr. Hall	Director	1 Nguyễn Huệ
7	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	Mr. John	Technical Manager	1 Lê Duẩn

Hình 6-18: Sử dụng phép toán LIKE.

7.2. Mệnh đề WHERE và phép toán AND

Phép toán AND cho phép bạn kết hợp hai biểu thức chẳng hạn như khai báo trong ví dụ 6-19.

M® 192

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu**Ví dụ 6-19: Khai báo phép toán AND**

```
SELECT CompanyNameInVietnamese,
ContactName, ContactTitle, Address
FROM Customers
WHERE ProvinceID = 'HCM'
AND ContactTitle = 'Director'
ORDER BY CustomerID DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-19.

	CompanyNameInVietnamese	ContactName	ContactTitle	Address
1	Công ty Cổ phần Suzumi Vietnam	Mr. Kazaminui	Director	101 Hoàng Văn Thụ
2	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam	Mr. Hall	Director	1 Nguyễn Huệ

Hình 6-19: Sử dụng phép toán AND.

7.3. Mệnh đề WHERE và phép toán IS NULL

Do NULL không phải là giá trị trong cơ sở dữ liệu, chính vì vậy bạn phải sử dụng IS NULL hay IS NOT NULL để so sánh giá trị trong cột dữ liệu với NULL.

Ví dụ 6-20: Khai báo phép toán IS NULL

```
SELECT CompanyNameInVietnamese,
ContactName, ContactTitle, Address
FROM Customers
WHERE USDAccountNo IS NULL
ORDER BY CustomerID DESC
GO
```

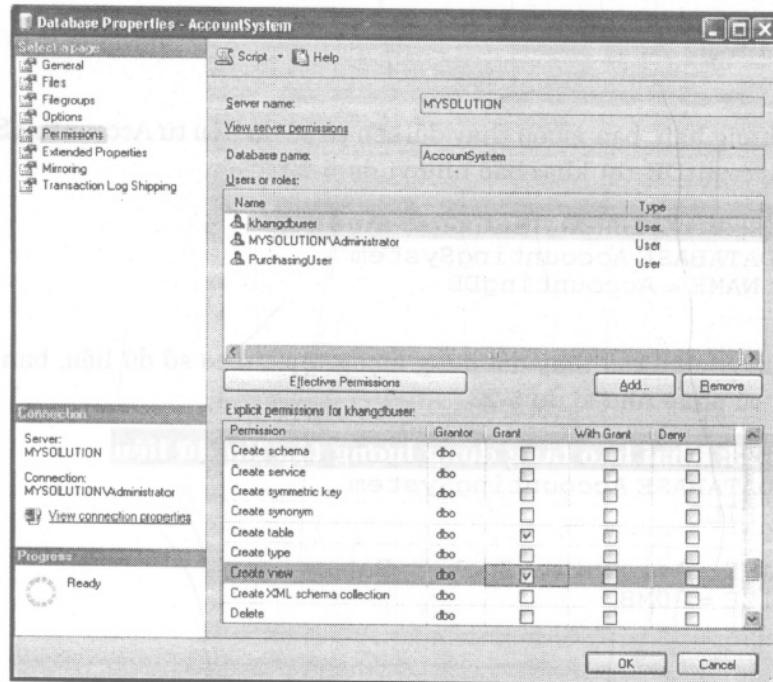
Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-20.

	CompanyNameInVietnamese	ContactName	ContactTitle	Address
1	Trung tâm giáo dục Vietnam	Mr. Lee Bob	Country Manager	1 An Huy
2	Công ty Cổ phần ReruitVietnam	Mr. John Bob	Country Manager	1 An Trung
3	Công ty Đa quốc gia UFCA	Mr. Handar, Ms. Mori	Country Manager	1 Hoà An
4	Tập đoàn UCLIA USA	Ms. Lenard	Country Manager	98A Bình Điền, Tp Biên Hòa

Hình 6-20: Sử dụng phép toán IS NULL.

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

161

**Hình 5-20: Quyền tạo đối tượng View.**

Chú ý: Chúng ta sẽ tiếp tục tìm hiểu phát biểu CREATE PROCEDURE, CREATE TRIGGER, CREATE DEFAULT, CREATE FUNCTION trong tập kế tiếp.

2. PHÁT BIỂU ALTER

Sau khi tìm hiểu cách khai báo phát biểu CREATE dùng để tạo cơ sở dữ liệu, Table và View trong phần trên, nếu có nhu cầu thay đổi cấu trúc của chúng thì bạn có thể sử dụng trình MS hoặc phát biểu ALTER.

2.1. Phát biểu ALTER DATABASE

Để đổi tên cơ sở dữ liệu, bạn có thể sử dụng phát biểu ALTER DATABASE với cú pháp như sau:

```
ALTER DATABASE database_name
{
    <add_or_modify_files>
    | <add_or_modify_filegroups>
    | <set_database_options>
    | MODIFY NAME = new_database_name
```

```
| COLLATE collation_name  
}  
[;]
```

Chẳng hạn, bạn muốn thay đổi tên cơ sở dữ liệu từ AccountingSystem thành AccountDB thì khai báo như ví dụ 5-22.

Ví dụ 5-22: Khai báo đổi tên cơ sở dữ liệu

```
ALTER DATABASE AccountingSystem  
MODIFY NAME = AccountingDB  
GO
```

Khi có nhu cầu thay đổi dung lượng tập tin cơ sở dữ liệu, bạn có thể sử dụng cú pháp như ví dụ 5-23.

Ví dụ 5-23: Khai báo tăng dung lượng tập tin dữ liệu

```
ALTER DATABASE AccountingSystem  
MODIFY FILE  
    (  
        NAME = AccountingSystem_Da  
        SIZE = 30MB  
    );  
GO
```

Trong trường hợp thêm tập tin .ndf cho cơ sở dữ liệu, bạn sử dụng cú pháp như ví dụ 5-24.

Ví dụ 5-24: Khai báo thêm tập tin .ndf

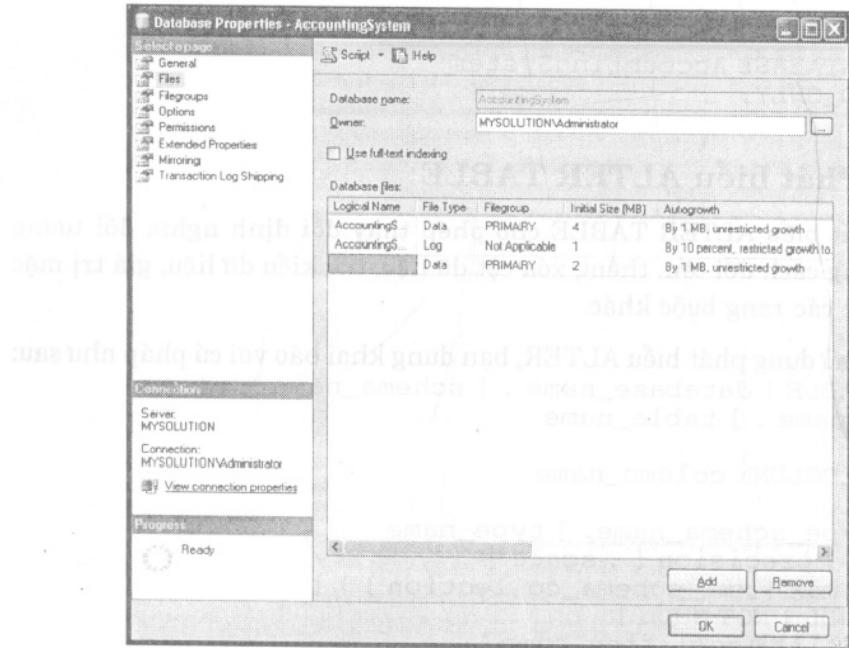
```
ALTER DATABASE AccountingSystem  
ADD FILE
```

```
ADD FILE
(
    NAME = AccountingSystem_Data2,
    FILENAME = 'C:\AccountingSystem_Data2.ndf',
    SIZE = 5MB,
    MAXSIZE = 100MB,
    FILEGROWTH = 5MB
)
GO
```

Chú ý: Sau khi thực thi phát biểu trên, bạn có thể tìm thấy tập tin .ndf xuất hiện trong ngăn Properties | File của cơ sở dữ liệu AccountingSystem như hình 5-21.

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

163

**Hình 5-21:** Thêm tập tin .ndf.

Tương tự như vậy, nếu muốn loại bỏ tập tin .ndf của cơ sở dữ liệu trên thì bạn sử dụng cú pháp như ví dụ 5-25.

Ví dụ 5-25: Khai báo loại bỏ tập tin .ndf

```
ALTER DATABASE AccountingSystem
REMOVE FILE AccountingSystem_Data2;
GO
```

Ngoài ra, bạn cũng có thể dời vị trí của tập tin .mdf hay .ndf bằng cách sử dụng cú pháp như ví dụ 5-26.

Ví dụ 5-26: Khai báo dời vị trí của tập tin .mdf hay .ndf

```
ALTER DATABASE AccountingSystem
MODIFY FILE
(
    NAME = AccountingSystem_Data2,
    FILENAME = N'D:\AccountingSystem_Data2.ndf'
);
GO
```

Lưu ý: Nếu muốn cấu hình cơ sở dữ liệu chỉ cho phép một người sử dụng hoặc đa người dùng chỉ đọc thì sử dụng cú pháp với mệnh đề SET.

Chẳng hạn, bạn muốn cài đặt cơ sở dữ liệu có trạng thái chỉ đọc thì khai báo như ví dụ 5-27.

Ví dụ 5-27: Khai báo cơ sở dữ liệu chỉ đọc

```
ALTER DATABASE AccountingSystem
SET READ_ONLY;
GO
```

2.2. Phát biểu ALTER TABLE

Phát biểu ALTER TABLE cho phép thay đổi định nghĩa đối tượng Table bằng cách đổi tên, thêm, xóa cột dữ liệu, đổi kiểu dữ liệu, giá trị mặc định hoặc các ràng buộc khác.

Để sử dụng phát biểu ALTER, bạn dùng khai báo với cú pháp như sau:

```
ALTER TABLE [ database_name . [ schema_name ] . |
schema_name . ] table_name
{
    ALTER COLUMN column_name
    {
        [ type_schema_name . ] type_name
        [ ( { precision [ , scale ]
        | max | xml_schema_collection } ) ]
        [ NULL | NOT NULL ]
        [ COLLATE collation_name ]
        | { ADD | DROP } { ROWGUIDCOL | PERSISTED }
    }
    | [ WITH { CHECK | NOCHECK } ] ADD
    {
        <column_definition>
        | <computed_column_definition>
        | <table_constraint>
    } [ ,...n ]
    | DROP
    {
        { CONSTRAINT } constraint_name
        [ WITH ( <drop_clustered_constraint_option>
        [ ,...n ] ) ]
        | COLUMN column_name
    } [ ,...n ]
    | [ WITH { CHECK | NOCHECK } ]
        { CHECK | NOCHECK } CONSTRAINT
        { ALL | constraint_name [ ,...n ] }
    | { ENABLE | DISABLE } TRIGGER
        { ALL | trigger_name [ ,...n ] }
    | SWITCH [ PARTITION
        source_partition_number_expression ]
        TO [ schema_name . ] target_table
        [ PARTITION
            target_partition_number_expression ]
    }
    [ ; ]
```

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu165 

Chẳng hạn, để thêm cột dữ liệu có tên MaxDebt có kiểu int và giá trị mặc định là 100.000.000 vào bảng Customers thì khai báo như ví dụ 5-28.

Ví dụ 5-28: Khai báo thêm cột dữ liệu

```
ALTER TABLE Customers
ADD MaxDebt int DEFAULT 100000000
GO
```

Trong trường hợp dữ liệu đã tồn tại trong bảng Customers, sau khi thêm cột dữ liệu vào bảng, bạn muốn giá trị mặc định diền vào cột MaxDebt của mẫu tin đã tồn tại thì sử dụng mệnh đề "with values" được khai báo như ví dụ 5-29.

Ví dụ 5-29: Khai báo diền dữ liệu vào cột vừa thêm

```
ALTER TABLE Customers
ADD MaxDebt int DEFAULT 100000000
with values
GO
```

Nếu muốn loại bỏ cột dữ liệu khỏi bảng dữ liệu, bạn có thể sử dụng cú pháp như ví dụ 5-30.

Ví dụ 5-30: Khai báo loại bỏ cột dữ liệu

```
ALTER TABLE Customers
DROP COLUMN MaxDebt
GO
```

Chú ý: Lỗi sẽ phát sinh nếu cột dữ liệu có ràng buộc (CONSTRAINT). Giả sử, chúng ta vừa khai báo cột MaxDebt có giá trị mặc định, nên khi loại bỏ cột này lỗi phát sinh như sau:

```
Msg 5074, Level 16, State 1, Line 1
The object 'DF__Customers__MaxDe__52AE4273' is dependent
on column 'MaxDebt'.
Msg 4922, Level 16, State 9, Line 1
ALTER TABLE DROP COLUMN MaxDebt failed because one or more
objects access this column.
```

Như vậy, để loại bỏ cột MaxDebt trong bảng dữ liệu Customers, bạn phải loại bỏ ràng buộc này với cú pháp như ví dụ 5-31.

Ví dụ 5-31: Khai báo xóa CONSTRAINT

```
ALTER TABLE Customers
DROP CONSTRAINT DF__Customers__MaxDe__52AE4273
GO
```

Chú ý: Để biết được CONSTRAINT của đối tượng Default trong bảng dữ liệu, bạn có thể khai báo phát biểu SELECT có cấu trúc như ví dụ 5-32.

M® 166**Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu****Ví dụ 5-32: Khai báo tìm tên CONSTRAINT**

```
SELECT name FROM SYS.objects
where type = 'D'
and object_id =
(
    select default_object_id
    from sys.columns
    where name = 'MaxDebt'
    and object_id =
        (
            select object_id
            from sys.tables
            where name = 'Customers'
        )
)
GO
```

Trong trường hợp thêm cột dữ liệu là khóa ngoại, bạn có thể khai báo với cú pháp tương tự như ví dụ 5-33.

Ví dụ 5-33: Khai báo thêm cột khóa ngoại

```
ALTER TABLE Banks
ADD ProvinceID char(3)
REFERENCES Provinces(ProvinceID)
GO
```

Trong đó, cột ProvinceID vừa thêm vào bảng Banks được tham chiếu (N-1) đến cột ProvinceID là cột khóa chính trong bảng Provinces.

Nếu muốn thay đổi kiểu dữ liệu của cột đang tồn tại, bạn khai báo tương tự như ví dụ 5-34.

Ví dụ 5-34: Khai báo thay đổi kiểu dữ liệu

```
ALTER TABLE Customers
ALTER COLUMN MaxDebt decimal (5, 2)
GO
```

2.3. Phát biểu ALTER VIEW

Phát biểu ALTER VIEW cho phép thay đổi cấu trúc đối tượng VIEW bằng cách định nghĩa lại phát biểu SQL dạng SELECT, thêm, loại bỏ cột dữ liệu, mệnh đề.

Để sử dụng phát biểu ALTER, bạn dùng khai báo với cú pháp như sau:

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

Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu

167

Trong đó, view_name là tên của đối tượng View cần thay đổi, select_statement là phát biểu SQL dạng SELECT. Giả sử, chúng ta đã tạo View có tên vwPayments với cấu trúc như ví dụ 5-35.

Ví dụ 5-35: Khai báo tạo View có tên vwPayments

```
CREATE VIEW vwPayments
AS
SELECT
    dbo.Suppliers.SupplierID,
    dbo.Suppliers.CompanyNameInVietnamese,
    dbo.Payments.PaymentNo,
    dbo.Payments.PaymentDate,
    dbo.Payments.Amount,
    dbo.Payments.CurrencyID,
    dbo.Payments.ExchangeRate,
    dbo.Payments.PaymentAmount
FROM dbo.Suppliers
INNER JOIN dbo.Payments
ON dbo.Suppliers.SupplierID = dbo.Payments.SupplierID
```

Sau đó, nếu có nhu cầu thêm hay loại bỏ cột dữ liệu thì sử dụng phát biểu ALTER VIEW. Chẳng hạn, bạn thêm cột ContactName trong bảng Suppliers vào trong View bằng cách khai báo như ví dụ 5-36.

Ví dụ 5-36: Khai báo thay đổi cấu trúc View

```
ALTER VIEW vwPayments
AS
SELECT
    dbo.Suppliers.SupplierID,
    dbo.Suppliers.CompanyNameInVietnamese,
    dbo.Suppliers.ContactName,
    dbo.Payments.PaymentNo,
    dbo.Payments.PaymentDate,
    dbo.Payments.Amount,
    dbo.Payments.CurrencyID,
    dbo.Payments.ExchangeRate,
    dbo.Payments.PaymentAmount
FROM dbo.Suppliers
INNER JOIN dbo.Payments
ON dbo.Suppliers.SupplierID = dbo.Payments.SupplierID
GO
```

Tương tự như trong trường hợp tạo View, bạn có thể sử dụng một trong ba tùy chọn của thuộc tính View với từ WITH.

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

M[®] 168**Chương 5: Phát biểu T-SQL dạng định nghĩa dữ liệu**

Để sử dụng thuộc tính này, bạn có thể khai báo View trong ví dụ 5-35 hay 5-36 thành ví dụ 5-37 với từ khóa ENCRYPTION như sau.

Ví dụ 5-37: Khai báo mã hóa View

```
ALTER VIEW vwPayments
    WITH ENCRYPTION
AS
SELECT
    dbo.Suppliers.SupplierID,
    dbo.Suppliers.CompanyNameInVietnamese,
    dbo.Suppliers.ContactName,
    dbo.Payments.PaymentNo,
    dbo.Payments.PaymentDate,
    dbo.Payments.Amount,
    dbo.Payments.CurrencyID,
    dbo.Payments.ExchangeRate,
    dbo.Payments.PaymentAmount
FROM dbo.Suppliers
INNER JOIN dbo.Payments
ON dbo.Suppliers.SupplierID = dbo.Payments.SupplierID
GO
```

Chú ý: Để tạo View với cấu trúc mà nó cho phép bạn lấy ra tập dữ liệu như mong muốn thì bạn cần kết hợp dữ liệu từ các bảng hay View với nhiều mệnh đề khác nhau như: JOIN, WHERE, GROUP BY.

3. PHÁT BIỂU DROP

Phát biểu DROP cho phép bạn xóa cơ sở dữ liệu, đối tượng cơ sở dữ liệu khỏi cơ sở dữ liệu hiện hành. Chẳng hạn, bạn có thể sử dụng phát biểu DROP DATABASE, DROP TABLE, DROP VIEW.

3.1. Phát biểu DROP DATABASE

Phát biểu DROP DATABASE cho phép bạn xóa cơ sở dữ liệu cùng với các đối tượng cơ sở dữ liệu bằng cú pháp như sau:

```
DROP DATABASE { database_name | database_snapshot_name }
[ ,...n ]
[ ; ]
```

Chẳng hạn, bạn khai báo để xóa cơ sở dữ liệu có tên AccountingSystem với cú pháp như ví dụ 5-38.

Ví dụ 5-38: Khai báo xóa cơ sở dữ liệu

```
DROP DATABASE AccountingSystem
GO
```

3.2. Phát biểu DROP TABLE

Phát biểu DROP TABLE cho phép bạn xóa bảng dữ liệu cùng với các thành phần liên quan với cú pháp như sau:

```
DROP TABLE [ database_name . [ schema_name ] .
| schema_name . ]
    table_name [ ,...n ] [ ; ]
```

Trong đó, database_name là tên cơ sở dữ liệu nếu bạn không đứng tại cơ sở dữ liệu có đối tượng cần xóa, schema_name ứng với chủ nhân của đối tượng cần xóa và table_name là tên đối tượng Table cần xóa.

Chẳng hạn, để xóa bảng dữ liệu có tên Users do tài khoản người sử dụng là khangdbuser tạo ra, bạn thực hiện như ví dụ 5-39.

Ví dụ 5-39: Khai báo xóa bảng dữ liệu

```
DROP TABLE khangdbuser.Users
GO
```

Nếu bạn đang đứng tại cơ sở dữ liệu AccountSystem mà xóa bảng dữ liệu có tên SalesInvoiceDetails trong cơ sở dữ liệu AccountingSystem thì khai báo như ví dụ 5-40.

Ví dụ 5-40: Khai báo xóa bảng dữ liệu trong cơ sở dữ liệu khác

```
DROP TABLE AccountingSystem.dbo.SalesInvoiceDetails
GO
```

Chú ý: Nếu bạn xóa bảng dữ liệu đang có quan hệ với bảng khác thì lỗi sẽ phát sinh. Ví dụ, nếu bạn xóa bảng dữ liệu có tên Banks thì thông báo lỗi sẽ phát sinh như sau:

```
Msg 3726, Level 16, State 1, Line 1
Could not drop object 'AccountingSystem.dbo.Banks'
because it is referenced by a FOREIGN KEY constraint.
```

3.3. Phát biểu DROP VIEW

Tương tự như hai phát biểu DROP DATABASE và DROP TABLE vừa trình bày ở trên, phát biểu DROP VIEW cho phép bạn xóa đối tượng VIEW khỏi cơ sở dữ liệu với cú pháp như sau:

```
DROP VIEW [ schema_name . ]
view_name [ ,...n ] [ ; ]
```

Trong đó, schema_name là tên cơ sở dữ liệu và chủ nhân của đối tượng View, view_name chính là tên đối tượng View.

Chẳng hạn, bạn có thể xóa đối tượng View có tên vwPayments vừa tạo ra trong các ví dụ trên như ví dụ 5-41.

Ví dụ 5-41: Khai báo xóa View

```
DROP VIEW vwPayments  
GO
```

Chú ý: Bạn có thể tham khảo cách xóa các đối tượng cơ sở dữ liệu khác như: DROP PROCEDURE, DROP TRIGGER, DROP RULE, DROP DEFAULT trong tập kế tiếp.

4. KẾT CHƯƠNG

Chúng ta vừa tiếp tục tìm hiểu phát biểu dạng DDL dùng để tạo cơ sở dữ liệu, đối tượng Table và View rồi đến phát biểu dùng để thay đổi cấu trúc ALTER, sau đó là phát biểu DROP dùng để xóa cơ sở dữ liệu hay đối tượng Table và View.

Sau khi tạo cơ sở dữ liệu hay đối tượng cơ sở dữ liệu, bạn có thể tìm thấy thông tin của chúng trong các bảng dữ liệu hệ thống tương ứng là sys.databases, sys.tables, sys.columns và sys.views.

Trong hai chương kế tiếp, chúng ta sẽ tiếp tục tìm hiểu phát biểu SQL dạng DML mà đại diện của chúng bao gồm phát biểu: SELECT, INSERT, UPDATE và DELETE với các mệnh đề liên quan.

Chương 6:

PHÁT BIỂU T-SQL CƠ BẢN DẠNG TRUY VẤN DỮ LIỆU

Tóm tắt chương 6

Bạn đã tìm hiểu chi tiết về các phát biểu CREATE, ALTER và DROP dùng cho cơ sở dữ liệu và đối tượng của cơ sở dữ liệu.

Trong chương này, chúng ta cũng tìm hiểu một số phát biểu SELECT dùng để truy vấn dữ liệu với các mệnh đề như: ORDER BY, TOP, DISTINCT, INTO, WHERE, GROUP BY, COMPUTE và JOIN, đây là phát biểu thường được sử dụng trong quá trình thi hành ứng dụng.

Các vấn đề chính sẽ được đề cập:

- ✓ Phát biểu SELECT cơ bản.
- ✓ Mệnh đề ORDER BY.
- ✓ Từ khóa TOP, DISTINCT.
- ✓ Phát biểu SELECT và mệnh đề INTO.
- ✓ Phát biểu SELECT và từ khóa AS.
- ✓ Mệnh đề WHERE.
- ✓ Mệnh đề GROUP BY, COMPUTE.
- ✓ Mệnh đề INNER JOIN, LEFT JOIN.
- ✓ Phép toán UNION, EXCEPT, INTERSECT.
- ✓ Phân trang dữ liệu với phát biểu WITH.
- ✓ Lấy tập mẫu tin ngẫu nhiên.

1. PHÁT BIỂU SELECT

Sau khi triển khai ứng dụng, hầu hết các tác vụ của người sử dụng đều liên quan đến 4 phát biểu SELECT, INSERT, UPDATE và DELETE; trong đó phát biểu SELECT là phát biểu được sử dụng nhiều nhất.

Tuy nhiên, trong chương này chúng ta tập trung tìm hiểu phát biểu SELECT với mệnh đề thường sử dụng khi truy vấn dữ liệu hay khai báo trong đối tượng VIEW và STORED PROCEDURE.

Với chức năng chính là cho phép chúng ta truy vấn dữ liệu dưới nhiều hình thức khác nhau dựa vào các mệnh đề WHERE, JOIN, GROUP BY, ORDER BY, TOP, ... để lấy ra tập dữ liệu theo yêu cầu.

Ngoài các mệnh đề vừa giới thiệu ở trên, thông qua phát biểu SELECT bạn có thể sử dụng biểu thức, hàm để tạo ra cột dữ liệu dựa trên các cột dữ liệu đã có hay giá trị cụ thể nhằm làm phong phú dữ liệu mà bạn cần lấy ra.

Chú ý: Phát biểu SELECT kết hợp cùng với các mệnh đề liên quan sẽ cho phép bạn khai báo đối tượng View và Procedure thường trú trong cơ sở dữ liệu.

Trước khi bắt đầu làm việc với phát biểu SELECT với các mệnh đề, chúng ta thử tìm hiểu cấu trúc của phát biểu này thông qua cú pháp như sau:

```
SELECT select_list
[ INTO new_table_name ]
FROM table_list
[ WHERE search_conditions ]
[ GROUP BY group_by_list ]
[ HAVING search_conditions ]
[ ORDER BY order_list [ ASC | DESC ] ]
```

Chú ý: Cấu trúc của phát biểu SELECT rất phức tạp, chúng ta chỉ tìm hiểu phát biểu này qua từng trường hợp cụ thể.

1.1. Phát biểu SELECT đơn giản

Trong đó, select_list là danh sách gồm một hay nhiều cột dữ liệu hay biểu thức tính toán tạo nên cột dữ liệu cần kết xuất. Dấu * tương ứng với mọi cột dữ liệu có trong bảng dữ liệu nguồn được khai báo sau mệnh đề FROM.

Ví dụ, bạn muốn liệt kê tất cả các cột có trong bảng dữ liệu thì sử dụng cú pháp như sau:

```
SELECT * FROM TABLENAME
```

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu173 

Chẳng hạn, bạn khai báo phát biểu để liệt kê mọi thông tin của danh sách khách hàng trong bảng Customers thì sử dụng cú pháp như ví dụ 6-1.

Ví dụ 6-1: Khai báo liệt kê danh sách khách hàng

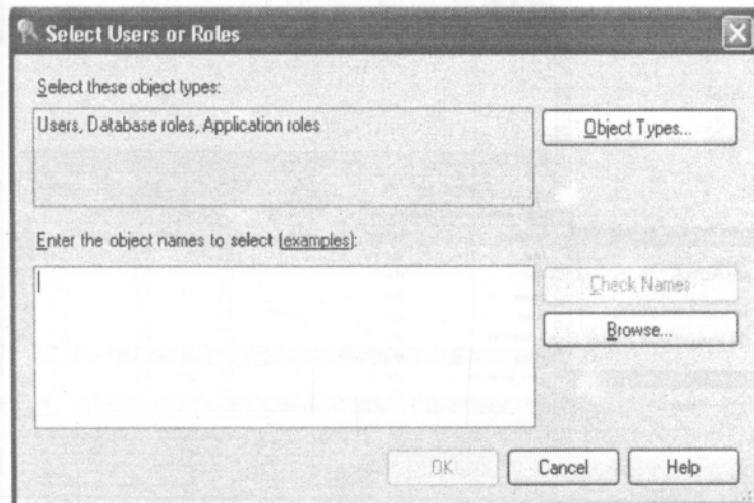
```
SELECT * FROM Customers
GO
```

Nếu thực thi phát biểu trên, bạn có thể tìm thấy kết quả trình bày như hình 6-1.

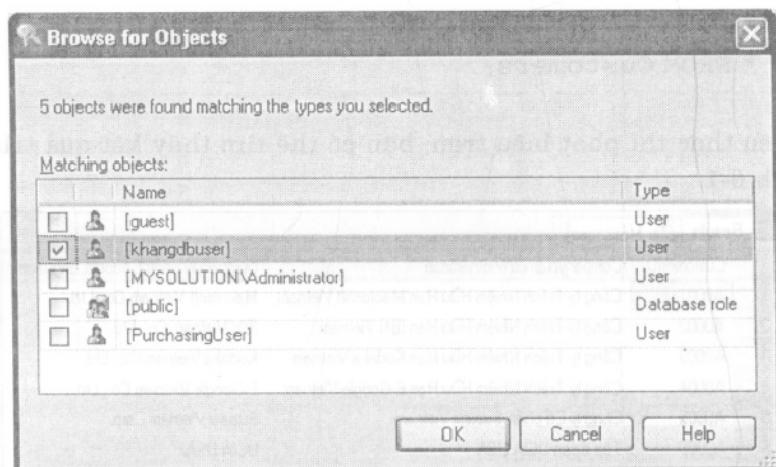
CustomerID	CompanyNameInVietnamese	CompanyNameInSecondLanguage
1 A0001	Công ty Trách Nhiệm Hữu Hạn Macrosoft Vietnam	Macrosoft Vietnam Co., Ltd.
2 A0002	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	IBM Vietnam Co., Ltd.
3 A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	Kodaka Vietnam Co., Ltd.
4 A0004	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	E-Google Vietnam Co., Ltd.
5 A0005	Công ty Cổ phần Suzumi Vietnam	Suzumi Vietnam Corp.
6 A0006	Tập đoàn UCIA USA	UCIA USA
7 A0007	Công ty Đa quốc gia UFCA	UFCA Corp.
8 A0008	Công ty Cổ phần ReruiVietnam	ReruiVietnam Corp.
9 A0009	Trung tâm giáo dục Vietnam	Vietnam Education Center

Hình 6-1: Danh sách khách hàng.

Chú ý: Để thực thi phát biểu SELECT hay liệt kê mẫu tin trong bảng, người sử dụng cần có quyền SELECT. Chẳng hạn, nếu bạn muốn gán quyền truy vấn dữ liệu bằng MS cho bảng Categories, bạn chọn vào nút Add và cửa sổ xuất hiện như hình 6-1-1.

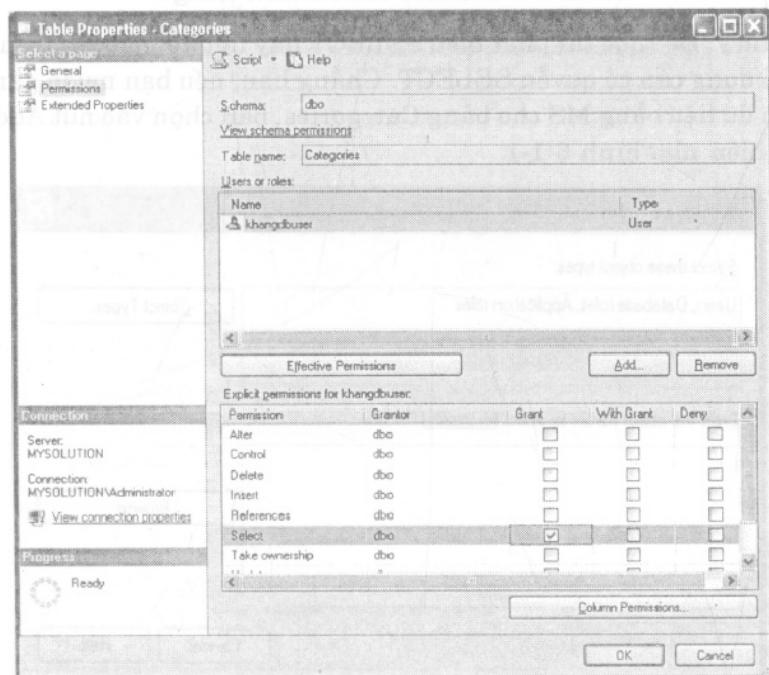
**Hình 6-1-1: Thêm tài khoản người sử dụng.**

Nhấn nút Browse, cửa sổ liệt kê danh sách người sử dụng xuất hiện như hình 6-1-2.



Hình 6-1-2: Chọn tên người sử dụng.

Nhấn nút OK, rồi chọn vào CheckBox ứng với quyền SELECT như hình 6-1-3.



Hình 6-1-3: Cấp quyền SELECT.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

175

Trong trường hợp chỉ định một số cột dữ liệu trong bảng dữ liệu thì khai báo với cú pháp như sau:

```
SELECT Col1, Col2, Col3 FROM TableName
```

Chẳng hạn, bạn liệt kê mã, tên và địa chỉ khách hàng trong bảng Customers thì sử dụng cú pháp như ví dụ 6-2.

Ví dụ 6-2: Khai báo liệt kê khách hàng

```
SELECT CustomerID, CompanyNameInVietnamese, Address
FROM Customers
GO
```

Khi thực thi phát biểu trên, bạn có thể tìm thấy kết quả trình bày như hình 6-2.

	CustomerID	CompanyNameInVietnamese	Address
1	A0001	Công ty Trách Nhiệm Hữu Hạn Macrosoft Vietnam	1 Lê Duẩn
2	A0002	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	1 Nguyễn Huệ
3	A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	101 Nguyễn Tuệ
4	A0004	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	1001 Nguyễn Vệ
5	A0005	Công ty Cổ phần Suzumi Vietnam	101 Hoàng Văn Thụ
6	A0006	Tập đoàn UCIA USA	98A Bình Điền, Tp Biên Hòa
7	A0007	Công ty Đa quốc gia UFCA	1 Hòa An
8	A0008	Công ty Cổ phần ReruitVietnam	1 An Trung
9	A0009	Trung tâm giáo dục Vietnam	1 An Huy

Hình 6-2: Danh sách khách hàng.

1.2. Biểu thức trong phát biểu SELECT

Nếu bạn khai báo biểu thức trong phát biểu SELECT thì sử dụng cú pháp tương tự như sau:

```
SELECT Col1, Col2, Col3+Col4 AS Col34, Expression1 AS
Col5
FROM TableName
```

Chẳng hạn, bạn thiết lập biểu thức bằng cách lấy cột ContactName cộng với cột ContactTitle trong bảng Customers rồi đặt tên là ContactPerson như ví dụ 6-3.

Ví dụ 6-3: Khai báo biến thức trong phát biểu SELECT

```
SELECT CompanyNameInVietnamese,
       ContactName + ' - ' + ContactTitle
    As ContactPerson
   FROM Customers
  GO
```

Lưu ý: Tùy thuộc vào kiểu dữ liệu, bạn sử dụng phép toán tương ứng cũng như một số hàm có sẵn trong SQL Server 2005.

Khi thực thi phát biểu trên, bạn có thể tìm thấy kết quả trình bày như hình 6-3.



	CompanyNameInVietnamese	ContactPerson
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	Mr. John-Techical Manager
2	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	Mr. Hall-Director
3	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	Mr. Leo Chan Hy-Director
4	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	Mr. Li Chan Heo-Director
5	Công ty Cổ phần Suzumi Vietnam	Mr. Kazaminui-Director
6	Tập đoàn UCIA USA	Ms. Lenard-Country Manager
7	Công ty Đa quốc gia UFCA	Mr. Handar, Ms. Mori-Country Manager
8	Công ty Cổ phần ReruitVietnam	Mr. John Bob-Country Manager
9	Trung tâm giáo dục Vietnam	Mr. Lee Bob-Country Manager

Hình 6-3: Danh sách khách hàng.

2. MỆNH ĐỀ ORDER BY

2.1. Phát biểu SELECT với mệnh đề ORDER BY

Bạn có thể sắp xếp dữ liệu khi lấy ra bằng cách chỉ định mệnh đề ORDER BY với cú pháp như sau:

```
SELECT * FROM TABLENAME
ORDER BY {COLUMN_NAME | EXPRESSIONS}
{ASC | DESC}
```

Trong đó, COLUMN_NAME ứng với tên cột dữ liệu và EXPRESSIONS ứng với biểu thức.

Chẳng hạn, để liệt kê danh sách khách hàng theo thứ tự giảm dần dựa trên cột CustomerID thì khai báo như ví dụ 6-4.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

193 M®

7.4. Mệnh đề WHERE và phép toán IN

Khi biết trước số lượng giá trị hay biểu thức cần so sánh cho cột dữ liệu thì bạn có thể sử dụng phép toán OR như ví dụ 6-20-1.

Ví dụ 6-20-1: Khai báo phép toán OR

```
SELECT CompanyNameInVietnamese,
       ContactName, ContactTitle
    FROM Customers
   WHERE ProvinceID = 'HCM'
     OR ProvinceID = 'DNA'
  ORDER BY CustomerID DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-20-1.

	CompanyNameInVietnamese	ContactName	ContactTitle
1	Công ty Cổ phần Rerui Vietnam	Mr. John Bob	Country Manager
2	Công ty Đa quốc gia UFCAC	Mr. Handar, Ms. Mori	Country Manager
3	Tập đoàn UCIA USA	Ms. Lenard	Country Manager
4	Công ty Cổ phần Suzumi Vietnam	Mr. Kazaminui	Director
5	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam	Mr. Hall	Director
6	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	Mr. John	Technical Manager

Hình 6-20-1: Sử dụng phép toán OR.

Bạn có thể sử dụng phép toán IN hay NOT IN để so sánh cột dữ liệu, biểu thức với tập cột dữ liệu hay tập giá trị. Chẳng hạn, chúng ta có thể khai báo phép toán IN như ví dụ 6-21.

Ví dụ 6-21: Khai báo phép toán IN

```
SELECT CompanyNameInVietnamese, ContactName, ContactTitle  
FROM Customers  
WHERE ProvinceID IN ('HCM', 'HAN', 'DNA')  
ORDER BY CustomerID DESC  
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-21.

7.5. Mệnh đề WHERE và phép toán BETWEEN

Phép toán BETWEEN cho phép bạn lọc dữ liệu theo cột trong khoảng giá trị nào đó. Chẳng hạn, bạn sử dụng phép toán BETWEEN để liệt kê danh sách hóa đơn bán hàng trong bảng SalesInvoices như ví dụ 6-22.

M® 194

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

	CompanyName (in Vietnamese)	ContactName	ContactTitle
1	Công ty Cổ phần Reruit Vietnam	Mr. John Bob	Country Manager
2	Công ty Đa quốc gia UFCA	Mr. Handar, Ms. Mori	Country Manager
3	Tập đoàn UCIA USA	Ms. Lenard	Country Manager
4	Công ty Cổ phần Suzumi Vietnam	Mr. Kazaminui	Director
5	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	Mr. Li Chan Heo	Director
6	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	Mr. Leo Chan Hy	Director
7	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam	Mr. Hall	Director
8	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	Mr. John	Technical Manager

Hình 6-21: Sử dụng phép toán IN.**Ví dụ 6-22: Khai báo phép toán BETWEEN**

```
SELECT InvoiceNo, DueDate, CustomerID
FROM SalesInvoices
WHERE DueDate BETWEEN '1/1/2007'
    AND '12/31/2007'
ORDER BY DueDate DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-22.

	InvoiceNo	DueDate	CustomerID
1	SI00000012	2007-10-20 00:00:00	A0001
2	SI00000013	2007-10-20 00:00:00	A0005
3	SI00000010	2007-10-19 00:00:00	A0001
4	SI00000011	2007-10-19 00:00:00	A0002
5	SI00000009	2007-10-18 00:00:00	A0008
6	SI00000007	2007-10-17 00:00:00	A0006
7	SI00000008	2007-10-17 00:00:00	A0007
8	SI00000005	2007-10-14 10:00:00	A0004
9	SI00000006	2007-10-14 00:00:00	A0005
10	SI00000004	2007-10-13 00:00:00	A0001
11	SI00000003	2007-10-12 00:00:00	A0003
12	SI00000002	2007-10-11 00:00:00	A0002
13	SI00000001	2007-10-10 00:00:00	A0001

Hình 6-22: Sử dụng phép toán BETWEEN.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

195 M®

7.6. Mệnh đề WHERE và phép toán khác

Ngoài các phép toán trên, bạn cũng có thể sử dụng phép toán `=`, `>`, `<`, `<>`, `!=`, `>=`, `<=` trong mệnh đề WHERE. Chẳng hạn, bạn sử dụng phép toán `!=` để liệt kê danh sách sản phẩm trong bảng SalesInvoiceDetails như ví dụ 6-23.

Ví dụ 6-23: Khai báo phép toán !=

```
SELECT InvoiceNo, ProductID, Quantity,  
Price, Discount, VATRate,  
Quantity*Price*(1+VATRate/100)-Discount AS InvoiceAmount  
FROM SalesInvoiceDetails  
WHERE Discount != 0  
ORDER BY InvoiceAmount ASC  
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình

	InvoiceNo	ProductID	Quantity	Price	Discount	VATRate	InvoiceAmount
1	S100000013	P00004	5	12500	30000	10	38750.000000
2	S100000012	P00004	5	12500	30000	10	38750.000000
3	S100000013	P00005	5	12500	30000	10	38750.000000
4	S100000013	P00006	5	12500	30000	10	38750.000000
5	S100000009	P00004	5	12500	15000	10	53750.000000
6	S100000012	P00002	25	10500	30000	10	258750.000000
7	S100000007	P00006	25	14500	10000	10	388750.000000
8	S100000008	P00003	35	12500	30000	10	451250.000000
9	S100000010	P00004	35	12500	30000	10	451250.000000
10	S100000010	P00002	35	12500	30000	10	451250.000000
11	S100000010	P00003	35	12500	30000	10	451250.000000
12	S100000009	P00003	35	12500	15000	10	466250.000000
13	S100000008	P00006	35	14500	30000	10	528250.000000
14	S100000002	P00003	100	10000	100000	10	1000000.000000
15	S100000003	P00002	100	10000	50000	10	1050000.000000
16	S100000001	P00001	100	10000	50000	10	1050000.000000
17	S100000005	P00003	120	12500	55000	10	1595000.000000
18	S100000007	P00005	125	13500	60000	10	1796250.000000
19	S100000001	P00002	200	10000	50000	10	2150000.000000
20	S100000003	P00003	300	10000	50000	10	3250000.000000
21	S100000005	P00001	250	15000	50000	10	4075000.000000

Hình 6-23: Sử dụng phép toán !=

8. PHÁT BIỂU SELECT VỚI MỆNH ĐỀ GROUP BY

Trở lại phần trình bày từ khóa DISTINCT, từ khóa này cho phép chúng ta chỉ lấy ra một mẫu tin trong N mẫu tin giống nhau hoàn toàn.

Trong trường hợp muốn nhóm những mẫu tin giống nhau theo một hay nhiều dữ liệu, bạn có thể sử dụng mệnh đề GROUP BY.

```
[ GROUP BY [ ALL ] group_by_expression [ ,...n ]
    [ WITH { CUBE | ROLLUP } ]
]
```

Những cột được khai báo sau mệnh đề GROUP BY sẽ được so sánh nếu bằng nhau sẽ nhóm thành một mẫu tin, trong khi đó những cột không theo sau GROUP BY sẽ được áp dụng các hàm gộp như: MAX, MIN, AVG, COUNT, COUNT_BIG, VAR, SUM, CHECKSUM, CHECKSUM_AGG, STDEV, STDEVP, GROUPING, VARP.

Chú ý: Ngoại trừ hàm COUNT, các hàm còn lại sẽ bỏ qua giá trị là NULL trong cột dữ liệu.

8.1. Sử dụng hàm MAX, MIN, COUNT, SUM, AVG

Bạn sử dụng các hàm cơ bản COUNT, MAX, SUM, MIN, AVG trong phát biểu SELECT để tính các giá trị khi lấy và nhóm dữ liệu từ bảng SalesInvoiceDetails như ví dụ 6-24.

Ví dụ 6-24: Sử dụng hàm MAX, MIN, COUNT, SUM, AVG

```
SELECT ProductID ,
COUNT(Quantity) TotalQuantity,
AVG(Price) AvgOfPrice,
MIN(Price) MinOfPrice,
MAX(Price) MaxOfPrice,
SUM(Discoun
t) AS TotalDiscount,
SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
FROM SalesInvoiceDetails
GROUP BY ProductID
ORDER BY TotalAmount DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-24.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

197

ProductID	TotalQuantity	AvgOfPrice	MinOfPrice	MaxOfPrice	TotalDiscount	TotalAmount
1 P0002	7	10857.142857	10000	12500	160000	8667500.000000
2 P0001	4	13125.000000	10000	15000	100000	7806250.000000
3 P0003	6	11666.666666	10000	12500	280000	7213750.000000
4 P0004	7	12214.285714	11500	12500	105000	2245150.000000
5 P0005	2	13000.000000	12500	13500	90000	1835000.000000
6 P0006	4	13500.000000	12500	14500	70000	1299500.000000

Hình 6-24: Sử dụng hàm MAX, MIN, COUNT, SUM và AVG.

Chú ý: Nếu không sử dụng mệnh đề ORDER BY thì dữ liệu sau khi nhóm sẽ không được sắp xếp, chính vì vậy chúng ta cần sử dụng mệnh đề này trong phát biểu SELECT có mệnh đề GROUP BY.

8.2. Sử dụng phép toán ROLLUP

Phép toán ROLLUP cho phép bạn thêm hàng dữ liệu ứng với các hàm tương ứng trong phát biểu SELECT nhưng tính toán trên những mẫu tin đã được GROUP BY.

Giả sử, bạn sử dụng mệnh đề GROUP BY để liệt kê danh sách mã sản phẩm ứng với ba hàm COUNT, AVG và SUM như ví dụ 6-25.

Ví dụ 6-25: Khai báo sử dụng mệnh đề GROUP BY

```
SELECT ProductID,
COUNT(Quantity) TotalQuantity,
AVG(Price) AvgOfPrice,
SUM(Discount) AS TotalDiscount,
SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
FROM SalesInvoiceDetails
GROUP BY ProductID
ORDER BY TotalAmount DESC
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-25.

ProductID	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
1 P0002	7	10857.142857	160000	8667500.000000
2 P0001	4	13125.000000	100000	7806250.000000
3 P0003	6	11666.666666	280000	7213750.000000
4 P0004	7	12214.285714	105000	2245150.000000
5 P0005	2	13000.000000	90000	1835000.000000
6 P0006	4	13500.000000	70000	1299500.000000

Hình 6-25: Sử dụng hàm COUNT, SUM và AVG.

M® 198**Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu**

Bằng cách sử dụng phép toán ROLLUP, bạn có thể tìm thấy mẫu tin mới thêm vào đầu hay cuối tập dữ liệu phụ thuộc vào mệnh đề ORDER BY và mệnh đề GROUP BY được áp dụng cho tập mẫu tin kết quả đã áp dụng mệnh đề GROUP BY trước đó.

Chẳng hạn, bạn khai báo ví dụ 6-25 có sử dụng phép toán ROLLUP như ví dụ 6-26.

Ví dụ 6-26: Khai báo phép toán ROLLUP

```
SELECT ProductID,
       COUNT(Quantity) TotalQuantity,
       AVG(Price) AvgOfPrice,
       SUM(Discoun t) AS TotalDiscount,
       SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
       TotalAmount
  FROM SalesInvoiceDetails
 GROUP BY ProductID WITH ROLLUP
 ORDER BY TotalAmount DESC
 GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày với mẫu tin đầu tiên là kết quả áp dụng mệnh đề GROUP BY trên 6 mẫu tin còn lại với giả định giá trị ProductID là NULL như hình 6-26.

	ProductID	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
1	NULL	30	12133.333333	805000	29067150.000000
2	P00002	7	10857.142857	160000	8667500.000000
3	P00001	4	13125.000000	100000	7806250.000000
4	P00003	6	11666.666666	280000	7213750.000000
5	P00004	7	12214.285714	105000	2245150.000000
6	P00005	2	13000.000000	90000	1835000.000000
7	P00006	4	13500.000000	70000	1299500.000000

Hình 6-26: Sử dụng phép toán ROLLUP.

Lưu ý: Để xử lý Null trong cột của hàng đầu tiên, bạn có thể khai báo lại ví dụ trên như sau:

```
WITH NULLVALUE
AS
(
    SELECT ProductID,
           COUNT(Quantity) TotalQuantity,
           AVG(Price) AvgOfPrice,
           SUM(Discoun t) AS TotalDiscount,
```

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

199

```

        SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
    FROM SalesInvoiceDetails
    GROUP BY ProductID WITH ROLLUP
)
SELECT ISNULL(ProductID, 'Summary'),
TotalQuantity, AvgOfPrice, TotalDiscount, TotalAmount
FROM NULLVALUE
ORDER BY TotalAmount DESC
GO

```

Trong trường hợp có nhiều cột dữ liệu khai báo sau mệnh đề GROUP BY, bạn có thể tìm thấy kết quả sẽ được nhóm theo từng nhóm dựa trên cột dữ liệu khai báo sau mệnh đề GROUP BY.

Chẳng hạn, đối với trường hợp này chúng ta sử dụng mệnh đề GROUP BY với hai cột InvoiceNo và ProductID như ví dụ 6-27.

Ví dụ 6-27: Khai báo sử dụng phép toán ROLLUP

```

SELECT InvoiceNo, ProductID,
COUNT(Quantity) TotalQuantity,
AVG(Price) AvgOfPrice,
SUM(Discount) AS TotalDiscount,
SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
FROM SalesInvoiceDetails
GROUP BY InvoiceNo, ProductID WITH ROLLUP
ORDER BY TotalAmount Desc
GO

```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày với mẫu tin đầu tiên là kết quả áp dụng mệnh đề GROUP BY trên 41 mẫu tin còn lại đã áp dụng GROUP BY, trong mỗi nhóm dựa trên cột InvoiceNo sẽ thêm mẫu tin để áp dụng mệnh đề GROUP BY cho những mẫu tin chi tiết như hình 6-27.

8.3. Sử dụng phép toán CUBE

Phép toán ROLLUP cho phép bạn thêm mẫu tin và áp dụng mệnh đề GROUP BY cho kết quả đã sử dụng mệnh đề GROUP BY của cột dữ liệu thứ nhất được khai báo sau mệnh đề GROUP BY.

Phép toán CUBE bao gồm trường hợp ROLLUP và thêm những mẫu tin tương tự như vậy cho trường hợp cột thứ hai khai báo sau mệnh đề GROUP BY.

Chẳng hạn, bạn khai báo phép toán CUBE trong phát biểu SELECT có dùng mệnh đề GROUP BY với hai cột dữ liệu là InvoiceNo và ProductID như ví dụ 6-28.

M® 200 **Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu**

	InvoiceNo	ProductID	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
22	S10000008	P00006	1	14500.000000	30000	528250.000000
23	S10000008	NULL	2	13500.000000	60000	979500.000000
24	S10000009	P00001	1	12500.000000	0	206250.000000
25	S10000009	P00003	1	12500.000000	15000	466250.000000
26	S10000009	P00004	1	12500.000000	15000	53750.000000
27	S10000009	NULL	3	12500.000000	30000	726250.000000
28	S10000010	P00002	1	12500.000000	30000	451250.000000
29	S10000010	P00003	1	12500.000000	30000	451250.000000
30	S10000010	P00004	2	12250.000000	30000	517250.000000
31	S10000010	NULL	4	12375.000000	90000	1419750.000000
32	S10000011	P00002	1	10500.000000	0	288750.000000
33	S10000011	P00006	1	12500.000000	0	343750.000000
34	S10000011	NULL	2	11500.000000	0	632500.000000
35	S10000012	P00002	1	10500.000000	30000	258750.000000
36	S10000012	P00004	1	12500.000000	30000	38750.000000
37	S10000012	NULL	2	11500.000000	60000	297500.000000
38	S10000013	P00004	2	12000.000000	30000	51400.000000
39	S10000013	P00005	1	12500.000000	30000	38750.000000
40	S10000013	P00006	1	12500.000000	30000	38750.000000
41	S10000013	NULL	4	12250.000000	90000	128900.000000
42	NULL	NULL	30	12133.333333	805000	29067150.000...

Query execu... [nganuef (3.0 SP2)] [NGANUEF\ngandt (52)] [AccountSystem] [00:00:02] [42 rows]

Hình 6-27: Sử dụng phép toán ROLLUP.**Ví dụ 6-28: Khai báo sử dụng phép toán CUBE**

```
SELECT InvoiceNo, ProductID,
COUNT(Quantity) TotalQuantity,
AVG(Price) AvgOfPrice,
SUM(Discount) AS TotalDiscount,
SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
FROM SalesInvoiceDetails
GROUP BY InvoiceNo, ProductID WITH CUBE
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày bao gồm một số mẫu tin thêm vào là kết quả áp dụng mệnh đề GROUP BY với phép toán ROLLUP dựa trên cột InvoiceNo, bên dưới bạn có thể tìm thấy một số

Chương 6: Phát biểu T-SQL cơ bản đang truy vấn dữ liệu

201 M®

mẫu tin được áp dụng mệnh đề GROUP BY cho những mẫu tin chi tiết dựa trên cột ProductID như hình 6-28.

	InvoiceNo	ProductID	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
28	S100000010	P00002	1	12500.000000	30000	451250.000000
29	S100000010	P00003	1	12500.000000	30000	451250.000000
30	S100000010	P00004	2	12250.000000	30000	517250.000000
31	S100000010	NULL	4	12375.000000	90000	1419750.000000
32	S100000011	P00002	1	10500.000000	0	288750.000000
33	S100000011	P00006	1	12500.000000	0	343750.000000
34	S100000011	NULL	2	11500.000000	0	632500.000000
35	S100000012	P00002	1	10500.000000	30000	258750.000000
36	S100000012	P00004	1	12500.000000	30000	38750.000000
37	S100000012	NULL	2	11500.000000	60000	297500.000000
38	S100000013	P00004	2	12000.000000	30000	51400.000000
39	S100000013	P00005	1	12500.000000	30000	38750.000000
40	S100000013	P00006	1	12500.000000	30000	38750.000000
41	S100000013	NULL	4	12250.000000	90000	128900.000000
42	NULL	NULL	30	12133.333333	805000	29067150.000000
43	NULL	P00001	4	13125.000000	100000	7806250.000000
44	NULL	P00002	7	10857.142857	160000	8667500.000000
45	NULL	P00003	6	11666.666666	280000	7213750.000000
46	NULL	P00004	7	12214.285714	105000	2245150.000000
47	NULL	P00005	2	13000.000000	90000	1835000.000000
48	NULL	P00006	4	13500.000000	70000	1299500.000000

Hình 6-28: Sử dụng phép toán CUBE.

8.4. Sử dụng hàm GROUPING

Chúng ta vừa tìm hiểu hai phép toán ROLLUP và CUBE ở các ví dụ trên trong mêm đề GROUP BY.

Hàm GROUPING cho phép bạn thêm cột dữ liệu và có giá trị là 1 ứng với cột trong mẫu tin thêm mới vào tập dữ liệu trả về ứng với phép toán CUBE hay ROLLUP.

Giả sử, bạn sử dụng mệnh đề GROUP BY để liệt kê danh sách mã sản phẩm ứng với ba hàm COUNT, AVG và SUM với phép toán ROLLUP như ví dụ 6-29.

Ví dụ 6-29: Khai báo sử dụng mảng để GROUP BY

```
SELECT ProductID,  
COUNT(Quantity) TotalQuantity,
```

M® 202**Chương 6:** Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

```

AVG(Price) AvgOfPrice,
SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
FROM SalesInvoiceDetails
GROUP BY ProductID WITH ROLLUP
ORDER BY TotalAmount DESC
GO

```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-29.

	ProductID	TotalQuantity	AvgOfPrice	TotalAmount
1	NULL	30	12133.333333	29067150.000000
2	P00002	7	10857.142857	8667500.000000
3	P00001	4	13125.000000	7806250.000000
4	P00003	6	11666.666666	7213750.000000
5	P00004	7	12214.285714	2245150.000000
6	P00005	2	13000.000000	1835000.000000
7	P00006	4	13500.000000	1299500.000000

Hình 6-29: Sử dụng hàm COUNT, SUM và AVG.

Bạn cũng có thể sử dụng hàm GROUPING trong phát biểu SELECT với mệnh đề GROUP BY và phép toán ROLLUP như ví dụ 6-30.

Ví dụ 6-30: Khai báo hàm GROUPING

```

SELECT ProductID,
Grouping(ProductID) As 'GroupingOfProductID',
COUNT(Quantity) TotalQuantity,
AVG(Price) AvgOfPrice,
SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
FROM SalesInvoiceDetails
GROUP BY ProductID WITH ROLLUP
ORDER BY TotalAmount DESC
GO

```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-30.

Trong trường hợp có N cột dữ liệu khai báo sau mệnh đề GROUP BY, nếu bạn sử dụng N hàm GROUPING thì kết quả trình bày sẽ có N cột dữ liệu thêm vào.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

203

	ProductID	GroupingOfProductID	TotalQuantity	AvgOfPrice	TotalAmount
1	NULL	1	30	12133.333333	29067150.000000
2	P00002	0	7	10857.142857	8667500.000000
3	P00001	0	4	13125.000000	7806250.000000
4	P00003	0	6	11666.666666	7213750.000000
5	P00004	0	7	12214.285714	2245150.000000
6	P00005	0	2	13000.000000	1835000.000000
7	P00006	0	4	13500.000000	1299500.000000

Hình 6-30: Sử dụng hàm GROUPING.

Chẳng hạn, bạn khai báo phát biểu SELECT với mệnh đề GROUP BY và phép toán ROLLUP cùng với hàm GROUPING như ví dụ 6-31.

Ví dụ 6-31: Khai báo nhiều cột sau mệnh đề GROUP BY

```
SELECT InvoiceNo, ProductID, Grouping(InvoiceNo) As 'GroupOfInvoiceNo',
Grouping(ProductID) As 'GroupOfProductID',
COUNT(Quantity) TotalQuantity,
AVG(Price) AvgOfPrice,
SUM(Discount) AS TotalDiscount,
SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
TotalAmount
FROM SalesInvoiceDetails
GROUP BY InvoiceNo , ProductID WITH ROLLUP
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-31.

	InvoiceNo	ProductID	GroupOfInvoiceNo	GroupOfProductID	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
1	S10000001	P00001	0	0	1	10000.000000	50000	1050000.000000
2	S10000001	P00002	0	0	1	10000.000000	50000	2150000.000000
3	S10000001	NULL	0	1	2	10000.000000	100000	3200000.000000
4	S10000002	P00002	0	0	1	10000.000000	0	2200000.000000
5	S10000002	P00003	0	0	1	10000.000000	100000	1000000.000000
6	S10000002	NULL	0	1	2	10000.000000	100000	3200000.000000
7	S10000003	P00002	0	0	1	10000.000000	50000	1050000.000000
8	S10000003	P00003	0	0	1	10000.000000	50000	3250000.000000
9	S10000003	NULL	0	1	2	10000.000000	100000	4300000.000000
10	S10000004	P00001	0	0	1	15000.000000	0	2475000.000000
11	S10000004	P00004	0	0	1	12000.000000	0	1584000.000000
12	S10000004	NULL	0	1	2	13500.000000	0	4059000.000000
13	S10000005	P00001	0	0	1	15000.000000	50000	4075000.000000
14	S10000005	P00003	0	0	1	12500.000000	55000	1595000.000000
15	S10000005	NULL	0	1	2	13750.000000	105000	5670000.000000
16	S10000006	P00002	0	0	1	12500.000000	0	2268750.000000
17	S10000006	NULL	0	1	1	12500.000000	0	2268750.000000
18	S10000007	P00005	0	0	1	13500.000000	60000	1796250.000000
19	S10000007	P00006	0	0	1	14500.000000	10000	388750.000000
20	S10000007	NULL	0	1	2	14000.000000	70000	2165000.000000
21	S10000008	P00003	0	0	1	12500.000000	30000	451250.000000
22	S10000008	P00006	0	0	1	14500.000000	30000	528250.000000
23	S10000008	NULL	0	1	2	13500.000000	60000	979500.000000
24	S10000009	P00001	0	0	1	12500.000000	0	205250.000000

Hình 6-31: Sử dụng nhiều hàm GROUPING.

M® 204**Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu****8.5. Phát biểu SELECT với mệnh đề HAVING**

Tương tự như chức năng của mệnh đề WHERE, nhưng mệnh đề HAVING cho phép bạn trích lọc dữ liệu khi sử dụng mệnh đề GROUP BY.

Trong khi mệnh đề WHERE khai báo sử dụng trước mệnh đề GROUP BY thì mệnh đề HAVING khai báo sau mệnh đề GROUP BY.

Nếu lọc dữ liệu theo cột không khai báo sau mệnh đề GROUP BY thì bạn sử dụng các hàm tương ứng. Chẳng hạn, chúng ta khai báo mệnh đề HAVING như ví dụ 6-32.

Ví dụ 6-32: Khai báo mệnh đề HAVING

```
SELECT InvoiceNo,
       COUNT(Quantity) TotalQuantity,
       AVG(Price) AvgOfPrice,
       SUM(Discount) AS TotalDiscount,
       SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
       TotalAmount
  FROM SalesInvoiceDetails
 GROUP BY InvoiceNo
 HAVING SUM(Discount) >0
 GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-32.

	InvoiceNo	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
1	SI00000001	2	10000.000000	100000	3200000.000000
2	SI00000002	2	10000.000000	100000	3200000.000000
3	SI00000003	2	10000.000000	100000	4300000.000000
4	SI00000005	2	13750.000000	105000	5670000.000000
5	SI00000007	2	14000.000000	70000	2185000.000000
6	SI00000008	2	13500.000000	60000	979500.000000
7	SI00000009	3	12500.000000	30000	726250.000000
8	SI00000010	4	12375.000000	90000	1419750.000000
9	SI00000012	2	11500.000000	60000	297500.000000
10	SI00000013	4	12250.000000	90000	128900.000000

Hình 6-32: Sử dụng mệnh đề HAVING.

Tuy nhiên, bạn cũng có thể khai báo mệnh đề HAVING với cột dữ liệu khai báo sau mệnh đề GROUP BY như ví dụ 6-33.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

205

Ví dụ 6-33: Khai báo mệnh đề HAVING

```
SELECT InvoiceNo,
       COUNT(Quantity) TotalQuantity,
       AVG(Price) AvgOfPrice,
       SUM(Discount) AS TotalDiscount,
       SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
       TotalAmount
  FROM SalesInvoiceDetails
 GROUP BY InvoiceNo
 HAVING InvoiceNo like 'SI%'
 GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-33.

	InvoiceNo	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
1	SI00000001	2	10000.000000	100000	3200000.000000
2	SI00000002	2	10000.000000	100000	3200000.000000
3	SI00000003	2	10000.000000	100000	4300000.000000
4	SI00000004	2	13500.000000	0	4059000.000000
5	SI00000005	2	13750.000000	105000	5670000.000000
6	SI00000006	1	12500.000000	0	2268750.000000
7	SI00000007	2	14000.000000	70000	2185000.000000
8	SI00000008	2	13500.000000	60000	979500.000000
9	SI00000009	3	12500.000000	30000	726250.000000
10	SI00000010	4	12375.000000	90000	1419750.000000
11	SI00000011	2	11500.000000	0	632500.000000
12	SI00000012	2	11500.000000	60000	297500.000000
13	SI00000013	4	12250.000000	90000	128900.000000

Hình 6-33: Sử dụng mệnh đề HAVING.

Lưu ý: Bạn cũng có thể sử dụng mệnh đề WHERE thay vì mệnh đề HAVING bằng cách khai báo như ví dụ 6-34.

Ví dụ 6-34: Khai báo mệnh đề WHERE thay HAVING

```
SELECT InvoiceNo,
       COUNT(Quantity) TotalQuantity,
       AVG(Price) AvgOfPrice,
       SUM(Discount) AS TotalDiscount,
       SUM(Quantity*Price*(1+VATRate/100)-Discount) AS
       TotalAmount
  FROM SalesInvoiceDetails
 WHERE InvoiceNo like 'SI%'
 GROUP BY InvoiceNo
```

Khi thực thi phát biểu trong ví dụ trên, kết quả sẽ trình bày như hình 6-34.

	InvoiceNo	TotalQuantity	AvgOfPrice	TotalDiscount	TotalAmount
1	SI00000001	2	10000.000000	100000	3200000.000000
2	SI00000002	2	10000.000000	100000	3200000.000000
3	SI00000003	2	10000.000000	100000	4300000.000000
4	SI00000004	2	13500.000000	0	4059000.000000
5	SI00000005	2	13750.000000	105000	5670000.000000
6	SI00000006	1	12500.000000	0	2268750.000000
7	SI00000007	2	14000.000000	70000	2185000.000000
8	SI00000008	2	13500.000000	60000	979500.000000
9	SI00000009	3	12500.000000	30000	726250.000000
10	SI00000010	4	12375.000000	90000	1419750.000000
11	SI00000011	2	11500.000000	0	1632500.000000
12	SI00000012	2	11500.000000	60000	297500.000000
13	SI00000013	4	12250.000000	90000	128900.000000

Hình 6-34: Sử dụng mệnh đề WHERE thay HAVING.

Chú ý: Kết quả trình bày trong hình 6-33 và 6-34 giống nhau, tuy nhiên nếu bạn sử dụng mệnh đề WHERE thì dữ liệu lọc trước khi áp dụng mệnh đề GROUP BY. Trong khi đó, mệnh đề HAVING sẽ trích lọc dữ liệu sau khi áp dụng GROUP BY.

9. PHÁT BIỂU SELECT VỚI MỆNH ĐỀ COMPUTE

9.1. Mệnh đề COMPUTE

Mệnh đề COMPUTE cho phép bạn sử dụng các hàm tương tự như trong mệnh đề GROUP BY là SUM, AVG, MIN, MAX, hay COUNT và tạo ra mẫu tin ứng với các hàm mà bạn khai báo.

Giả sử, bạn khai báo liệt kê danh sách mẫu tin trong bảng PurchaseInvoiceDetails như ví dụ 6-35.

Ví dụ 6-35: Khai báo liệt kê danh sách mẫu tin

```
SELECT InvoiceNo, ProductID,
       Quantity, Price, Quantity*Price*(1+VATRate/100)-Discount
AS TotalAmount
```

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

207

```
FROM PurchaseInvoiceDetails
WHERE ProductID = 'P00001'
OR ProductID = 'P00002'
GO
```

Nếu thực thi phát biểu trong ví dụ trên thì kết quả trình bày như hình 6-35.

The screenshot shows the SQL Server Management Studio interface with two tabs: 'Results' and 'Messages'. The 'Results' tab is selected and displays a table with 12 rows of data. The columns are labeled: InvoiceNo, ProductID, Quantity, Price, and TotalAmount. The data is as follows:

InvoiceNo	ProductID	Quantity	Price	TotalAmount
1	P100000001	100	8000	880000.000000
2	P100000002	200	8500	1870000.000000
3	P100000004	150	9000	1485000.000000
4	P100000005	150	9000	1485000.000000
5	P100000006	100	9500	1045000.000000
6	P100000008	350	9500	3657500.000000
7	P100000009	350	9800	3773000.000000
8	P100000001	300	8000	2640000.000000
9	P100000002	250	8500	2337500.000000
10	P100000005	50	9000	495000.000000
11	P100000007	350	9500	3657500.000000
12	P100000008	350	10...	3850000.000000

Hình 6-35: Danh sách sản phẩm mua.

Để sử dụng mệnh đề COMPUTE, bạn khai báo như ví dụ 6-36.

Ví dụ 6-36: Khai báo sử dụng mệnh đề COMPUTE

```
SELECT InvoiceNo, ProductID,
Quantity, Price,
Quantity*Price*(1+VATRate/100)-Discount AS TotalAmount
FROM PurchaseInvoiceDetails
WHERE ProductID = 'P00001'
OR ProductID = 'P00002'
COMPUTE SUM(Quantity), AVG(Price),
SUM(Quantity*Price*(1+VATRate/100)-Discount)
GO
```

Khác với mệnh đề GROUP BY, bạn sử dụng mệnh đề COMPUTE thì kết quả trả về là một mẫu tin ứng với tập dữ liệu thứ hai như hình 6-36.

Chú ý: Nếu sử dụng ngôn ngữ lập trình .NET thì bạn có thể lấy ra tập dữ liệu thứ hai bằng cách sử dụng đối tượng DataSet hay DataReader.

The screenshot shows a SQL query results window. The results grid displays 11 rows of data from a table with columns: InvoiceNo, ProductID, Quantity, Price, and TotalAmount. The COMPUTE BY summary row at the bottom provides aggregate values: sum(Quantity) = 2700, avg(Price) = 9025.000000, and sum(TotalAmount) = 27175500.000000.

	InvoiceNo	ProductID	Quantity	Price	TotalAmount
1	P100000001	P00001	100	8000	8800000.000000
2	P100000002	P00001	200	8500	18700000.000000
3	P100000004	P00001	150	9000	14850000.000000
4	P100000005	P00002	150	9000	14850000.000000
5	P100000006	P00002	100	9500	10450000.000000
6	P100000008	P00001	350	9500	36575000.000000
7	P100000009	P00001	350	9800	37730000.000000
8	P100000001	P00002	300	8000	26400000.000000
9	P100000002	P00002	250	8500	23375000.000000
10	P100000005	P00002	50	9000	4950000.000000
11	P100000007	P00001	350	9500	36575000.000000

	sum	avg	sum
1	2700	9025.000000	27175500.000000

Hình 6-36: Sử dụng mệnh đề COMPUTE.

9.2. Phát biểu SELECT với mệnh đề COMPUTE BY

Tương tự như mệnh đề COMPUTE thêm mẫu tin và tính giá trị dựa trên các hàm khai báo sau mệnh đề này cho tập dữ liệu trả về bằng phát biểu SELECT, bạn sử dụng mệnh đề COMPUTE BY để chỉ định cột không khai báo sau mệnh đề COMPUTE trong mệnh đề BY.

Chú ý: Đặc điểm của BY là cột data được sắp xếp thứ tự trong ORDER BY

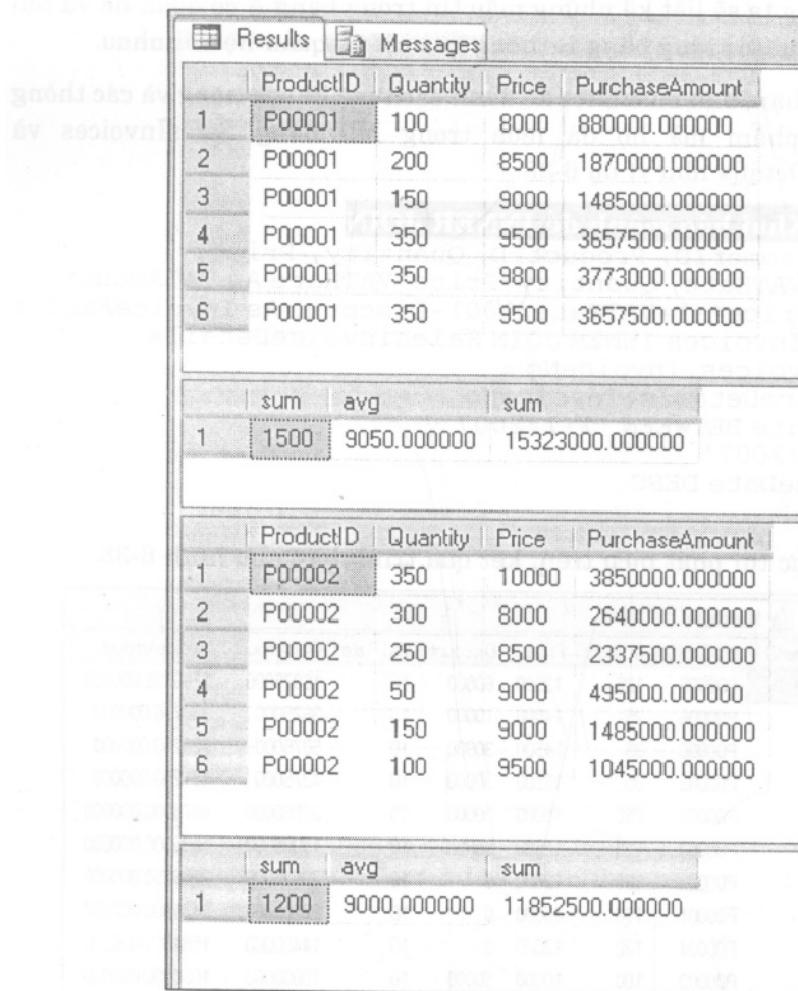
Chẳng hạn, viết lại ví dụ 6-36 thành ví dụ 6-37 có sử dụng mệnh đề COMPUTE BY như sau:

Ví dụ 6-37: Khai báo mệnh đề COMPUTE BY

```
SELECT ProductID, Quantity, Price,
Quantity*Price*(1+VATRate/100)-Discount AS
PurchaseAmount
FROM PurchaseInvoiceDetails
WHERE ProductID = 'P00001'
OR ProductID = 'P00002'
ORDER BY ProductID
COMPUTE SUM(Quantity), AVG(Price),
SUM(Quantity*Price*(1+VATRate/100)-Discount)
BY ProductID
GO
```

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu209 

Tương tự như mệnh đề COMPUTE, khi sử dụng mệnh đề COMPUTE BY thì kết quả trả về là một mẫu tin ứng với tập dữ liệu thứ hai như hình 6-37.



The screenshot shows the SQL Server Management Studio Results pane with two result sets. The first result set is for ProductID P00001, containing 6 rows of data and a summary row at the bottom. The second result set is for ProductID P00002, also containing 6 rows of data and a summary row at the bottom. The columns for both sets are ProductID, Quantity, Price, and PurchaseAmount.

	ProductID	Quantity	Price	PurchaseAmount
1	P00001	100	8000	880000.000000
2	P00001	200	8500	1870000.000000
3	P00001	150	9000	1485000.000000
4	P00001	350	9500	3657500.000000
5	P00001	350	9800	3773000.000000
6	P00001	350	9500	3657500.000000

	sum	avg	sum
1	1500	9050.000000	15323000.000000

	ProductID	Quantity	Price	PurchaseAmount
1	P00002	350	10000	3850000.000000
2	P00002	300	8000	2640000.000000
3	P00002	250	8500	2337500.000000
4	P00002	50	9000	495000.000000
5	P00002	150	9000	1485000.000000
6	P00002	100	9500	1045000.000000

	sum	avg	sum
1	1200	9000.000000	11852500.000000

Hình 6-37: Sử dụng mệnh đề COMPUTE BY.

10. PHÁT BIỂU SELECT VỚI MỆNH ĐỀ JOIN

Trong những phần trình bày ở trên, chúng ta chỉ quan tâm đến dữ liệu trong một bảng dữ liệu, trong phần này chúng ta tiếp tục tìm hiểu cách truy vấn dữ liệu trên nhiều bảng dữ liệu có quan hệ với nhau.

10.1. Mệnh đề INNER JOIN

Mệnh đề INNER JOIN được biết đến như mệnh đề liên kết bằng, có nghĩa là chúng ta sẽ liệt kê những mẫu tin trong bảng A có quan hệ và tồn tại những mẫu tin trong bảng B thông qua cột có quan hệ với nhau.

Chẳng hạn, bạn muốn liệt kê danh sách mã khách hàng và các thông tin về sản phẩm mà họ đã mua trong hai bảng SalesInvoices và SalesInvoiceDetails như ví dụ 6-38.

Ví dụ 6-38: Khai báo mệnh đề INNER JOIN

```
SELECT CustomerID, ProductID, Quantity, Price,
Discount, VATRate, Quantity*Price*VATRate As VATAmount,
Quantity*Price*(1+VATRate/100)-Discount As InvoiceAmount
FROM SalesInvoices INNER JOIN SalesInvoiceDetails
ON SalesInvoices.InvoiceNo =
SalesInvoiceDetails.InvoiceNo
WHERE DueDate BETWEEN '1/1/2007'
AND '10/17/2007'
ORDER BY DueDate DESC
GO
```

Khi thực thi phát biểu trên, kết quả trình bày như hình 6-38.

	CustomerID	ProductID	Quantity	Price	Discount	VATRate	VATAmount	InvoiceAmount
1	A0006	P00005	125	13500	60000	10	16875000	1796250.000000
2	A0006	P00006	25	14500	10000	10	3625000	388750.000000
3	A0007	P00006	35	14500	30000	10	5075000	528250.000000
4	A0007	P00003	35	12500	30000	10	4375000	451250.000000
5	A0004	P00001	250	15000	50000	10	37500000	4075000.000000
6	A0004	P00003	120	12500	55000	10	15000000	1595000.000000
7	A0005	P00002	165	12500	0	10	20625000	2268750.000000
8	A0001	P00001	150	15000	0	10	22500000	2475000.000000
9	A0001	P00004	120	12000	0	10	14400000	1584000.000000
10	A0003	P00002	100	10000	50000	10	10000000	1050000.000000
11	A0003	P00003	300	10000	50000	10	30000000	3250000.000000
12	A0002	P00003	100	10000	100000	10	10000000	1000000.000000
13	A0002	P00002	200	10000	0	10	20000000	2200000.000000
14	A0001	P00001	100	10000	50000	10	10000000	1050000.000000
15	A0001	P00002	200	10000	50000	10	20000000	2150000.000000

Hình 6-38: Sử dụng mệnh đề INNER JOIN.

Trong trường hợp nhiều bảng dữ liệu, bạn có thể khai báo tương tự như ví dụ 6-39.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

211

Ví dụ 6-39: Khai báo mệnh đề INNER JOIN

```

SELECT
    CompanyNameInVietnamese, ProductID, Quantity,
    Price, Discount, VATRate,
    Quantity*Price*VATRate As VATAmount,
    Quantity*Price*(1+VATRate/100)-Discount As
    InvoiceAmount
FROM SalesInvoices INNER JOIN SalesInvoiceDetails
ON SalesInvoices.InvoiceNo =
SalesInvoiceDetails.InvoiceNo
INNER JOIN Customers
ON Customers.CustomerID = SalesInvoices.CustomerID
WHERE SalesInvoices.DueDate BETWEEN '1/1/2007'
    AND '10/17/2007'
ORDER BY SalesInvoices.DueDate DESC
GO

```

Khi thực thi phát biểu trên, kết quả trình bày tương tự như hình 6-39.

CompanyNameInVietnamese	ProductID	Quantity	Price	Discount	VATRate	VATAmount	InvoiceAmount
Tập đoàn UCL USA	P00005	125	13500	60000	10	16875000	1796250.000000
Tập đoàn UCL USA	P00006	25	14500	10000	10	3625000	388750.000000
Công ty Đa quốc gia UFC	P00006	35	14500	30000	10	5075000	528250.000000
Công ty Đa quốc gia UFC	P00003	35	12500	30000	10	4375000	451250.000000
Công ty Trách Nhiệm Hỗn Hợp E-Google Vietnam	P00001	250	15000	50000	10	37500000	4075000.000000
Công ty Trách Nhiệm Hỗn Hợp E-Google Vietnam	P00003	120	12500	55000	10	15000000	1595000.000000
Công ty Cổ phần Suzuki Vietnam	P00002	165	12500	0	10	20625000	2268750.000000
Công ty Trách Nhiệm Hỗn Hợp Microsoft Vietnam	P00001	150	15000	0	10	22500000	2475000.000000
Công ty Trách Nhiệm Hỗn Hợp Microsoft Vietnam	P00004	120	12000	0	10	14400000	1584000.000000
Công ty Trách Nhiệm Hỗn Hợp Kodaka Vietnam	P00002	100	10000	50000	10	10000000	1050000.000000
Công ty Trách Nhiệm Hỗn Hợp Kodaka Vietnam	P00003	300	10000	50000	10	30000000	3250000.000000
Công ty Trách Nhiệm Hỗn Hợp IBN Vietnam	P00003	100	10000	100000	10	10000000	1000000.000000
Công ty Trách Nhiệm Hỗn Hợp IBN Vietnam	P00002	200	10000	0	10	20000000	2200000.000000
Công ty Trách Nhiệm Hỗn Hợp Macsoft Vietnam	P00001	100	10000	50000	10	10000000	1050000.000000
Công ty Trách Nhiệm Hỗn Hợp Macsoft Vietnam	P00002	200	10000	50000	10	20000000	2150000.000000

Hình 6-39: Sử dụng mệnh đề INNER JOIN.

Bạn có thể sử dụng chuỗi ký tự để khai báo bí danh cho bảng dữ liệu thay vì sử dụng tên bảng dữ liệu.

Chẳng hạn, bạn có thể khai báo ví dụ trên bằng cách đặt bí danh cho bảng dữ liệu như ví dụ 6-40.

Ví dụ 6-40: Khai báo bí danh cho bảng dữ liệu

```

SELECT CompanyNameInVietnamese,
ProductID, Quantity, Price
FROM SalesInvoices S INNER JOIN SalesInvoiceDetails D
ON S.InvoiceNo = D.InvoiceNo
INNER JOIN Customers C

```

M® 212

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

```

ON C.CustomerID = S.CustomerID
WHERE S.DueDate BETWEEN '1/1/2007'
AND '12/31/2007'
ORDER BY S.DueDate DESC
GO

```

Khi thực thi phát biểu trên, kết quả trình bày tương tự như hình 6-40.

	CompanyNameInVietnamese	ProductID	Quantity	Price
1	Công ty Trách Nhiệm Hữu Hạn Macromsoft Vietnam	P00002	25	10500
2	Công ty Trách Nhiệm Hữu Hạn Macromsoft Vietnam	P00004	5	12500
3	Công ty Cổ phần Suzumi Vietnam	P00004	5	12500
4	Công ty Cổ phần Suzumi Vietnam	P00005	5	12500
5	Công ty Cổ phần Suzumi Vietnam	P00006	5	12500
6	Công ty Cổ phần Suzumi Vietnam	P00004	1	11500
7	Công ty Trách Nhiệm Hữu Hạn Macromsoft Vietnam	P00002	35	12500
8	Công ty Trách Nhiệm Hữu Hạn Macromsoft Vietnam	P00003	35	12500
9	Công ty Trách Nhiệm Hữu Hạn Macromsoft Vietnam	P00004	35	12500
10	Công ty Trách Nhiệm Hữu Hạn Macromsoft Vietnam	P00004	5	12000
11	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam	P00002	25	10500
12	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam	P00006	25	12500
13	Công ty Cổ phần ReruitVietnam	P00001	15	12500
14	Công ty Cổ phần ReruitVietnam	P00003	35	12500
15	Công ty Cổ phần ReruitVietnam	P00004	5	12500
16	Tập đoàn UCIA USA	P00005	125	13500
17	Tập đoàn UCIA USA	P00006	25	14500
18	Công ty Đa quốc gia UFCA	P00006	35	14500
19	Công ty Đa quốc gia UFCA	P00003	35	12500
20	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	P00001	250	15000
21	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	P00003	120	12500
22	Công ty Cổ phần Suzumi Vietnam	P00002	165	12500
23	Công ty Trách Nhiệm Hữu Hạn Macromsoft Vietnam	P00001	150	15000

Hình 6-40: Bí danh bảng dữ liệu.

Chú ý: Mệnh đề WHERE cho phép trích lọc dữ liệu sau khi kết hợp giữa ba bảng dữ liệu.

Ngoài ra, bạn có thể sử dụng khóa AS để khai báo bí danh của bảng dữ liệu như đã sử dụng từ khóa này để đặt tên cho cột dữ liệu.

Chẳng hạn, bạn sử dụng từ khóa AS để khai báo bí danh cho bảng dữ liệu trong phát biểu SELECT với mệnh đề INNER JOIN như ví dụ 6-41.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

213

Ví dụ 6-41: Khai báo từ khóa AS

```

SELECT CompanyNameInVietnamese,
ProductID, Quantity, Price
FROM PurchaseInvoices AS P
INNER JOIN PurchaseInvoiceDetails AS D
ON P.InvoiceNo = D.InvoiceNo
INNER JOIN Suppliers AS S
ON S.SupplierID = P.SupplierID
WHERE P.DueDate BETWEEN '1/1/2007'
AND '12/31/2007'
ORDER BY P.DueDate DESC
GO

```

Khi thực thi phát biểu trên, kết quả trình bày tương tự như hình 6-41.

	CompanyNameInVietnamese	ProductID	Quantity	Price
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	P00001	350	9500
2	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	P00002	350	10000
3	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	350	9800
4	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00005	20	9500
5	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00006	50	10500
6	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00007	400	10500
7	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00002	100	9500
8	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00003	150	9500
9	Công ty Cổ phần Yamaka Việt Nam	P00004	150	10500
10	Công ty Cổ phần Yamaka Việt Nam	P00001	350	9500
11	Công ty Cổ phần Yamaka Việt Nam	P00005	250	10500
12	Công ty Cổ phần Yamaka Việt Nam	P00006	150	12500
13	Công ty Cổ phần Yamaka Việt Nam	P00007	250	8500
14	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	150	9000
15	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00004	150	10000
16	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00006	150	11500
17	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00007	250	9500
18	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	P00002	150	9000
19	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	P00002	50	9000
20	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	100	8000
21	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00002	300	8000
22	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00003	400	8000
23	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00004	100	9000
24	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00001	200	8500

Hình 6-41: Sử dụng từ khóa AS.

Chú ý: Bạn cũng có thể sử dụng mệnh đề WHERE để thay thế mệnh đề INNER JOIN bằng cách khai báo tương tự như ví dụ 6-42.

M® 214

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu**Ví dụ 6-42: Mệnh đề WHERE thay mệnh đề INNER JOIN**

```

SELECT CompanyNameInVietnamese, ProductID,
Quantity, Price, Quantity*price*(1+VATRate/100)-Discount
As PurchaseAmount
FROM PurchaseInvoices AS P,
PurchaseInvoiceDetails AS D,
Suppliers AS S
WHERE P.InvoiceNo = D.InvoiceNo
AND S.SupplierID = P.SupplierID
ORDER BY P.DueDate DESC
GO

```

Khi thực thi phát biểu trên, kết quả trình bày tương tự như hình 6-42.

	CompanyNameInVietnamese	ProductID	Quantity	Price	PurchaseAmount
1	Công ty Trách Nhiệm Hữu Hạn Macrosoft Vietnam	P00001	350	9500	3657500
2	Công ty Trách Nhiệm Hữu Hạn Macrosoft Vietnam	P00002	350	10000	3850000
3	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	350	9800	3773000
4	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00005	20	9500	209000
5	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00006	50	10500	577500
6	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00007	400	10500	4620000
7	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00002	100	9500	1045000
8	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00003	150	9500	1567500
9	Công ty Cổ phần Yamaka Việt Nam	P00004	150	10500	1732500
10	Công ty Cổ phần Yamaka Việt Nam	P00001	350	9500	3657500
11	Công ty Cổ phần Yamaka Việt Nam	P00005	250	10500	2887500
12	Công ty Cổ phần Yamaka Việt Nam	P00006	150	12500	2062500
13	Công ty Cổ phần Yamaka Việt Nam	P00007	250	8500	2337500
14	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	150	9000	1485000
15	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00004	150	10000	1650000
16	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00006	150	11500	1897500
17	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00007	250	9500	2612500
18	Công ty Trách Nhiệm Hữu Hạn Macrosoft Vietnam	P00002	150	9000	1485000
19	Công ty Trách Nhiệm Hữu Hạn Macrosoft Vietnam	P00002	50	9000	495000
20	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	100	8000	880000
21	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00002	300	8000	2640000
22	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00003	400	8000	3520000
23	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00004	100	9000	990000
24	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00001	200	8500	1870000
25	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00002	250	8500	2337500

Hình 6-42: Sử dụng mệnh đề WHERE thay INNER JOIN.

Bạn có thể trích lọc dữ liệu trong khi sử dụng mệnh đề INNER JOIN bằng cách sử dụng phép toán AND hay OR.

Chẳng hạn, để liệt kê danh sách nhà cung cấp và sản phẩm của họ đã cung cấp cho công ty có số lượng lớn hơn 100 bằng cách khai báo phép toán AND như ví dụ 6-43.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

215

Ví dụ 6-43: Phép toán AND và mệnh đề INNER JOIN

```

SELECT CompanyNameInVietnamese,
       ProductID, Quantity, Price
  FROM PurchaseInvoices AS P
 INNER JOIN PurchaseInvoiceDetails AS D
    ON P.InvoiceNo = D.InvoiceNo
 INNER JOIN Suppliers AS S
    ON S.SupplierID = P.SupplierID
   AND Quantity > 100
 WHERE P.DueDate BETWEEN '1/1/2007'
   AND '12/31/2007'
 ORDER BY P.DueDate DESC
 GO
  
```

Khi thực thi phát biểu trên, kết quả trình bày tương tự như hình 6-43.

	CompanyNameInVietnamese	ProductID	Quantity	Price
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	P00001	350	9500
2	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	P00002	350	10000
3	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	350	9800
4	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00007	400	10500
5	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00003	150	9500
6	Công ty Cổ phần Yamaka Việt Nam	P00004	150	10500
7	Công ty Cổ phần Yamaka Việt Nam	P00001	350	9500
8	Công ty Cổ phần Yamaka Việt Nam	P00005	250	10500
9	Công ty Cổ phần Yamaka Việt Nam	P00006	150	12500
10	Công ty Cổ phần Yamaka Việt Nam	P00007	250	8500
11	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00001	150	9000
12	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00004	150	10000
13	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00006	150	11500
14	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00007	250	9500
15	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	P00002	150	9000
16	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00002	300	8000
17	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	P00003	400	8000
18	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00001	200	8500
19	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00002	250	8500
20	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	P00003	150	8500

Hình 6-43: Phép toán AND và mệnh đề INNER JOIN.

10.2. Mệnh đề LEFT JOIN

Như đã giới thiệu trong phần trình bày mệnh đề INNER JOIN, bạn có thể trình bày danh sách khách hàng có mua hàng.

M® 216

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

Giả sử, bạn liệt kê danh sách khách hàng bằng cách khai báo phát biểu SELECT như ví dụ 6-44.

Ví dụ 6-44: Khai báo liệt kê danh sách khách hàng

```
SELECT CustomerID, CompanyNameInVietnamese
FROM Customers
GO
```

Khi thực thi phát biểu trong ví dụ trên, bạn có thể tìm thấy danh sách khách hàng như hình 6-44.

	CustomerID	CompanyNameInVietnamese
1	A0001	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam
2	A0002	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam
3	A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam
4	A0004	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam
5	A0005	Công ty Cổ phần Suzumi Vietnam
6	A0006	Tập đoàn UCIA USA
7	A0007	Công ty Đa quốc gia UFCA
8	A0008	Công ty Cổ phần ReruitVietnam
9	A0009	Trung tâm giáo dục Vietnam

Hình 6-44: Danh sách khách hàng.

Chẳng hạn, bạn sử dụng mệnh đề INNER JOIN như ví dụ 6-45.

Ví dụ 6-45: Liệt kê danh sách khách hàng có mua hàng

```
SELECT CompanyNameInVietnamese,
C.CustomerID, InvoiceNo
FROM Customers C
INNER JOIN SalesInvoices S
ON S.CustomerID = C.CustomerID
ORDER BY S.DueDate DESC
```

Thực thi phát biểu trên, bạn sẽ nhận được danh sách khách hàng có mua hàng như hình 6-45.

Tuy nhiên, bạn không thể trình bày danh sách khách hàng không tồn tại hóa đơn trong bảng SalesInvoices bằng mệnh đề INNER JOIN.

Thay vào đó, nếu có nhu cầu trình bày danh sách khách hàng có hay không tồn tại hóa đơn trong bảng SalesInvoices, bạn có thể sử dụng mệnh đề LEFT JOIN như ví dụ 6-46.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

217

	CompanyNameInVietnamese	CustomerID	InvoiceNo
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000012
2	Công ty Cổ phần Suzumi Vietnam	A0005	SI00000013
3	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000010
4	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002	SI00000011
5	Công ty Cổ phần ReruitVietnam	A0008	SI00000009
6	Tập đoàn UCIA USA	A0006	SI00000007
7	Công ty Đa quốc gia UFCA	A0007	SI00000008
8	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	A0004	SI00000005
9	Công ty Cổ phần Suzumi Vietnam	A0005	SI00000006
10	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000004
11	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	A0003	SI00000003
12	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002	SI00000002
13	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000001

Hình 6-45: Danh sách khách hàng có mua hàng.**Ví dụ 6-46: Khai báo mệnh đề LEFT JOIN**

```
SELECT CompanyNameInVietnamese,
C.CustomerID, InvoiceNo
FROM Customers C
LEFT JOIN SalesInvoices S
ON S.CustomerID = C.CustomerID
ORDER BY S.DueDate DESC
GO
```

Khác với trường hợp mệnh đề INNER JOIN, bạn có thể tìm thấy danh sách khách hàng liệt kê bằng mệnh đề LEFT JOIN như hình 6-46.

	CompanyNameInVietnamese	CustomerID	InvoiceNo
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000012
2	Công ty Cổ phần Suzumi Vietnam	A0005	SI00000013
3	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000010
4	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002	SI00000011
5	Công ty Cổ phần ReruitVietnam	A0008	SI00000009
6	Tập đoàn UCIA USA	A0006	SI00000007
7	Công ty Đa quốc gia UFCA	A0007	SI00000008
8	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	A0004	SI00000005
9	Công ty Cổ phần Suzumi Vietnam	A0005	SI00000006
10	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000004
11	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	A0003	SI00000003
12	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002	SI00000002
13	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000001
14	Trung tâm giáo dục Vietnam	A0009	NULL

Hình 6-46: Danh sách khách hàng.

M® 218

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

Trong đó, 1 khách hàng cuối cùng không có hóa đơn trong bảng SalesInvoices.

Chú ý: Bạn có thể thay thế NULL bằng giá trị khác bằng cách sử dụng hàm ISNULL như ví dụ 6-47.

Ví dụ 6-47: Khai báo hàm ISNULL

```
SELECT CompanyNameInVietnamese, C.CustomerID,
ISNULL(InvoiceNo, '') As InvoiceNo
FROM Customers C
LEFT JOIN SalesInvoices S
ON S.CustomerID = C.CustomerID
ORDER BY S.DueDate DESC
GO
```

Khác với kết quả trình bày trong ví dụ 6-46, bạn có thể tìm thấy những khách hàng liệt kê không có hóa đơn xuất hiện như hình 6-47.

	CompanyNameInVietnamese	CustomerID	InvoiceNo
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000012
2	Công ty Cổ phần Suzumi Vietnam	A0005	SI00000013
3	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000010
4	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002	SI00000011
5	Công ty Cổ phần ReruitVietnam	A0008	SI00000009
6	Tập đoàn UCIA USA	A0006	SI00000007
7	Công ty Đa quốc gia UFCAC	A0007	SI00000008
8	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	A0004	SI00000005
9	Công ty Cổ phần Suzumi Vietnam	A0005	SI00000006
10	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000004
11	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	A0003	SI00000003
12	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002	SI00000002
13	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	SI00000001
14	Trung tâm giáo dục Vietnam	A0009	

Hình 6-47: Sử dụng hàm ISNULL.

10.3. Mệnh đề RIGHT JOIN

Ngược lại mệnh đề LEFT JOIN là mệnh đề RIGHT JOIN, nếu bạn đổi vị trí của bảng dữ liệu thứ nhất thành bảng dữ liệu thứ hai và ngược lại thì có kết quả như nhau.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

219

Chẳng hạn, bạn khai báo mệnh đề RIGHT JOIN để liệt kê danh sách nhà cung cấp có hay không tồn tại hóa đơn bán hàng trong bảng PurchaseInvoices như ví dụ 6-48.

Ví dụ 6-48: Khai báo mệnh đề RIGHT JOIN

```
SELECT CompanyNameInVietnamese,
S.SupplierID, InvoiceNo
FROM PurchaseInvoices P
RIGHT JOIN Suppliers S
ON S.SupplierID = P.SupplierID
ORDER BY P.DueDate DESC
GO
```

Thực thi phát biểu trong ví dụ trên, bạn có thể tìm thấy danh sách nhà cung cấp như hình 6-48.

	CompanyNameInVietnamese	SupplierID	InvoiceNo
1	Công ty Trách Nhiệm Hữu Hạn Ajinomoto Việt Nam	S0001	P100000009
2	Công ty Trách Nhiệm Hữu Hạn Macrsoft Vietnam	S0004	P100000008
3	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	S0002	P100000006
4	Công ty Cổ phần Yamaka Việt Nam	S0003	P100000007
5	Công ty Cổ phần Yamaka Việt Nam	S0003	P100000003
6	Công ty Trách Nhiệm Hữu Hạn Macrsoft Vietnam	S0004	P100000005
7	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	S0001	P100000004
8	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	S0001	P100000001
9	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	S0002	P100000002
10	Công ty Trách Nhiệm Hữu Hạn Delle Vietnam	S0005	NULL

Hình 6-48: Danh sách nhà cung cấp.

Chú ý: Nếu bạn thay mệnh đề RIGHT JOIN thành mệnh đề LEFT JOIN như ví dụ 6-49 thì dữ liệu kết xuất sẽ khác đi như hình 6-49.

Ví dụ 6-49: Khai báo mệnh đề LEFT JOIN

```
SELECT CompanyNameInVietnamese,
S.SupplierID, InvoiceNo
FROM PurchaseInvoices P
LEFT JOIN Suppliers S
ON S.SupplierID = P.SupplierID
ORDER BY P.DueDate DESC
GO
```

Kết quả trình bày như hình 6-49 khi thực thi phát biểu SELECT trên.

M® 220

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

	CompanyNameInVietnamese	SupplierID	InvoiceNo
1	Công ty Trách Nhiệm Hữu Hạn Macosoft Vietnam	S0004	P100000008
2	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	S0001	P100000009
3	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	S0002	P100000006
4	Công ty Cổ phần Yamaka Việt Nam	S0003	P100000007
5	Công ty Cổ phần Yamaka Việt Nam	S0003	P100000003
6	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	S0001	P100000004
7	Công ty Trách Nhiệm Hữu Hạn Macosoft Vietnam	S0004	P100000005
8	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	S0001	P100000001
9	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	S0002	P100000002

Hình 6-49: Dùng mệnh đề LEFT JOIN thay vì RIGHT JOIN.**10.4. Mệnh đề FULL JOIN**

Mệnh đề FULL JOIN là kết hợp mệnh đề LEFT JOIN và RIGHT JOIN. Điều này có nghĩa là khi bạn sử dụng mệnh đề FULL JOIN thì kết quả trình bày bao gồm những mẫu tin trong bảng A có hay không tồn tại trong bảng B và ngược lại.

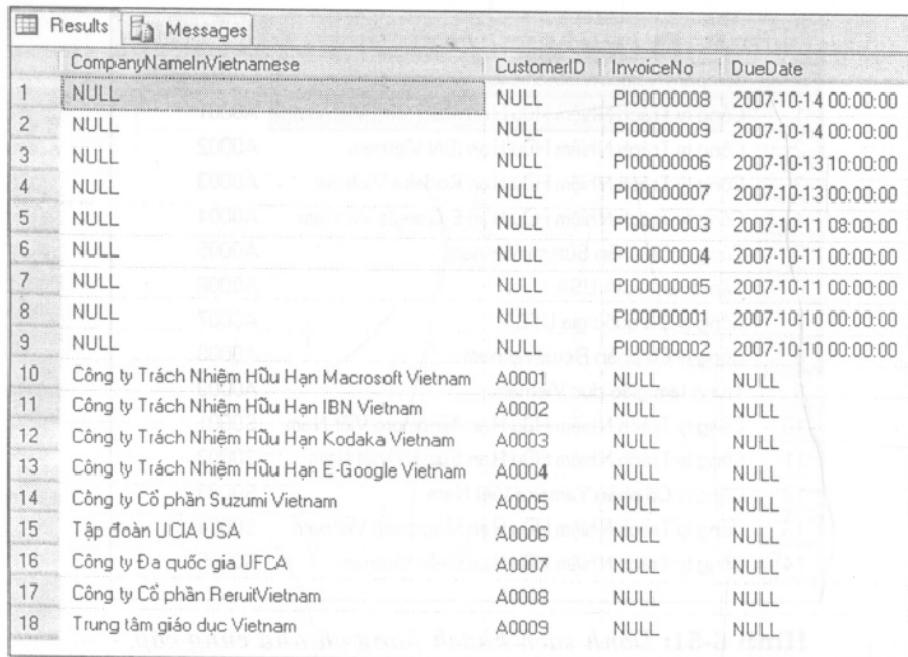
Hầu hết dữ liệu tồn tại trong cơ sở dữ liệu đều có ràng buộc và thống nhất, chính vì vậy chúng ta thường sử dụng mệnh đề INNER JOIN, LEFT JOIN hay RIGHT JOIN.

Tuy nhiên, bạn có thể tham khảo cách sử dụng mệnh đề FULL JOIN đối với hai bảng dữ liệu Customers và PurchaseInvoices như ví dụ 6-50.

Ví dụ 6-50: Khai báo mệnh đề FULL JOIN

```
SELECT CompanyNameInVietnamese,
C.CustomerID, InvoiceNo, P.DueDate
FROM PurchaseInvoices P
FULL JOIN Customers C
ON C.CustomerID = P.SupplierID
ORDER BY P.DueDate DESC
GO
```

Khi thực thi phát biểu trên, bạn có thể tìm thấy danh sách mẫu tin bao gồm những mẫu tin của cả hai bảng PurchaseInvoices và Customers như hình 6-50.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu 221 


	CompanyNameInVietnamese	CustomerID	InvoiceNo	DueDate
1	NULL	NULL	P100000008	2007-10-14 00:00:00
2	NULL	NULL	P100000009	2007-10-14 00:00:00
3	NULL	NULL	P100000006	2007-10-13 10:00:00
4	NULL	NULL	P100000007	2007-10-13 00:00:00
5	NULL	NULL	P100000003	2007-10-11 08:00:00
6	NULL	NULL	P100000004	2007-10-11 00:00:00
7	NULL	NULL	P100000005	2007-10-11 00:00:00
8	NULL	NULL	P100000001	2007-10-10 00:00:00
9	NULL	NULL	P100000002	2007-10-10 00:00:00
10	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001	NULL	NULL
11	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002	NULL	NULL
12	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	A0003	NULL	NULL
13	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	A0004	NULL	NULL
14	Công ty Cổ phần Suzumi Vietnam	A0005	NULL	NULL
15	Tập đoàn UCIA USA	A0006	NULL	NULL
16	Công ty Đa quốc gia UFCA	A0007	NULL	NULL
17	Công ty Cổ phần ReruiVietnam	A0008	NULL	NULL
18	Trung tâm giáo dục Vietnam	A0009	NULL	NULL

Hình 6-50: Sử dụng mệnh đề FULL JOIN.

11. PHÉP TOÁN UNION, EXCEPT, INTERSECT

11.1. Phát biểu SELECT với phép toán UNION

Phép toán UNION cho phép kết hợp nhiều tập dữ liệu từ nhiều phát biểu SELECT thành một nếu số cột dữ liệu bằng nhau và kiểu dữ liệu của các cột tương ứng phải tương thích.

Chẳng hạn, bạn liệt kê danh sách khách hàng và nhà cung cấp thành một tập dữ liệu bằng cách sử dụng phép toán UNION như ví dụ 6-51.

Ví dụ 6-51: Khai báo sử dụng phép toán UNION

```
SELECT CompanyNameInVietnamese, CustomerID
FROM Customers
UNION
SELECT CompanyNameInVietnamese, SupplierID
FROM Suppliers
GO
```

Khi thực thi phát biểu SELECT với phép toán UNION, bạn sẽ nhận được danh sách khách hàng và nhà cung cấp như hình 6-51.

	CompanyNameInVietnamese	CustomerID
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001
2	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002
3	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	A0003
4	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	A0004
5	Công ty Cổ phần Suzumi Vietnam	A0005
6	Tập đoàn UCIA USA	A0006
7	Công ty Đa quốc gia UFCA	A0007
8	Công ty Cổ phần ReruitVietnam	A0008
9	Trung tâm giáo dục Vietnam	A0009
10	Công ty Trách Nhiệm Hữu Hạn Ajinomoto Việt Nam	S0001
11	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	S0002
12	Công ty Cổ phần Yamaka Việt Nam	S0003
13	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	S0004
14	Công ty Trách Nhiệm Hữu Hạn Delle Vietnam	S0005

Hình 6-51: Danh sách khách hàng và nhà cung cấp.

Chú ý: Phần tựa đề của kết quả trình bày khi sử dụng phép toán UNION sẽ lấy tên cột dữ liệu của phát biểu SELECT thứ nhất.

Bạn có thể sử dụng mệnh đề ORDER BY để sắp xếp dữ liệu sau khi kết hợp bằng mệnh đề UNION. Do tựa đề được lấy ra từ phát biểu SELECT thứ nhất, nên khi khai báo mệnh đề ORDER BY, bạn phải sử dụng tựa đề của bảng dữ liệu thứ nhất như ví dụ 6-52.

Ví dụ 6-52: Khai báo sắp xếp dữ liệu

```
SELECT CompanyNameInVietnamese, CustomerID
FROM Customers
UNION
SELECT CompanyNameInVietnamese, SupplierID
FROM Suppliers
ORDER BY CustomerID DESC
GO
```

Nếu thực thi phát biểu SELECT với phép toán UNION và mệnh đề ORDER BY, bạn sẽ nhận được danh sách khách hàng và nhà cung cấp như hình 6-52.

Chú ý: Khi sử dụng phép toán UNION không có từ khóa ALL, kết quả sẽ loại bỏ những trùng lặp. Chẳng hạn, nếu bạn khai báo phát biểu SELECT với phép toán UNION như ví dụ 6-53.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

223



	CompanyNameInVietnamese	CustomerID
1	Công ty Trách Nhiệm Hữu Hạn Delle Vietnam	S0005
2	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	S0004
3	Công ty Cổ phần Yamaka Việt Nam	S0003
4	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	S0002
5	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	S0001
6	Trung tâm giáo dục Vietnam	A0009
7	Công ty Cổ phần ReruitVietnam	A0008
8	Công ty Đa quốc gia UFCA	A0007
9	Tập đoàn UCIA USA	A0006
10	Công ty Cổ phần Suzumi Vietnam	A0005
11	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	A0004
12	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	A0003
13	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	A0002
14	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	A0001

Hình 6-52: Sử dụng mệnh đề ORDER BY.**Ví dụ 6-53: Khai báo phép toán UNION**

```
SELECT CompanyNameInVietnamese
FROM Customers
UNION
SELECT CompanyNameInVietnamese
FROM Suppliers
ORDER BY CompanyNameInVietnamese DESC
GO
```

Danh sách gồm 13 nhà cung cấp và khách hàng trình bày như hình 6-53.

Tuy nhiên, thực tế trong cơ sở dữ liệu tồn tại một đối tác vừa là khách hàng vừa là nhà cung cấp, để liệt kê đầy đủ danh sách khách hàng và nhà cung cấp, bạn sử dụng từ khóa ALL như ví dụ 6-54.

Ví dụ 6-54: Khai báo từ khóa ALL

```
SELECT CompanyNameInVietnamese
FROM Customers
UNION ALL
SELECT CompanyNameInVietnamese
FROM Suppliers
ORDER BY CompanyNameInVietnamese DESC
GO
```

M® 224

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

Results		Messages
	CompanyName	Vietnamese
1	Trung tâm giáo dục Vietnam	
2	Tập đoàn UCIA USA	
3	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	
4	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	
5	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	
6	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	
7	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	
8	Công ty Trách Nhiệm Hữu Hạn Delle Vietnam	
9	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	
10	Công ty Đa quốc gia UFCA	
11	Công ty Cổ phần Yamaka Việt Nam	
12	Công ty Cổ phần Suzumi Vietnam	
13	Công ty Cổ phần ReruitVietnam	

Hình 6-53: Phép toán UNION không có từ khóa ALL.

Khi thực thi phát biểu trên, danh sách gồm 14 mẫu tin ứng với nhà cung cấp và khách hàng trình bày như hình 6-54.

Results		Messages
	CompanyName	Vietnamese
1	Trung tâm giáo dục Vietnam	
2	Tập đoàn UCIA USA	
3	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam	
4	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	
5	Công ty Trách Nhiệm Hữu Hạn Macosoft Vietnam	
6	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	
7	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	
8	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	
9	Công ty Trách Nhiệm Hữu Hạn Delle Vietnam	
10	Công ty Trách Nhiệm Hữu Hạn Ajinomoro Việt Nam	
11	Công ty Đa quốc gia UFCA	
12	Công ty Cổ phần Yamaka Việt Nam	
13	Công ty Cổ phần Suzumi Vietnam	
14	Công ty Cổ phần ReruitVietnam	

Hình 6-54: Sử dụng từ khóa ALL.

11.2. Phát biểu SELECT với phép toán EXCEPT

Trong khi phép toán UNION cho phép kết hợp nhiều tập dữ liệu từ các phát biểu SELECT lại với nhau, phép toán EXCEPT cho phép liệt kê danh sách những mẫu tin tồn tại trong bảng dữ liệu thứ nhất mà không tồn tại trong bảng thứ hai.

Giả sử, bạn liệt kê danh sách tên khách hàng trong bảng Customers và nhà cung cấp trong bảng Suppliers như ví dụ 6-55.

Ví dụ 6-55: Liệt kê danh sách khách hàng

```
SELECT CompanyNameInVietnamese
FROM Customers
SELECT CompanyNameInVietnamese
FROM Suppliers
GO
```

Khi thực thi hai phát biểu SELECT trên, kết quả sẽ trình bày như hình 6-55.

	CompanyNameInVietnamese
1	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam
2	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam
3	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam
4	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam
5	Công ty Cổ phần Suzumi Vietnam
6	Tập đoàn UCIA USA
7	Công ty Đa quốc gia UFCA
8	Công ty Cổ phần ReutriVietnam
9	Trung tâm giáo dục Vietnam

	CompanyNameInVietnamese
1	Công ty Trách Nhiệm Hữu Hạn Ajinomoto Việt Nam
2	Công ty Trách Nhiệm Hữu Hạn Suzumi Việt Nam
3	Công ty Cổ phần Yamaka Việt Nam
4	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam
5	Công ty Trách Nhiệm Hữu Hạn Delle Vietnam

Hình 6-55: Danh sách khách hàng và nhà cung cấp.

Nếu bạn muốn lấy ra danh sách tên những khách hàng không phải là nhà cung cấp thì khai báo phép toán EXCEPT như ví dụ 6-56.

Ví dụ 6-56: Khai báo phép toán EXCEPT

```
SELECT CompanyNameInVietnamese
FROM Customers
EXCEPT
SELECT CompanyNameInVietnamese
FROM Suppliers
ORDER BY CompanyNameInVietnamese DESC
GO
```

Chú ý: Trong cơ sở dữ liệu tồn tại tên công ty “Công ty Trách Nhiệm Hữu Han Microsoft Vietnam” vừa là khách hàng vừa là nhà cung cấp.

Tương tự như trên, nếu thực thi phát biểu trong ví dụ 6-56 thì kết quả trình bày như hình 6-56.

	CompanyNameVNVietnamese
1	Trung tâm giáo dục Vietnam
2	Tập đoàn UCIA USA
3	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam
4	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam
5	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam
6	Công ty Đa quốc gia UFCA
7	Công ty Cổ phần Suzumi Vietnam
8	Công ty Cổ phần ReruitVietnam

Hình 6-56: Sử dụng phép toán EXCEPT.

11.3. Phát biểu SELECT với phép toán INTERSECT

Ngược lại với trường hợp sử dụng phép toán EXCEPT, nếu bạn muốn lấy ra mẫu tin vừa tồn tại trong bảng thứ nhất vừa tồn tại trong bảng thứ hai thì dùng Intersect.

Chẳng hạn, để lấy ra danh sách tên những khách hàng cũng là nhà cung cấp thì khai báo phép toán INTERSECT như ví dụ 6-57.

Ví dụ 6-57: Khai báo phép toán INTERSECT

```
SELECT CompanyNameInVietnamese
FROM Customers
INTERSECT
SELECT CompanyNameInVietnamese
FROM Suppliers
ORDER BY CompanyNameInVietnamese DESC
GO
```

Chương 6: Phát biểu T-SQL cơ bản dang truy vấn dữ liệu

227 M®

Nếu thực thi phát biểu trong ví dụ 6-57 thì kết quả trình bày như hình 6-57.

```
MY SOLUTION.A...statement.sql* Summary
SELECT CompanyNameInVietnamese
FROM Customers
INTERSECT
SELECT CompanyNameInVietnamese
FROM Suppliers
ORDER BY CompanyNameInVietnamese DESC
GO
```

Hình 6-57: Sử dụng phép toán INTERSECT.

12. PHÂN TRANG VỚI PHÁT BIỂU WITH

Một trong những đặc điểm mới được giới thiệu trong SQL Server 2005 là cho phép bạn lấy ra mẫu tin từ vị trí thứ i đến vị trí thứ j (i:j).

Để làm điều này, trước tiên bạn sử dụng khái niệm biểu thức bảng bằng cách sử dụng phát biểu WITH với cú pháp như sau:

```
[ WITH <common_table_expression> ]
SELECT select_list [ INTO new_table ]
[ FROM table_source ] [ WHERE search_condition ]
[ GROUP BY group_by_expression ]
[ HAVING search_condition ]
[ ORDER BY order_expression [ ASC | DESC ] ]
```

Chú ý: Bạn khai báo biểu thức bảng với mệnh đề WITH có tên tùy ý và ngay sau từ khóa AS là phát biểu truy vấn SELECT

Chẳng hạn, bạn khai báo phát biểu WITH với biểu thức bảng dữ liệu có tên SuppliersAndPurchaseInvoice với phát biểu SELECT như ví dụ 6-58.

Ví dụ 6-58: Khai báo biến thức bảng

Ví dụ 3-38. Khai báo biến thực bằng WITH SuppliersAndPurchasesInvoice

W.I.
A.S.

2

```
SELECT SupplierID, ProductID, Quantity,  
Price, VATRate, Quantity*Quantity*VATRate VATAmount
```

M® 228

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

```

Quantity*Price*(1+VATRate/100)-Discount As
PurchaseAmount
FROM PurchaseInvoices AS P
INNER JOIN PurchaseInvoiceDetails AS D
ON P.InvoiceNo = D.InvoiceNo

)
SELECT * FROM SuppliersAndPurchaseInvoice
GO

```

Chú ý: Bạn có thể sử dụng phát biểu WITH để khai báo biểu thức bảng trong đối tượng VIEW hay STORED PROCEDURE.

Khi thực thi phát biểu trên, bạn có thể tìm thấy danh sách hóa đơn mua hàng trình bày như hình 6-58.

	SupplierID	ProductID	Quantity	Price	VATRate	VATAmount	PurchaseAmount
1	S0001	P00001	100	8000	10	100000	880000.000000
2	S0002	P00001	200	8500	10	400000	1870000.000000
3	S0003	P00005	250	10500	10	625000	2887500.000000
4	S0001	P00001	150	9000	10	225000	1485000.000000
5	S0004	P00002	150	9000	10	225000	1485000.000000
6	S0002	P00002	100	9500	10	100000	1045000.000000
7	S0003	P00004	150	10500	10	225000	1732500.000000
8	S0004	P00001	350	9500	10	1225000	3657500.000000
9	S0001	P00001	350	9800	10	1225000	3773000.000000
10	S0001	P00002	300	8000	10	900000	2640000.000000
11	S0002	P00002	250	8500	10	625000	2337500.000000
12	S0003	P00006	150	12500	10	225000	2062500.000000
13	S0001	P00004	150	10000	10	225000	1650000.000000
14	S0004	P00002	50	9000	10	25000	495000.000000
15	S0002	P00003	150	9500	10	225000	1567500.000000
16	S0003	P00001	350	9500	10	1225000	3657500.000000
17	S0004	P00002	350	10000	10	1225000	3850000.000000
18	S0001	P00005	20	9500	10	4000	209000.000000
19	S0001	P00003	400	8000	10	1600000	3520000.000000
20	S0002	P00003	150	8500	10	225000	1402500.000000
21	S0003	P00007	250	8500	10	625000	2337500.000000
22	S0001	P00006	150	11500	10	225000	1897500.000000
23	S0001	P00006	50	10500	10	25000	577500.000000
24	S0001	P00004	100	9000	10	100000	990000.000000
25	S0002	P00004	50	9500	10	25000	522500.000000
26	S0001	P00007	250	9500	10	625000	2612500.000000
27	S0001	P00007	400	10500	10	1600000	4620000.000000

Hình 6-58: Danh sách hóa đơn mua hàng.

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

229

Để có dữ liệu được sắp xếp, bạn sử dụng mệnh đề ORDER BY. Tuy nhiên, trong trường hợp này bạn cần kết hợp mệnh đề OVER như ví dụ 6-59.

Ví dụ 6-59: Khai báo mệnh đề OVER

```
WITH SuppliersAndPurchaseInvoice
AS
(
    SELECT P.InvoiceNo, ProductID, Quantity,
    Price, VATRate, Quantity*Quantity*VATRate VATAmount,
    Quantity*Price*(1+VATRate/100) -Discount As
    PurchaseAmount,
    ROW_NUMBER() OVER(ORDER BY P.InvoiceNo) AS 'RowNumber'
    FROM PurchaseInvoices AS P
    INNER JOIN PurchaseInvoiceDetails AS D
    ON P.InvoiceNo = D.InvoiceNo
)
SELECT * FROM SuppliersAndPurchaseInvoice
GO
```

Khi thực thi phát biểu trên, bạn sẽ nhận được kết quả trình bày được sắp xếp theo cột RowNumber như hình 6-59.

	InvoiceNo	ProductID	Quantity	Price	VATRate	VATAmount	PurchaseAmount	RowNumber
1	P100000001	P00001	100	8000	10	100000	880000.000000	1
2	P100000001	P00002	300	8000	10	900000	2640000.000000	2
3	P100000001	P00003	400	8000	10	1600000	3520000.000000	3
4	P100000001	P00004	100	9000	10	100000	990000.000000	4
5	P100000002	P00004	50	9500	10	25000	522500.000000	5
6	P100000002	P00003	150	8500	10	225000	1402500.000000	6
7	P100000002	P00002	250	8500	10	625000	2337500.000000	7
8	P100000002	P00001	200	8500	10	400000	1870000.000000	8
9	P100000003	P00005	250	10...	10	625000	2887500.000000	9
10	P100000003	P00006	150	12...	10	225000	2062500.000000	10
11	P100000003	P00007	250	8500	10	625000	2337500.000000	11
12	P100000004	P00006	150	11...	10	225000	1897500.000000	12
13	P100000004	P00007	250	9500	10	625000	2612500.000000	13
14	P100000004	P00004	150	10...	10	225000	1650000.000000	14
15	P100000004	P00001	150	9000	10	225000	1485000.000000	15
16	P100000005	P00002	150	9000	10	225000	1485000.000000	16
17	P100000005	P00002	50	9000	10	25000	495000.000000	17
18	P100000006	P00003	150	9500	10	225000	1567500.000000	18
19	P100000006	P00002	100	9500	10	100000	1045000.000000	19
20	P100000007	P00004	150	10...	10	225000	1732500.000000	20
21	P100000007	P00001	350	9500	10	1225000	3657500.000000	21
22	P100000008	P00002	350	10...	10	1225000	3850000.000000	22
23	P100000008	P00001	350	9500	10	1225000	3657500.000000	23
24	P100000009	P00001	350	9800	10	1225000	3773000.000000	24
25	P100000009	P00005	20	9500	10	4000	209000.000000	25
26	P100000009	P00007	400	10...	10	1600000	4620000.000000	26
27	P100000009	P00006	50	10...	10	25000	577500.000000	27

Hình 6-59: Khai báo mệnh đề ORDER BY và OVER.

Dựa vào cột dữ liệu RowNumber, bạn có thể lấy ra mẫu tin thứ i đến j bằng cách khai báo mệnh đề ORDER BY trong phát biểu SELECT.

Chẳng hạn, bạn khai báo sử dụng mệnh đề ORDER BY dựa trên cột RowNumber từ ví dụ 6-59 thành ví dụ như ví dụ 6-60.

Ví dụ 6-60: Sắp xếp xuất hiện dựa vào cột RowNumber

```
WITH SuppliersAndPurchaseInvoice
AS
(
    SELECT P.InvoiceNo, ProductID, Quantity,
    Price, VATRate, Quantity*Quantity*VATRate VATAmount,
    Quantity*Price*(1+VATRate/100)-Discount As
    PurchaseAmount,
    ROW_NUMBER() OVER(ORDER BY P.InvoiceNo) AS 'RowNumber'
    FROM PurchaseInvoices AS P
    INNER JOIN PurchaseInvoiceDetails AS D
    ON P.InvoiceNo = D.InvoiceNo
)
SELECT * FROM SuppliersAndPurchaseInvoice
WHERE RowNumber >= 10 and RowNumber<=20
GO
```

Khi thực thi phát biểu trên, bạn sẽ nhận được kết quả trình bày được sắp xếp theo cột RowNumber như hình 6-60.

	InvoiceNo	ProductID	Quantity	Price	VATRate	VATAmount	PurchaseAmount	RowNumber
1	P10000003	P00006	150	12500	10	225000	2062500.000000	10
2	P10000003	P00007	250	8500	10	625000	2337500.000000	11
3	P10000004	P00006	150	11500	10	225000	1897500.000000	12
4	P10000004	P00007	250	9500	10	625000	2612500.000000	13
5	P10000004	P00004	150	10000	10	225000	1650000.000000	14
6	P10000004	P00001	150	9000	10	225000	1485000.000000	15
7	P10000005	P00002	150	9000	10	225000	1485000.000000	16
8	P10000005	P00002	50	9000	10	25000	495000.000000	17
9	P10000006	P00003	150	9500	10	225000	1567500.000000	18
10	P10000006	P00002	100	9500	10	100000	1045000.000000	19
11	P10000007	P00004	150	10500	10	225000	1732500.000000	20

Hình 6-60: Trích lọc mẫu tin thứ 10 đến 20.

Bạn cũng có thể khai báo chỉ định cột dữ liệu cho phép lấy ra từ tập dữ liệu khi sử dụng phát biểu WITH như ví dụ 6-61.

Ví dụ 6-61: Khai báo cột dữ liệu

```
WITH SuppliersAndPurchaseInvoice
(
    InvoiceNo, ProductID, Quantity, Price, VATRate,
    VATAmount, PurchaseAmount, RowNumber)
```

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

231

```

AS
(
    SELECT P.InvoiceNo, ProductID, Quantity,
    Price, VATRate, Quantity*Quantity*VATRate VATAmount ,
    Quantity*Price*(1+VATRate/100)-Discount As
    PurchaseAmount,
    ROW_NUMBER() OVER(ORDER BY P.InvoiceNo) AS 'RowNumber'
    FROM PurchaseInvoices AS P
    INNER JOIN PurchaseInvoiceDetails AS D
    ON P.InvoiceNo = D.InvoiceNo
)
SELECT * FROM SuppliersAndPurchaseInvoice
WHERE RowNumber >= 10 and RowNumber<=20
GO

```

Khi thực thi phát biểu trên, bạn sẽ nhận được kết quả trình bày được sắp xếp theo cột RowNumber như hình 6-61.

	Results	Messages						
	InvoiceNo	ProductID	Quantity	Price	VATRate	VATAmount	PurchaseAmount	RowNumber
1	P100000003	P00006	150	12500	10	225000	2062500.000000	10
2	P100000003	P00007	250	8500	10	625000	2337500.000000	11
3	P100000004	P00006	150	11500	10	225000	1897500.000000	12
4	P100000004	P00007	250	9500	10	625000	2612500.000000	13
5	P100000004	P00004	150	10000	10	225000	1650000.000000	14
6	P100000004	P00001	150	9000	10	225000	1485000.000000	15
7	P100000005	P00002	150	9000	10	225000	1485000.000000	16
8	P100000005	P00002	50	9000	10	25000	495000.000000	17
9	P100000006	P00003	150	9500	10	225000	1567500.000000	18
10	P100000006	P00002	100	9500	10	100000	1045000.000000	19
11	P100000007	P00004	150	10500	10	225000	1732500.000000	20

Hình 6-61: Khai báo cột dữ liệu trình bày.

Khi lập trình ứng dụng bằng ngôn ngữ lập trình C# 2005 hay Visual Basic 2005, thay vì sử dụng phương thức Fill của đối tượng SqlDataAdapter để lấy ra mẫu tin từ thứ i đến j, thì bạn có thể sử dụng biểu thức bảng như vừa trình bày ở trên để lấy ra trang dữ liệu cần trình bày.

Bạn có thể chỉ định tên cột dữ liệu trong mệnh đề WITH có tên khác với tên cột dữ liệu trong phát biểu SQL bằng cách khai báo tương tự như ví dụ 6-62.

Ví dụ 6-62: Khai báo cột dữ liệu với tên mới

```

WITH SuppliersAndPurchaseInvoice
(Invoice, Product, Qty, UnitPrice,
VAT, TAXAmount, Amount)
AS

```

M® 232

Chương 6: Phát biểu T-SQL cơ bản dạng truy vấn dữ liệu

```

(
    SELECT P.InvoiceNo, ProductID, Quantity,
    Price, VATRate, Quantity*Quantity*VATRate VATAmount,
    Quantity*Price*(1+VATRate/100)-Discount As
    PurchaseAmount
    FROM PurchaseInvoices AS P
    INNER JOIN PurchaseInvoiceDetails AS D
    ON P.InvoiceNo = D.InvoiceNo
)
SELECT * FROM SuppliersAndPurchaseInvoice
GO

```

Trong đó, cột dữ liệu Invoice, Product, Qty, UnitPrice, VATAmount, Amount đều khác với tên cột dữ liệu khai báo trong phát biểu SELECT.

Khi thực thi phát biểu trên, kết quả sẽ trình bày tương tự như hình 6-62.

	Invoice	Product	Qty	UnitPrice	VAT	TAXAmount	Amount
1	P100000001	P00001	100	8000	10	100000	880000.000000
2	P100000002	P00001	200	8500	10	400000	1870000.000000
3	P100000003	P00005	250	10500	10	625000	2887500.000000
4	P100000004	P00001	150	9000	10	225000	1485000.000000
5	P100000005	P00002	150	9000	10	225000	1485000.000000
6	P100000006	P00002	100	9500	10	100000	1045000.000000
7	P100000007	P00004	150	10500	10	225000	1732500.000000
8	P100000008	P00001	350	9500	10	1225000	3657500.000000
9	P100000009	P00001	350	9800	10	1225000	3773000.000000
10	P100000001	P00002	300	8000	10	900000	2640000.000000
11	P100000002	P00002	250	8500	10	625000	2337500.000000
12	P100000003	P00006	150	12500	10	225000	2062500.000000
13	P100000004	P00004	150	10000	10	225000	1650000.000000
14	P100000005	P00002	50	9000	10	25000	495000.000000
15	P100000006	P00003	150	9500	10	225000	1567500.000000
16	P100000007	P00001	350	9500	10	1225000	3657500.000000
17	P100000008	P00002	350	10000	10	1225000	3850000.000000
18	P100000009	P00005	20	9500	10	4000	209000.000000
19	P100000001	P00003	400	8000	10	1600000	3520000.000000
20	P100000002	P00003	150	8500	10	225000	1402500.000000
21	P100000003	P00007	250	8500	10	625000	2337500.000000
22	P100000004	P00006	150	11500	10	225000	1897500.000000
23	P100000009	P00006	50	10500	10	25000	577500.000000

Hình 6-62: Khai báo cột dữ liệu với tên mới.

13. LẤY TẬP MẪU TIN NGẪU NHIÊN

Ngoài ra, bạn có thể lấy ra tập mẫu tin ngẫu nhiên thay vì chỉ định mẫu tin từ thứ i đến j bằng cách sử dụng hàm NEWID sau mệnh đề ORDER BY. Cách này thường được sử dụng khi làm các ứng dụng như thi trắc nghiệm.

Chẳng hạn, bạn khai báo phát biểu SELECT với mệnh đề ORDER BY và hàm NEWID để lấy 5 khách hàng một cách ngẫu nhiên trong bảng Customers như ví dụ 6-63.

Ví dụ 6-63: Khai báo sử dụng hàm NEWID

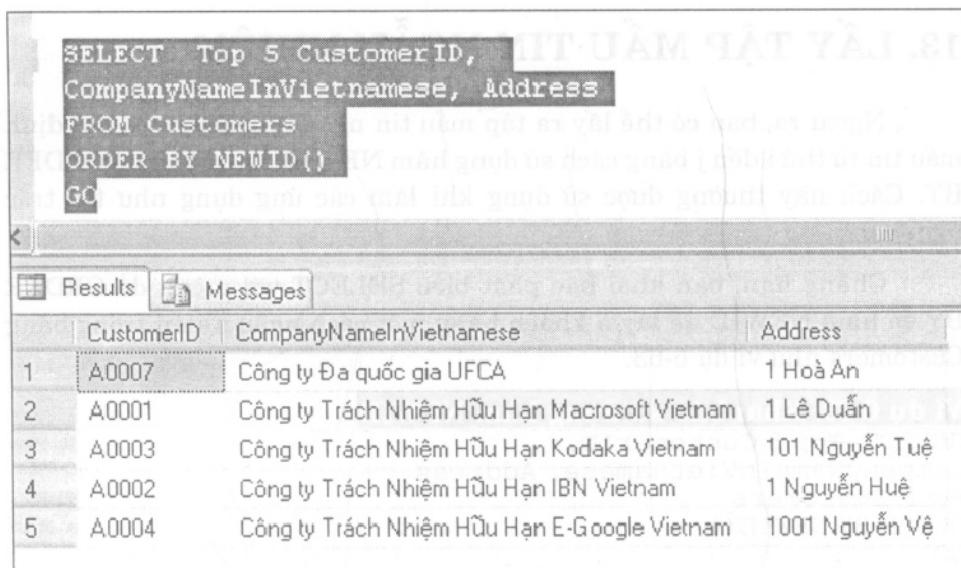
```
SELECT Top 5 CustomerID,
CompanyNameInVietnamese, Address
FROM Customers
ORDER BY NEWID()
GO
```

Khi thực thi lần đầu tiên, bạn có thể nhận được kết quả trả về như hình 6-63.

	CustomerID	CompanyNameInVietnamese	Address
1	A0001	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	1 Lê Duẩn
2	A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	101 Nguyễn Tuệ
3	A0009	Trung tâm giáo dục Vietnam	1 An Huy
4	A0005	Công ty Cổ phần Suzumi Vietnam	101 Hoàng Văn Thụ
5	A0002	Công ty Trách Nhiệm Hữu Hạn IBN Vietnam	1 Nguyễn Huệ

Hình 6-63: Lấy ngẫu nhiên ra 5 khách hàng.

Nếu bạn thực thi lần kế tiếp thì kết quả ngẫu nhiên trả về là danh sách 5 khách hàng khác như hình 6-64.



```

SELECT Top 5 CustomerID, CompanyNameInVietnamese, Address
FROM Customers
ORDER BY NEWID()
GO
  
```

The screenshot shows a SQL Server Management Studio window. At the top, there is a query editor pane containing the following T-SQL code:

```

SELECT Top 5 CustomerID, CompanyNameInVietnamese, Address
FROM Customers
ORDER BY NEWID()
GO
  
```

Below the query editor is a results grid. The grid has three columns: CustomerID, CompanyNameInVietnamese, and Address. The data is as follows:

	CustomerID	CompanyNameInVietnamese	Address
1	A0007	Công ty Đa quốc gia UFC&A	1 Hoà An
2	A0001	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	1 Lê Duẩn
3	A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	101 Nguyễn Tuệ
4	A0002	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	1 Nguyễn Huệ
5	A0004	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	1001 Nguyễn Vé

Hình 6-64: Danh sách 5 khách hàng.

14. KẾT CHƯƠNG

Chúng ta vừa tìm hiểu chi tiết phát biểu SELECT với các mệnh đề cho phép lấy ra tập tin đúng như yêu cầu. Với những kiến thức này, bạn có thể tạo ra các đối tượng View và Stored Procedure để có thể thực hiện các tác vụ truy vấn dữ liệu cho ứng dụng.

Bạn cũng có thể tạo ra tập dữ liệu dạng tóm tắt bằng cách sử dụng phép toán ROLLUP, CUBE cùng với hàm GROUPING với mệnh đề GROUP BY.

Ngoài ra, trong chương này bạn cũng được tham khảo cách trình bày dữ liệu dạng biểu thức bảng cùng với mệnh đề ORDER BY kết hợp với mệnh đề OVER để tạo ra bảng dữ liệu với cột dữ liệu tương ứng với thứ tự mẫu tin, rồi sử dụng mệnh đề WHERE để lấy ra tập dữ liệu từ mẫu tin thứ i đến j nhằm phục vụ cho quá trình cài đặt phân trang trên ứng dụng.

Trong chương kế tiếp, chúng ta tiếp tục tìm hiểu 3 phát biểu SQL dạng DML còn lại là INSERT, UPDATE và DELETE trước khi tiếp tục tìm hiểu đối tượng Stored Procedure, Function, Trigger trong tập kế tiếp.

Chương 7:

PHÁT BIỂU T-SQL CƠ BẢN DẠNG XỬ LÝ DỮ LIỆU

Tóm tắt chương 7

Tiếp theo phát biểu SELECT dùng để truy vấn dữ liệu cùng với một số mệnh đề, từ khóa, biểu thức và phép toán, cho phép chúng ta kết xuất tập dữ liệu theo yêu cầu của ứng dụng, bạn sẽ tiếp tục tìm hiểu 3 phát biểu dạng DML còn lại là INSERT, UPDATE và DELETE.

Mặc dù phát biểu SELECT được sử dụng nhiều nhất khi làm việc với cơ sở dữ liệu, nhưng để xử lý dữ liệu thì bạn không thể thiếu phát biểu INSERT, UPDATE và DELETE (DML).

Các vấn đề chính sẽ được đề cập:

- ✓ Phát biểu INSERT.
- ✓ Phát biểu UPDATE
- ✓ Phát biểu DELETE.

1. PHÁT BIỂU INSERT

Phát biểu INSERT cho phép bạn thêm mẩu tin vào bảng dữ liệu từ giá trị cụ thể, biểu thức, hàm, truy vấn từ bảng dữ liệu hay đối tượng View và biểu thức bảng dữ liệu.

Để thêm dữ liệu bằng phát biểu INSERT, bạn có thể sử dụng cú pháp như sau:

```
| WITH <common_table_expression> [ ,...n ] ]
INSERT
[ TOP ( expression ) | PERCENT ] ]
[ INTO ]
{ <object> | rowset_function_limited
[ WITH ( <Table_Hint_Limited> [ ...n ] ) ] }
```

M® 236**Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu**

```

}
(
[ ( column_list ) ]
[ <OUTPUT Clause> ]
{ VALUES ( { DEFAULT | NULL | expression } [ ,...n ] )
| derived_table
| execute_statement
}
)
! DEFAULT VALUES
[;]

```

Trong đó, tùy chọn object được định nghĩa như sau:

```

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

```

Để được thêm dữ liệu vào bảng bằng phát biểu INSERT bạn phải là thành viên của sysadmin, hay là chủ nhân của cơ sở dữ liệu (db_owner) hoặc được cấp quyền ghi dữ liệu (db_datawriter).

Nếu bạn là thành viên của sysadmin, db_owner, db_securityadmin hay chủ nhân của đối tượng bảng thì bạn có thể khai báo quyền thêm dữ liệu cho người sử dụng khác.

Chú ý: Bạn có thể tham khảo phần trình bày về người dùng trong SQL Server, người sử dụng cơ sở dữ liệu, quyền của người sử dụng trên từng đối tượng cơ sở dữ liệu trong những chương kế tiếp.

Tương tự như phát biểu SELECT, phát biểu INSERT có cấu trúc cũng rất phức tạp, chính vì vậy chúng ta sẽ tìm hiểu cách sử dụng phát biểu này qua từng phần cụ thể.

1.1. Phát biểu INSERT cơ bản

Khi sử dụng phát biểu INSERT, bạn có thể chỉ định cột dữ liệu trong bảng cần thêm vào bằng cách sử dụng cú pháp tương tự như sau:

```

INSERT INTO TABLENAME(Col1, Col2, ..., ColN)
VALUES (Value1, Value2, ..., ValueN)
INSERT TABLENAME(Col1, Col2, ..., ColN)
VALUES (Value1, Value2, ..., ValueN)

```

Chẳng hạn, bạn khai báo để thêm mẫu tin vào bảng dữ liệu có tên SupplierTypes với cú pháp như ví dụ 7-1.

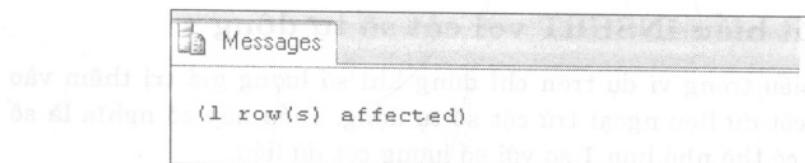
Chương 7: Phát biểu T-SQL cơ bản đang xử lý dữ liệu

237 M®

Ví dụ 7-1: Khai báo thêm mẩu tip

```
INSERT INTO SupplierTypes  
(SupplierTypeID, SupplierTypeName)  
VALUES('COR', 'Corporation')  
GO
```

Nếu thực thi phát biểu INSERT trong ví dụ trên, bạn có thể tìm thấy kết quả trình bày như hình 7-1.



Hình 7-1: Thêm mẫu tin vào bảng SupplierTypes

Chú ý: Khi thực thi phát biểu INSERT như trên thì chỉ một mẫu tin được thêm vào bảng dữ liệu.

Nếu bạn thực thi lại phát biểu trên một lần nữa thì lỗi sẽ phát sinh do dữ liệu trùng lặp tại cột khóa chính như sau:

do đó hệ thống lập lại cột khóa chính như sau:
Msg 2627, Level 14, State 1, Line 1
Violation of PRIMARY KEY constraint
'PK__SupplierTypes__4F47C5E3'.
Cannot insert duplicate key in object 'dbo.SupplierTypes'.
The statement has been terminated.

Nếu bạn thêm mẫu tin vào bảng mà cung cấp giá trị cho tất cả các cột dữ liệu theo đúng thứ tự của các cột thì không cần chỉ định cột dữ liệu trong phần khai báo bảng.

INSERT INTO TABLENAME
VALUES(Value1, Value2, ..., ValueN)

Giả sử, bảng dữ liệu CustomerTypes có hai cột CustomerTypeID và CustomerTypeName, giá trị truyền vào cũng có hai giá trị 'COR' và 'Corporation' thì bạn khai báo như ví dụ 7-2.

Ví dụ 7-2: Khai báo không chỉ định tên cột dữ liệu

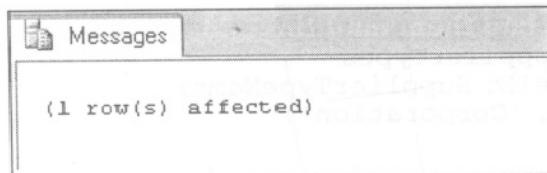
```
1. Tạo bảng CustomerTypes
```

```
CREATE TABLE CustomerTypes  
    (CustomerTypeID INT IDENTITY(1,1) PRIMARY KEY,  
     CustomerType NVARCHAR(50) NOT NULL);
```

```
2. Nhập dữ liệu vào bảng
```

```
INSERT INTO CustomerTypes  
VALUES ('COR', 'Corporation')  
GO
```

Khi thực thi phát biểu INSERT trong ví dụ trên, bạn có thể tìm thấy kết quả trình bày như hình 7-2.



Hình 7-2: Thêm mẫu tin vào bảng CustomerTypes.

1.2. Phát biểu INSERT với cột số tự động

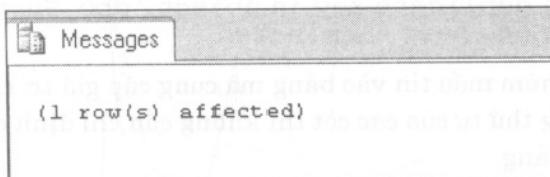
Phát biểu trong ví dụ trên chỉ đúng khi số lượng giá trị thêm vào bảng với số cột dữ liệu ngoại trừ cột số tự động. Điều này có nghĩa là số lượng giá trị có thể nhỏ hơn 1 so với số lượng cột dữ liệu.

Chẳng hạn, bạn thêm dữ liệu vào bảng Categories mà chỉ cần cung cấp hai giá trị thay vì ba, bởi vì bảng này có cột số tự động.

Ví dụ 7-3: Khai báo thêm dữ liệu vào bảng có cột số tự động

```
INSERT INTO Categories
VALUES ('School bag', 0)
GO
```

Khi thực thi phát biểu trên, bạn cũng nhận được kết quả trả về như hình 7-3.



Hình 7-3: Thêm mẫu tin thành công.

Chú ý: Phát biểu trên chỉ đúng khi cột số tự động nằm ở vị trí đầu tiên. Những cột có khai báo giá trị mặc định, bạn cũng phải khai báo trong phần VALUES.

Phát biểu INSERT có chỉ định cột dữ liệu cho bảng cần thêm mẫu tin sẽ được sử dụng hầu hết các mẫu nhập liệu của ứng dụng.

Bạn nên dùng cú pháp này để tránh trường hợp khi thêm mới cột dữ liệu vào bảng, đối với trường hợp này lỗi sẽ phát sinh nếu bạn khai báo không chỉ định cột dữ liệu trong bảng dữ liệu.

Nếu dữ liệu là Unicode, bạn cần chỉ định từ khóa N trước giá trị và bảo đảm kiểu dữ liệu của cột trong bảng phải là một trong các kiểu dữ liệu

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

239

NCHAR, NVARCHAR hay NTEXT khi sử dụng phát biểu INSERT như ví dụ 7-4.

Ví dụ 7-4: Khai báo thêm dữ liệu Unicode

```
INSERT INTO CustomerTypes
VALUES ('JVT', N'Công ty Liên Doanh')
GO
```

Khi thực thi phát biểu trên, bạn cũng nhận được kết quả trả về thành công, bằng cách sử dụng phát biểu SELECT để kiểm tra dữ liệu vừa thêm vào như hình 7-4.

	CustomerTypeID	CustomerTypeName
1	COR	Corporation
2	JVT	Công ty Liên Doanh
3	PRI	Private

Hình 7-4: Thêm dữ liệu Unicode.

1.3. Phát biểu INSERT từ bảng dữ liệu

Bạn có thể thêm dữ liệu vào bảng bằng cách lấy dữ liệu từ câu truy vấn, đối với trường hợp này số lượng mẫu tin phụ thuộc vào kết quả của câu truy vấn trả về.

```
INSERT INTO TABLENAME1 (Col1, Col2, ..., ColN)
SELECT Col1, Col2, ..., ColN FROM TABLENAME2
```

Hoặc bạn có thể không chỉ định tên cột dữ liệu bằng cách khai báo tương tự như sau:

```
INSERT INTO TABLENAME1
SELECT * FROM TABLENAME2 | VIEWNAME
```

Chú ý: Trong trường hợp trên, bạn phải bảo đảm số lượng cột dữ liệu của bảng nguồn (trả về dữ liệu) bằng hoặc ít hơn 1 cột nếu bảng đích có cột số tự động.

Chẳng hạn, bạn sao lưu dữ liệu của bảng SalesInvoices vào bảng SalesInvoicesForBackup bằng cách khai báo như ví dụ 7-5.

Ví dụ 7-5: Khai báo thêm dữ liệu từ câu truy vấn

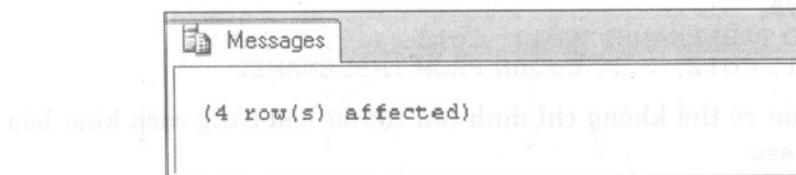
```
INSERT INTO SalesInvoicesForBackup
SELECT * FROM SalesInvoices
GO
```

Chú ý: bảng dữ liệu SalesInvoicesForBackup có cấu trúc hoàn toàn giống với cấu trúc của bảng SalesInvoices, bạn có thể tìm thấy phát biểu CREATE TABLE để tạo bảng này như ví dụ 7-6.

Ví dụ 7-6: Khai báo tạo bảng SalesInvoicesForBackup

```
CREATE TABLE SalesInvoicesForBackup
(
    InvoiceNo VARCHAR(10) NOT NULL PRIMARY KEY,
    InvoiceBatchNo VARCHAR(10) NULL,
    DueDate SMALLDATETIME DEFAULT GETDATE(),
    InvoiceTypeID CHAR(3) NOT NULL,
    CustomerID CHAR(5) NOT NULL ,
    CurrencyID CHAR(3) NOT NULL,
    ExchangeRate DECIMAL DEFAULT 0,
    DescriptionInVietnamese NVARCHAR(150) NOT NULL,
    DescriptionInSecondLanguage VARCHAR(150) NULL,
    ShowDescriptionInVietnamese BIT DEFAULT 1,
    InvoiceDiscontinued BIT DEFAULT 0,
    EntryDate SMALLDATETIME DEFAULT GETDATE(),
    UserName VARCHAR(10) NOT NULL
)
GO
```

Khi thực thi phát biểu trong ví dụ trên, kết quả trả về là 4 mẫu tin được thêm vào bảng SalesInvoicesForBackup từ bảng SalesInvoices như hình 7-5.



Hình 7-5: Thêm dữ liệu từ câu truy vấn.

1.4. Phát biểu INSERT với đối tượng View

Ngoài cách thêm dữ liệu trực tiếp vào bảng dữ liệu, bạn cũng có thể thêm dữ liệu vào Table thông qua đối tượng View của chính Table đó.

```
INSERT INTO VIEWNAME(Col1, Col2, ..., ColN)
VALUES (Value1, Value2, ..., ValueN)
```

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

241

Tuy nhiên, bạn có thể khai báo thêm dữ liệu vào Table thông qua View bằng cú pháp như sau:

```
INSERT INTO VIEWNAME
SELECT * FROM VIEWNAME + TABLENAME
```

Để làm điều này, trước tiên bạn khai báo đối tượng View cho bảng Banks bằng cú pháp như ví dụ 7-7.

Ví dụ 7-7: Khai báo đối tượng View

```
CREATE VIEW vwBANKS
AS
SELECT BankID, BankName, BankAddress, ProvinceID
FROM dbo.Banks
GO
```

Kế đến, bạn khai báo phát biểu INSERT để thêm dữ liệu vào bảng Banks thông qua đối tượng View như ví dụ 7-8.

Ví dụ 7-8: Khai báo thêm dữ liệu vào đối tượng View

```
INSERT INTO vwBanks (BankID, BankName, BankAddress,
ProvinceID)
VALUES ('SBA', 'Southern Bank',
--Dữ liệu Unicode
N'1 Lê Lai', 'HCM')
GO
```

1.5. Phát biểu INSERT với mệnh đề TOP

Tương tự như trường hợp sử dụng từ khóa TOP trong phân trình bày phát biểu SELECT, bạn cũng có thể sử dụng từ khóa này trong phát biểu INSERT bằng cách sử dụng cú pháp như sau:

```
INSERT TOP (10) INTO TABLENAME1
SELECT * FROM TABLENAME2 + VIEWNAME
```

Chú ý: Mệnh đề TOP thường được sử dụng trong phát biểu INSERT khi bạn thêm mẩu tin vào bảng đích từ câu truy vấn.

Chẳng hạn, bạn có thể khai báo phát biểu INSERT để thêm dữ liệu vào bảng PurchaseInvoiceDetailsForBackup từ bảng dữ liệu PurchaseInvoiceDetails như ví dụ 7-9.

Ví dụ 7-9: Khai báo mệnh đề TOP

--Sử dụng mệnh đề TOP

```
INSERT TOP(5) INTO PurchaseInvoiceDetailsForBackup
```

--Lấy dữ liệu từ bảng khác

```
SELECT * FROM PurchaseInvoiceDetails
GO
```

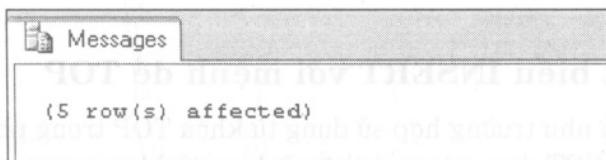
Trong đó, bảng dữ liệu PurchaseInvoiceDetailsForBackup có cấu trúc như ví dụ 7-10.

Ví dụ 7-10: Khai báo tạo bảng PurchaseInvoiceDetailsForBackup

```
CREATE TABLE PurchaseInvoiceDetailsForBackup
(
    OrdinalNumber tinyint NOT NULL,
    InvoiceNo VARCHAR(10) NOT NULL,
    ProductID VARCHAR(10) NOT NULL,
    Quantity DECIMAL DEFAULT 0,
    Price DECIMAL DEFAULT 0,
    Discount DECIMAL DEFAULT 0,
    VATRate DECIMAL DEFAULT 0,
    PRIMARY KEY
        (OrdinalNumber, InvoiceNo, ProductID)
)
GO
```

Mặc dù phát biểu SELECT * FROM PurchaseInvoiceDetails trả về danh sách nhiều mẫu tin, nhưng chỉ thêm 5 mẫu tin vào bảng PurchaseInvoiceDetailsForBackup.

Khi thực thi phát biểu INSERT trong ví dụ trên, bạn có thể tìm thấy kết quả trả về như hình 7-6.



Hình 7-6: Sử dụng mệnh đề TOP.

Chú ý: Bạn có thể sử dụng mệnh đề TOP trong phát biểu SELECT thay vì khai báo trong phát biểu INSERT tương tự như ví dụ 7-11.

Ví dụ 7-11: Khai báo mệnh đề TOP trong phát biểu SELECT

```
INSERT INTO PurchaseInvoiceDetailsForBackup
```

```
--Sử dụng mệnh đề TOP
SELECT TOP(5) * FROM PurchaseInvoiceDetails
GO
```

Khi thực thi phát biểu trên, bạn cũng nhận được kết quả tương tự như hình 7-6.

1.6. Phát biểu INSERT với phát biểu EXECUTE

Chúng ta vừa khai báo phát biểu INSERT để thêm dữ liệu vào bảng đích từ câu truy vấn dữ liệu dựa vào đối tượng Table hoặc View.

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

243

Nếu bạn có dữ liệu nguồn được trả về từ thủ tục nội tại thì sử dụng phát biểu EXECUTE như sau:

```
INSERT INTO TABLENAME
EXECUTE PROCEDURENAME
```

Để tìm hiểu cách sử dụng phát biểu EXECUTE, trước tiên bạn tạo bảng dữ liệu có tên SalesInvoiceDetailsForBackup với cấu trúc như ví dụ 7-12.

Ví dụ 7-12: Khai báo tạo bảng dữ liệu

```
CREATE TABLE SalesInvoiceDetailsForBackup
```

```
(  
    OrdinalNumber tinyint NOT NULL,  
    InvoiceNo VARCHAR(10) NOT NULL,  
    ProductID VARCHAR(10) NOT NULL,  
    Quantity DECIMAL DEFAULT 0,  
    Price DECIMAL DEFAULT 0,  
    Discount DECIMAL DEFAULT 0,  
    VATRate DECIMAL DEFAULT 0,  
    PRIMARY KEY (OrdinalNumber,  
    InvoiceNo, ProductID)
```

```
)  
GO
```

Kế đến, bạn khai báo thủ tục nội tại BackupSalesInvoices để lấy ra tập dữ liệu từ bảng dữ liệu có tên SalesInvoiceDetails như ví dụ 7-13.

Ví dụ 7-13: Khai báo thủ tục nội tại

```
CREATE PROC BackupSalesInvoices
AS
```

```
    SELECT * FROM SalesInvoiceDetails  
GO
```

Chú ý: Nếu bạn thực thi thủ tục nội tại trên, tập dữ liệu trả về như hình 7-7.

OrdinalNumber	InvoiceNo	ProductID	Quantity	Price	Discount	VATTaxRate	VATTaxAmount	InvoiceAmount	
1	S10000001	P0001	100	10000	10	10	100000	900000	
2	1	S10000002	P0003	100	10000	100000	10	100000	900000
3	1	S10000003	P0002	100	10000	10	10	100000	900000
4	2	S10000001	P0002	200	10000	10	10	200000	1800000
5	2	S10000002	P0002	200	10000	200000	10	200000	1800000
6	2	S10000003	P0003	300	10000	10	10	300000	2700000
7	3	S10000004	P0004	100	10000	10	10	100000	900000

Hình 7-7: Thực thi thủ tục nội tại BackupSalesInvoices.

Sau đó, bạn khai báo phát biểu INSERT để thêm dữ liệu bằng cách thực thi thủ tục nội tại như ví dụ 7-14.

Ví dụ 7-14: Khai báo phát biểu EXECUTE

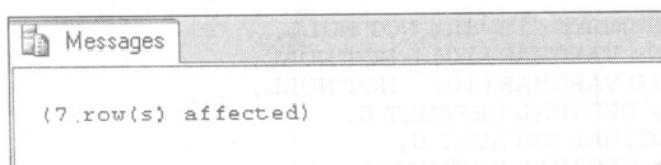
```
INSERT INTO SalesInvoiceDetailsForBackup
```

--Sử dụng phát biểu EXECUTE hay EXEC

```
EXECUTE BackupSalesInvoices
```

```
GO
```

Nếu thực thi phát biểu trên, bạn có thể nhận được kết quả trả về như hình 7-8.



Hình 7-8: Sử dụng phát biểu EXECUTE.

1.7. Phát biểu INSERT với phát biểu WITH

Tương tự như cách sử dụng phát biểu WITH trong phát biểu SELECT đã trình bày trong chương trước, bạn có thể sử dụng phát biểu WITH khi làm việc với phát biểu INSERT.

Chẳng hạn, bạn khai báo để thêm một số mẫu tin từ bảng dữ liệu có tên PurchaseInvoiceDetailsForBackup vào bảng dữ liệu có tên PurchaseInvoiceDetails bằng cách sử dụng phát biểu WITH như ví dụ 7-15.

Ví dụ 7-15: Khai báo sử dụng phát biểu WITH

--Sử dụng biểu thức bảng
WITH PurchaseForBackup

```
AS
```

```
(
```

```
    SELECT * FROM PurchaseInvoiceDetails  
    WHERE ProductID = 'P00001'
```

```
)
```

```
INSERT INTO PurchaseInvoiceDetailsForBackup  
SELECT * FROM PurchaseForBackup
```

```
GO
```

Lưu ý: Bạn có thể tìm hiểu thêm phát biểu INSERT trong phần trình bày thủ tục nội tại trong chương Thủ tục nội tại trong tập 2.

1.8. Phát biểu INSERT trên nhiều cơ sở dữ liệu

Nếu bạn thêm dữ liệu vào bảng dữ liệu nằm ở cơ sở dữ liệu khác với cơ sở dữ liệu hiện hành thì bạn có thể chỉ định tên cơ sở dữ liệu và chủ nhân của đối tượng Table bằng cách khai báo như sau:

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

Chẳng hạn, bạn đang đứng tại cơ sở dữ liệu AccountSystem nhưng thêm dữ liệu vào bảng Banks trong cơ sở dữ liệu AccountingSystem bằng phát biểu như ví dụ 7-16.

Ví dụ 7-16: Khai báo thêm dữ liệu vào bảng của cơ sở dữ liệu khác

```
INSERT INTO AccountingSystem.dbo.Banks
VALUES ('UAB', 'USA-Asia Commercial Bank',
'2 Lê Duan St., Dist.1', 'HCM')
GO
```

Trong trường hợp thêm dữ liệu vào bảng thuộc cơ sở dữ liệu hiện hành mà dữ liệu thì lấy ra từ cơ sở dữ liệu khác, bạn có thể khai báo như ví dụ 7-17.

Ví dụ 7-17: Khai báo lấy dữ liệu từ cơ sở dữ liệu khác

```
INSERT INTO Banks(BankID, BankName,
    BankAddress, ProvinceID)
Select * from AccountingSystem.dbo.Banks
GO
```

1.9. Phát biểu INSERT với mệnh đề OUTPUT

Mệnh đề OUTPUT cho phép lấy ra được tập dữ liệu vừa mới thêm vào bảng. Bạn có thể sử dụng mệnh đề này bằng cách sử dụng cú pháp như sau:

```
<OUTPUT_CLAUSE> ::= 
{
    [ OUTPUT <dml_select_list> INTO { @table_variable |
    output_table } [ { column_list } ] ]
    [ OUTPUT <dml_select_list> ]
}
<dml_select_list> ::=
{ <column_name> | scalar_expression } [ [AS]
column_alias_identifier ]
[ ,...n ]
<column_name> ::=
{ INSERTED | from_table_name } . { * | column_name }
```

M® 246**Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu**

Để khám phá mệnh đề OUTPUT, giả sử bạn tạo bảng dữ liệu có tên ImportDetailsForBackup bằng phát biểu CREATE TABLE như ví dụ 7-18.

Ví dụ 7-18: Khai báo tạo bảng ImportDetailsForBackup

```
CREATE TABLE ImportDetailsForBackup
(
    OrdinalNumber tinyint NOT NULL,
    ImportNo      VARCHAR(10)  NOT NULL,
    ProductID    VARCHAR(10)  NOT NULL,
    Quantity      DECIMAL DEFAULT 0,
    PRIMARY KEY
        (OrdinalNumber, ImportNo, ProductID)
)
GO
```

Kế đến, bạn khai báo phát biểu INSERT để thêm dữ liệu từ bảng ImportDetailsForBackup vào bảng ImportDetails bằng cách đồng thời thấy danh sách mẫu tin vừa thêm vào thông qua bí danh bảng INSERTED.

Ví dụ 7-19: Khai báo mệnh đề OUTPUT

```
INSERT INTO ImportDetailsForBackup
OUTPUT INSERTED.OrdinalNumber,
       INSERTED.ImportNo,
       INSERTED.ProductID,
       INSERTED.Quantity
SELECT * FROM ImportDetails
GO
```

Chú ý: Mệnh đề OUTPUT cho phép xuất dữ liệu ra màn hình để tham khảo hay sử dụng nó để thêm vào bảng dữ liệu khác.

Chẳng hạn, bạn khai báo thêm phát biểu SELECT trong ví dụ trên như ví dụ 7-20.

Ví dụ 7-20: Khai báo mệnh đề OUTPUT

```
INSERT INTO ImportDetailsForBackup
OUTPUT
       INSERTED.OrdinalNumber,
       INSERTED.ImportNo,
       INSERTED.ProductID,
       INSERTED.Quantity
SELECT * FROM ImportDetails
GO
SELECT * FROM ImportDetailsForBackup
GO
```

Lưu ý: Trước khi thực hiện phát biểu trong ví dụ trên, bạn có thể sử dụng phát biểu DELETE để xóa mẫu tin đã tồn tại trong bảng dữ liệu ImportDetailsForBackup. Ngoài ra, để tham khảo bảng dữ liệu định danh

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

247

INSERTED, bạn có thể tìm đọc chương trình bày về đối tượng TRIGGER trong tập 2.

Khi thực thi, bạn có thể tìm thấy danh sách mẫu tin thêm vào bảng ImportDetailsForBackup với danh sách mẫu tin do mệnh đề OUTPUT xuất ra sẽ bằng nhau như hình 7-9.

OrdinalNumber	ImportNo	ProductID	Quantity
1	I00000101	P00001	500
2	I00000101	P00002	600
3	I00000101	P00003	700

Hình 7-9: Sử dụng mệnh đề OUTPUT.

Ngoài ra, bạn cũng có thể thêm kết quả do mệnh đề OUTPUT kết xuất để thêm vào bảng dữ liệu nào đó thì sử dụng mẫu tin INTO.

Ví dụ, bạn khai báo phát biểu CREATE TABLE để tạo bảng ảo dùng để chứa danh sách mẫu tin do mệnh đề OUTPUT xuất ra. Sau đó, khai báo phát biểu INSERT với mệnh đề OUTPUT và INTO như ví dụ 7-21.

Ví dụ 7-21: Khai báo tạo bảng ảo chứa dữ liệu do OUTPUT xuất ra

```
CREATE TABLE #ImportDetailsTemporary
(
    OrdinalNumber      tinyint NOT NULL,
    ImportNo           VARCHAR(10)    NOT NULL,
    ProductID          VARCHAR(10)    NOT NULL,
    Quantity           DECIMAL DEFAULT 0,
    PRIMARY KEY (OrdinalNumber, ImportNo, ProductID)
)
GO
INSERT INTO ImportDetailsForBackup
OUTPUT INSERTED.OrdinalNumber,
        INSERTED.ImportNo,
        INSERTED.ProductID,
        INSERTED.Quantity
INTO #ImportDetailsTemporary
SELECT * FROM ImportDetails
SELECT * FROM #ImportDetailsTemporary
GO
```

Khi thực thi phát biểu CREATE TABLE, INSERT và SELECT trong ví dụ trên, bạn có thể tìm thấy danh sách mẫu tin thêm vào bảng tạm #ImportDetailsTemporary bằng mệnh đề OUTPUT và INTO trong khi

248

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

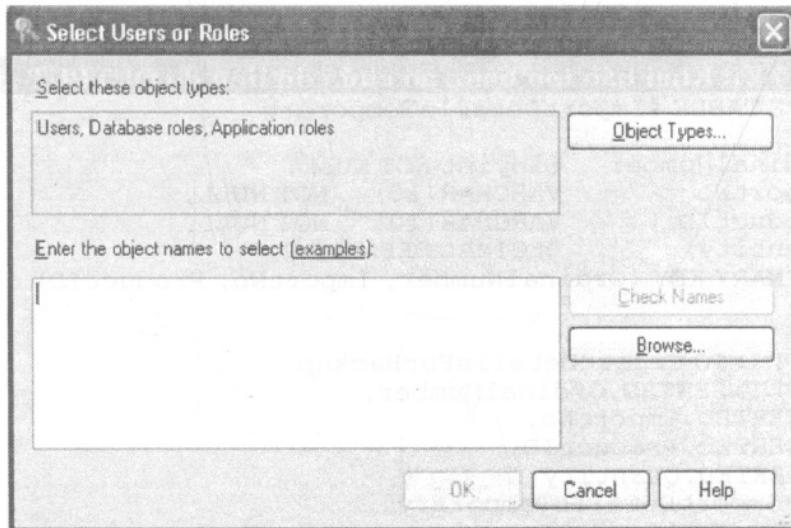
thêm vào bảng ImportDetailsForBackup từ bảng ImportDetails như hình 7-10.

	OrdinalNumber	ImportNo	ProductID	Quantity
1	1	100000101	P00001	500
2	2	100000101	P00002	600
3	3	100000101	P00003	700

Hình 7-10: Sử dụng mệnh đề OUTPUT và INTO.

Trong ví dụ ở trên, bạn cần thực thi phát biểu DELETE và DROP TABLE để xóa dữ liệu trong bảng ImportDetailsForBackup và loại bỏ bảng tạm ImportDetailsTemporary nếu chạy lại ví dụ trên.

Chú ý: Để thực thi phát biểu INSERT hay thêm mới mẫu tin trong bảng, người sử dụng cần có quyền INSERT. Chẳng hạn, nếu bạn muốn gán quyền thêm mới bảng MS cho bảng Customers, bạn chọn vào nút Add và cửa sổ xuất hiện như hình 7-11.

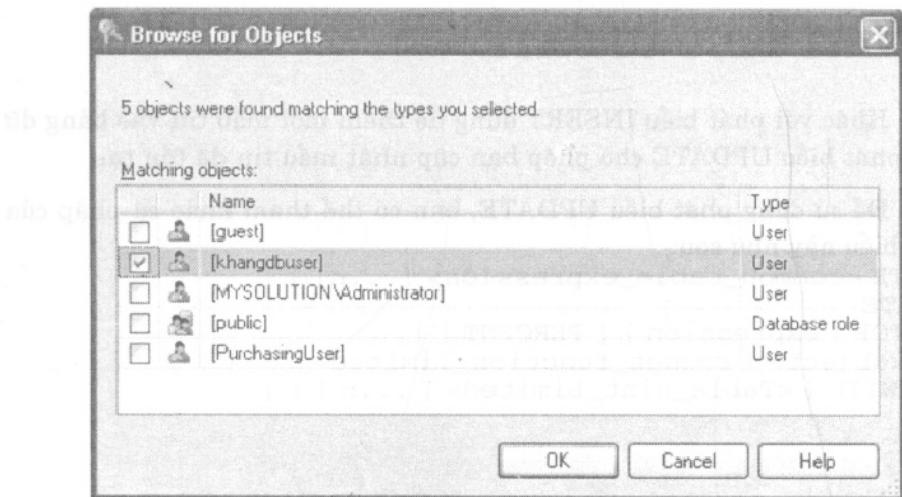


Hình 7-11: Thêm tài khoản người sử dụng.

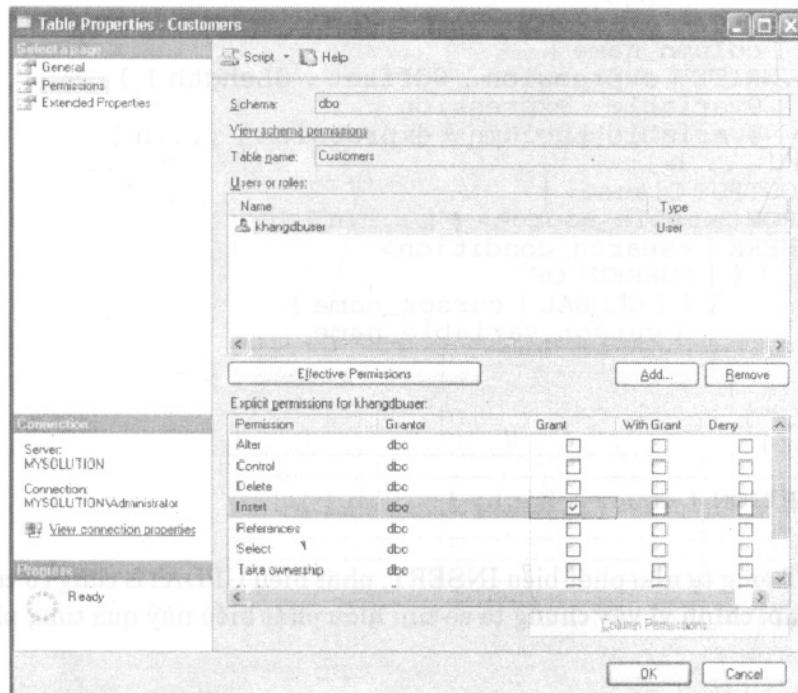
Nhấn nút Browse cửa sổ liệt kê danh sách người sử dụng xuất hiện như hình 7-12.

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

249

**Hình 7-12:** Chọn tên người sử dụng.

Nhấn nút OK, rồi chọn vào CheckBox ứng với quyền INSERT như hình 7-13.

**Hình 7-13:** Cấp quyền INSERT.

2. PHÁT BIỂU UPDATE

Khác với phát biểu INSERT dùng để thêm mới mẫu tin vào bảng dữ liệu, phát biểu UPDATE cho phép bạn cập nhật mẫu tin đã tồn tại.

Để sử dụng phát biểu UPDATE, bạn có thể tham khảo cú pháp của phát biểu này như sau:

```
[ WITH <common_table_expression> [ . . . n ] ]
UPDATE
[ TOP ( expression ) [ PERCENT ] ]
( <object> | rowset_function_limited
[ WITH ( <Table_Hint_Limited> [ . . . n ] ) ]
)
SET
{
    column_name = { expression | DEFAULT | NULL }
    | { udt_column_name. {
        property_name = expression
        | field_name = expression
        | method_name ( argument [ , . . . n ] )
    }
}
| column_name {
    .WRITE ( expression , @Offset , @Length )
    | @variable = expression
    | @variable = column_name = expression [ , . . . n ]
} [ , . . . n ]
[ <OUTPUT Clause> ]
[ FROM ( <table_source> ) [ , . . . n ] ]
[ WHERE { <search_condition>
    | ( [ CURRENT OF
        { { [ GLOBAL ] cursor_name }
        | cursor_variable_name
    }
}
}
]
[ OPTION ( <query_hint> [ , . . . n ] ) ]
[ ; ]
```

Tương tự như phát biểu INSERT, phát biểu UPDATE cũng có cú pháp phức tạp, chính vì vậy chúng ta sẽ tìm hiểu phát biểu này qua từng phần cơ bản.

2.1. Phát biểu UPDATE đơn giản

Bạn có thể cập nhật mẫu tin trong bảng dữ liệu bằng cách sử dụng cú pháp tương tự như sau:

```
UPDATE TABLENAME
SET Col1=Value1
[, Coli = Valuei]
[ WHERE whereCondition]
```

Trong đó, whereCondition mệnh đề cho phép lọc dữ liệu theo tiêu chí nào đó để cập nhật.

Chẳng hạn, bạn có thể cập nhật cột CountryID của bảng Provinces với giá trị là 'VNA' như ví dụ 7-22.

Ví dụ 7-22: Khai báo phát biểu UPDATE

```
UPDATE Provinces
SET CountryID = 'VNA'
```

Chú ý: Khi thực thi phát biểu UPDATE trên, tại cột CountryID của tất cả mẫu tin trong bảng Provinces sẽ được cập nhật với giá trị 'VNA'.

Tuy nhiên, thông thường khi sử dụng phát biểu UPDATE, chúng ta thường chỉ định số mẫu tin theo tiêu chí nào đó. Chẳng hạn, bạn khai báo ví dụ trên với mệnh đề WHERE như ví dụ 7-23.

Ví dụ 7-23: Khai báo phát biểu UPDATE

```
UPDATE Provinces
SET CountryID = 'VNA'
WHERE CountryID IS NULL
```

Trong trường hợp cập nhật giá trị cho nhiều cột, bạn sử dụng dấu phẩy phân cách như ví dụ 7-24.

Ví dụ 7-24: Khai báo cập nhật nhiều cột dữ liệu

```
UPDATE Products
SET ProductNameInVietnamese =
N'Túi xách dùng cho học sinh nữ',
ProductNameInSecondLanguage =
'School bag for Girl'
WHERE ProductID = 'P00002'
GO
```

2.2. Phát biểu UPDATE với phát biểu SELECT

Bạn có thể cập nhật cột dữ liệu của bảng bằng giá trị của cột dữ liệu từ bảng khác bằng cách kết hợp phát biểu UPDATE và phát biểu SELECT như ví dụ 7-25.

Ví dụ 7-25: Khai báo cập nhật dữ liệu từ bảng khác

```
UPDATE ProductInStocks
SET GoodProductInStock = (SELECT sum(Quantity) FROM
ImportDetails
WHERE ProductInStocks.ProductID = ProductID)
```

2.3. Phát biểu UPDATE với mệnh đề TOP

Tương tự như phát biểu INSERT, bạn cũng có thể sử dụng mệnh đề TOP trong phát biểu UPDATE với cú pháp như sau:

```
UPDATE TOP (n) TABLENAME
SET Col1=Value1
[,Coli = Valuei]
[ WHERE whereCondition]
```

Trong đó, n là số lượng mẫu tin cần cập nhật. Chẳng hạn, bạn chỉ cập nhật 5 mẫu tin đầu tiên trong bảng PurchaseInvoiceDetails bằng phát biểu như ví dụ 7-26.

Ví dụ 7-26: Khai báo cập nhật 5 mẫu tin

```
UPDATE TOP (5) PurchaseInvoiceDetails
SET Discount = 10
GO
```

Lưu ý: Nếu số lượng mẫu tin ít hơn tham số n thì phát biểu UPDATE sẽ cập nhật số mẫu tin mà nó tìm thấy.

2.4. Phát biểu UPDATE với mệnh đề OUTPUT

Tương tự như phát biểu INSERT, bạn cũng có thể sử dụng mệnh đề OUTPUT để lấy ra tập dữ liệu vừa được cập nhật bằng cú pháp như sau:

```
<OUTPUT_CLAUSE> ::=
{
    [ OUTPUT <dml_select_list> INTO { @table_variable |
    output_table } [ (column_list) ]
    [ OUTPUT <dml_select_list> ]
}
<dml_select_list> ::=
{ <column_name> | scalar_expression } [ [AS]
column_alias_identifier ]
[ ,...n ]
<column_name> ::=
{ INSERTED | from_table_name } . { * | column_name }
```

Chẳng hạn, bạn muốn biết những mẫu tin nào trong bảng SalesInvoiceDetails vừa được cập nhật bằng phát biểu UPDATE thì sử dụng mệnh đề OUTPUT như ví dụ 7-27.

Ví dụ 7-27: Khai báo sử dụng mệnh đề OUTPUT

```
UPDATE SalesInvoiceDetails
```

```
SET Discount = 10
```

--Sử dụng mệnh đề OUTPUT

```
OUTPUT
```

--Sử dụng bảng INSERTED

```
INSERTED.InvoiceNo,
```

```
INSERTED.ProductID,
```

```
INSERTED.Quantity,
```

```
INSERTED.Price
```

--Sử dụng mệnh đề WHERE

```
WHERE Discount = 0
```

```
GO
```

Khi thực phát biểu trong ví dụ trên, bạn có thể tìm thấy danh sách mẫu tin được cập giá trị tại cột Discount như hình 7-14.

	InvoiceNo	ProductID	Quantity	Price
1	SI00000001	P00001	100	10000
2	SI00000003	P00002	100	10000
3	SI00000001	P00002	200	10000
4	SI00000003	P00003	300	10000
5	SI00000004	P00004	100	10000

Hình 7-14: Sử dụng từ khóa OUTPUT.

Trong trường hợp muốn nắm giữ những mẫu tin do mệnh đề OUTPUT kết xuất, bạn có thể sử dụng mệnh đề INTO như phần trình bày trong phát biểu INSERT.

Để làm điều này, bạn tạo bảng dữ liệu dùng để chứa mẫu tin được cập nhật có tên là SalesInvoiceDetailsUpdated như ví dụ 7-28.

Ví dụ 7-28: Khai báo tạo bảng chứa dữ liệu

```
CREATE TABLE SalesInvoiceDetailsUpdated
```

```
(
```

```
OrdinalNumber tinyint NOT NULL,
```

```
InvoiceNo VARCHAR(10) NOT NULL,
```

```
ProductID VARCHAR(10) NOT NULL,
```

```
Quantity DECIMAL DEFAULT 0,
```

```
Price DECIMAL DEFAULT 0,  
Discount DECIMAL DEFAULT 0,  
VATRate DECIMAL DEFAULT 0,  
PRIMARY KEY (OrdinalNumber,  
InvoiceNo, ProductID)  
)  
GO
```

Kế đến, bạn khai báo phát biểu UPDATE để cập nhật cột Discount của mẫu tin có giá trị bằng 10 trong bảng SalesInvoiceDetails rồi thêm những mẫu tin đã cập nhật vào bảng SalesInvoiceDetailsUpdated như ví dụ 7-29.

Ví dụ 7-29: Khai báo lấy mẫu tin được cập nhật

UPDATE SalesInvoiceDetails

SET Discount = 10

--Sử dụng mệnh đề OUTPUT

INSERTED.OrdinalNumber,
INSERTED.InvoiceNo,
INSERTED.ProductID,
INSERTED.Quantity,
INSERTED.Price,
INSERTED.Discount,
INSERTED.VATRate

--Sử dụng mệnh đề INTO

INTO SalesInvoiceDetailsUpdated

WHERE Discount = 0

```
SELECT * FROM SalesInvoiceDetailsUpdated
```

GO

Nếu thực thi phát biểu trong ví dụ trên, bạn có thể tìm thấy danh sách mẫu tin được cập nhật trong bảng SalesInvoiceDetails xuất hiện trong bảng SalesInvoiceDetailsUpdated như hình 7-15.

	OrdinalNumber	InvoiceNo	ProductID	Quantity	Price	Discount	VATRate
1	1	S10000004	P00001	150	15000	10	10
2	1	S10000006	P00002	165	12500	10	10
3	1	S10000009	P00001	15	12500	10	10
4	1	S10000011	P00002	25	10500	10	10
5	2	S10000002	P00002	200	10000	10	10
6	2	S10000004	P00004	120	12000	10	10
7	2	S10000011	P00006	25	12500	10	10
8	4	S10000010	P00004	5	12000	10	10
9	4	S10000013	P00004	1	11500	10	10

Hình 7-15: Sử dụng mẫu tin INTO.

2.5. Phát biểu UPDATE với mệnh đề WITH

Như chúng ta tìm hiểu trong phát biểu SELECT và INSERT, mệnh đề WITH cho phép tạo tập dữ liệu theo tiêu chí cụ thể nào đó trước khi thi hành phát biểu SELECT hay INSERT.

Khi làm việc với phát biểu UPDATE, bạn cũng có thể sử dụng mệnh đề WITH tương tự như đã thao tác trong hai phát biểu SELECT và INSERT.

Giả sử, bạn muốn cập nhật giá bán tăng giá trị thêm 100 cho những sản phẩm đã bán tại thành phố Hà Nội.

Để làm điều này, trước tiên, bạn khai báo phát biểu SELECT để kiểm tra danh sách sản phẩm bán tại thành phố Hà Nội như ví dụ 7-30.

Ví dụ 7-30: Sản phẩm bán tại thành phố Hà Nội

```
Select DISTINCT ProductID
from Customers C, SalesInvoices S,
SalesInvoiceDetails D
Where C.CustomerID = S.CustomerID
AND S.InvoiceNo = D.InvoiceNo
AND C.ProvinceID = 'HAN'
```

Kế đến, bạn khai báo phát biểu UPDATE với mệnh đề WITH như ví dụ 7-31.

Ví dụ 7-31: Khai báo mệnh đề WITH với phát biểu UPDATE

```
WITH SalesInHANOI
AS
{
    Select DISTINCT ProductID
    from Customers C, SalesInvoices S,
    SalesInvoiceDetails D
    Where C.CustomerID = S.CustomerID
    AND S.InvoiceNo = D.InvoiceNo
    AND C.ProvinceID = 'HAN'
}

UPDATE SalesPrices
SET Price = Price + 100
WHERE ProductID in
(
    SELECT ProductID
    FROM SalesInHANOI
)
```

Chẳng hạn, trước đó danh sách bảng giá sản phẩm trong bảng SalesPrices trình bày như hình 7-16.

M® 256

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

ProductID
1 P00002
2 P00003

Hình 7-16: Giá trước khi cập nhật.

Sau khi thực thi phát biểu UPDATE với mệnh đề WITH trong ví dụ 7-31, bảng giá bây giờ trình bày như hình 7-17.

Messages
(2 row(s) affected)

Hình 7-17: Bảng giá sau khi cập nhật.

2.6. Phát biểu UPDATE với mệnh đề WRITE

Mệnh đề Write được sử dụng để thay đổi dữ liệu của các cột có kiểu dữ liệu là varchar(max), nvarchar(max), or varbinary(max).

Bạn có thể sử dụng mệnh đề Write trong phát biểu UPDATE với cú pháp như sau:

```
    .WRITE ( expression, @Offset , @Length )
```

Trong đó expression là giá trị sẽ được cập nhật vào cột dữ liệu, Offset ứng với giá trị bắt đầu và Length là chiều dài chuỗi trong cột dữ liệu sẽ bị thay thế bởi giá trị từ tham số expression.

Giả sử, bạn liệt kê danh sách diễn giải của sản phẩm trong bảng Products bằng phát biểu SELECT như ví dụ 7-32.

Ví dụ 7-32: Khai báo liệt kê diễn giải sản phẩm

```
SELECT ProductID, ProductDescriptionInSecondLanguage
FROM Products
GO
```

Khi thực thi phát biểu trên, diễn giải của sản phẩm trình bày như hình 7-18.

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

257

The screenshot shows the SQL Server Management Studio interface with the 'Results' tab selected. A table is displayed with the following data:

ProductID	ProductDescriptionInSecondLanguage
1	P00001 School bag
2	P00002 School bag for Girl
3	P00003 School bag for Boy
4	P00004 Cloth bag

Hình 7-18: Danh sách diễn giải của sản phẩm.

Để cập nhật cột dữ liệu ProductDescriptionInSecondLanguage bằng mệnh đề WRITE, bạn khai báo như ví dụ 7-33.

Ví dụ 7-33: Khai báo mệnh đề WRITE

```
UPDATE Products
SET ProductDescriptionInSecondLanguage
.WRITE ('Red c', 0, 1)
WHERE ProductID = 'P00004'
GO
```

Khi thực thi phát biểu UPDATE trong ví dụ trên, dữ liệu trong cột ProductDescriptionInSecondLanguage sẽ được thay thế chữ “C” thành “Red c” (tức bắt đầu vị trí thứ 0 đến vị trí thứ 1).

Để kiểm tra dữ liệu, bạn chạy phát biểu SELECT trong ví dụ 7-32, kết quả sẽ được trình bày như hình 7-19.

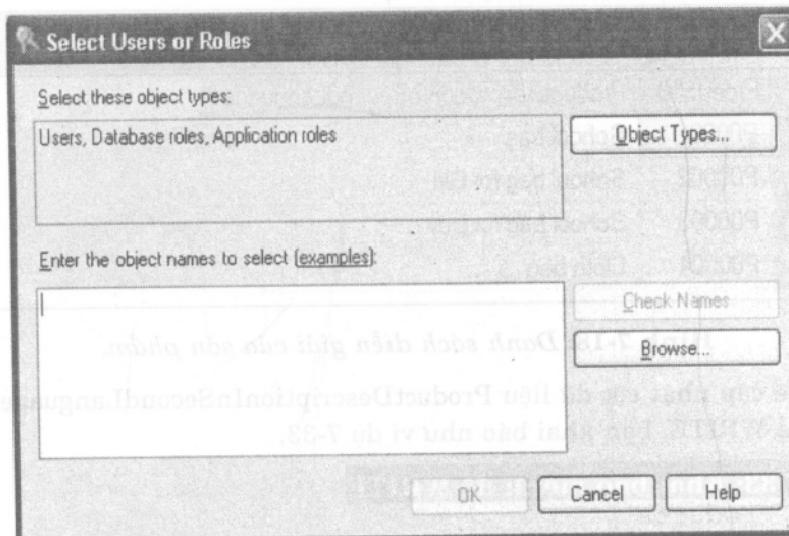
The screenshot shows the SQL Server Management Studio interface with the 'Results' tab selected. A table is displayed with the following data:

ProductID	ProductDescriptionInSecondLanguage
1	P00001 School bag
2	P00002 School bag for Girl
3	P00003 School bag for Boy
4	P00004 Red cloth bag

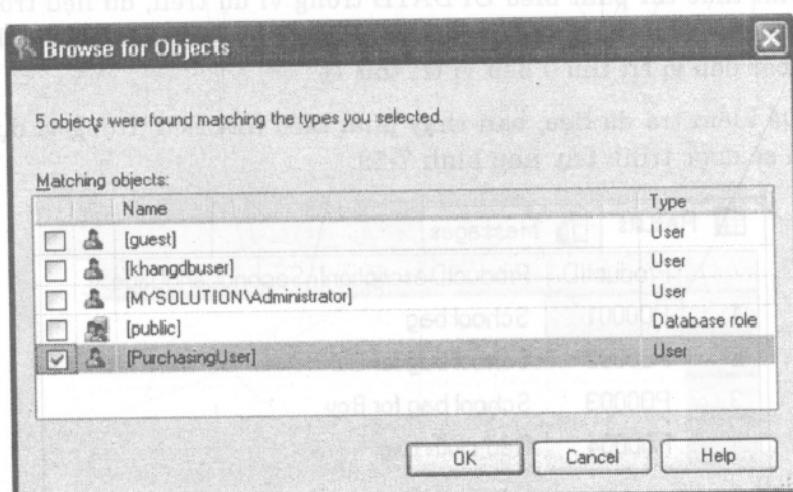
Hình 7-19: Cập nhật bằng mệnh đề WRITE.

Chú ý: Để thực thi phát biểu UPDATE hay cập nhật dữ liệu trong bảng, người sử dụng cần có quyền UPDATE. Chẳng hạn, nếu bạn muốn gán quyền cập nhật dữ liệu bằng MS cho bảng Suppliers, bạn chọn vào nút Add và cửa sổ xuất hiện như hình 7-20.

M® 258

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu**Hình 7-20:** Thêm tài khoản người sử dụng.

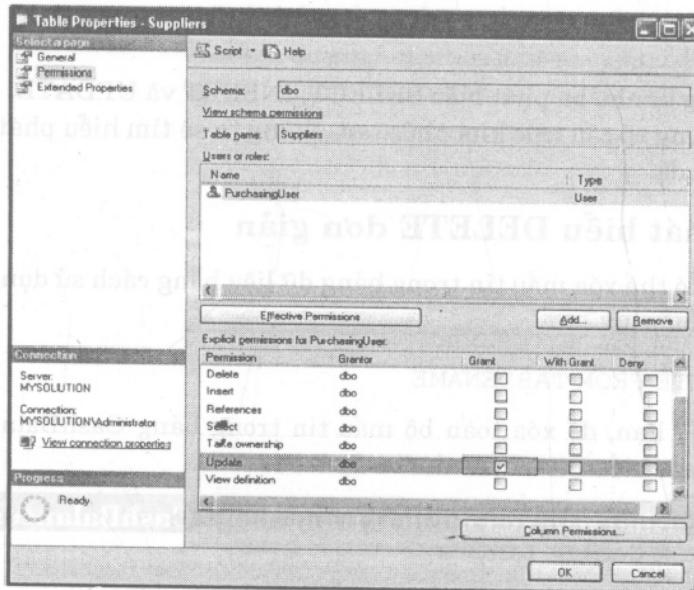
Nhấn nút Browse, cửa sổ liệt kê danh sách người sử dụng xuất hiện như hình 7-21.

**Hình 7-21:** Chọn tên người sử dụng.

Nhấn nút OK, rồi chọn vào CheckBox ứng với quyền UPDATE như hình 7-22.

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

259

**Hình 7-22: Cấp quyền UPDATE.**

3. PHÁT BIỂU DELETE

Phát biểu DELETE cho phép xóa một phần hay toàn bộ dữ liệu trong bảng có hay không dựa trên tiêu chí lọc dữ liệu.

Để sử dụng phát biểu DELETE, bạn sử dụng cú pháp tương tự như sau:

```
[ WITH <common_table_expression> [ ,...n ] ]
DELETE
[ TOP ( expression ) [ PERCENT ] ]
[ FROM ]
{ <object> | rowset_function_limited
[ WITH ( <table_hint_limited> [,...n] ) ]
[ <OUTPUT Clause> ]
[ FROM <table_source> [ ,...n ] ]
[ WHERE { <search_condition>
| { [ CURRENT OF
{ { [ GLOBAL ] cursor_name }
| cursor_variable_name
}
]
}
}
]
]
```

M® 260**Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu**

```
[ OPTION ( <Query Hint> [ ,...n ] ) ]
[ ; ]
```

Tương tự như ba phát biểu SELECT, INSERT và UPDATE, phát biểu DELETE cũng có cấu trúc khá phức tạp, chúng ta sẽ tìm hiểu phát biểu này qua từng ví dụ.

3.1. Phát biểu DELETE đơn giản

Bạn có thể xóa mẫu tin trong bảng dữ liệu bằng cách sử dụng cú pháp đơn giản như sau:

```
DELETE FROM TABLENAME
```

Chẳng hạn, để xóa toàn bộ mẫu tin trong bảng CashBalances, bạn khai báo phát biểu DELETE như ví dụ 7-34.

Ví dụ 7-34: Khai báo xóa dữ liệu trong bảng CashBalances

```
DELETE FROM CashBalances
GO
```

3.2. Phát biểu DELETE với mệnh đề WHERE

Tương tự như các phát biểu SQL dạng DML khác, mệnh đề WHERE cho phép bạn lọc dữ liệu theo tiêu chí nào đó để xóa. Chẳng hạn, bạn khai báo phát biểu DELETE để xóa mẫu tin trong bảng PurchaseInvoices tại cột InvoiceBatchNo có giá trị 'PBI005' thì khai báo như ví dụ 7-35.

Ví dụ 7-35: Khai báo mệnh đề WHERE

```
DELETE FROM PurchaseInvoices
WHERE InvoiceBatchNo = 'PBI005'
GO
```

Bạn có thể sử dụng mệnh đề WHERE với phép toán IN để trích lọc dữ liệu trong bảng dựa vào câu truy vấn con.

Ví dụ, bạn khai báo phát biểu DELETE với phép toán IN để xóa những mẫu tin trong bảng ExportDetails có mã sản phẩm bán tại tỉnh Đồng Nai (ứng với mã tỉnh thành là DNA).

Ví dụ 7-36: Khai báo phát biểu DELETE với phép toán IN

```
DELETE FROM ExportDetails
WHERE ProductID IN
(
    Select DISTINCT ProductID
    from Customers C, SalesInvoices S,
         SalesInvoiceDetails D
    Where C.CustomerID = S.CustomerID
```

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

261

```

    AND S.InvoiceNo = D.InvoiceNo
    AND C.ProvinceID = 'DNA'
)
GO

```

3.3. Phát biểu DELETE với mệnh đề TOP

Như đã trình bày trong phát biểu SELECT, INSERT và UPDATE, bạn sử dụng mệnh đề TOP để lấy ra số mẫu tin cụ thể hay tính theo phần trăm dựa vào tổng số mẫu tin trong bảng.

Bằng cách sử dụng mệnh đề TOP, bạn cũng có thể chỉ định phát biểu DELETE chỉ xóa những mẫu tin nằm trong khoảng mẫu tin cho phép.

Chẳng hạn, bạn khai báo để xóa 10 mẫu tin trong bảng ExportDetails có mã sản phẩm bán tại tỉnh Bình Dương (ứng với mã tỉnh thành là BDU) như ví dụ 7-37.

Ví dụ 7-37: Khai báo phát biểu DELETE với mệnh đề TOP

```

DELETE TOP(10)
FROM ExportDetails
WHERE ProductID IN
(
    Select DISTINCT ProductID
    from Customers C, SalesInvoices S,
        SalesInvoiceDetails D
    Where C.CustomerID = S.CustomerID
        AND S.InvoiceNo = D.InvoiceNo
        AND C.ProvinceID = 'BDU'
)
GO

```

Chú ý: Mặc dù mệnh đề WHERE có thể tìm thấy N mẫu tin ($N > 10$), nhưng mệnh đề TOP chỉ cho phép xóa 10 mẫu tin đầu tiên.

Ngoài ra, bạn cũng có thể chỉ định mệnh đề TOP với số thập phân kèm theo từ khóa PERCENT ứng với phần trăm cần xóa tương tự như ví dụ 7-38.

Ví dụ 7-38: Khai báo phát biểu DELETE với mệnh đề TOP

```

DELETE TOP(1.5) PERCENT
FROM ExportDetails
WHERE ProductID IN
(
    Select DISTINCT ProductID
    from Customers C, SalesInvoices S,
        SalesInvoiceDetails D
    Where C.CustomerID = S.CustomerID
        AND S.InvoiceNo = D.InvoiceNo
        AND C.ProvinceID = 'BDU'
)
GO

```

3.4. Phát biểu DELETE với mệnh đề OUTPUT

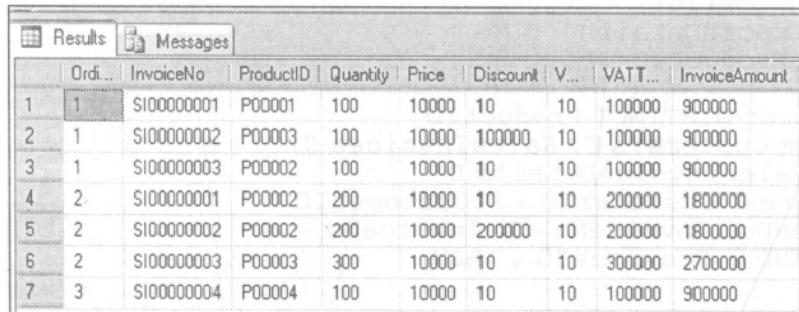
Mục đích của mệnh đề OUTPUT cho phép bạn kết xuất ra trình thực thi phát biểu SQL, chính vì vậy khi sử dụng mệnh đề này trong phát biểu DELETE, bạn có thể nhận được danh sách mẫu tin đã bị xóa.

Giả sử, bạn khai báo phát biểu INSERT để thêm danh sách mẫu tin vào bảng SalesInvoiceDetailsForBackup từ bảng SalesInvoiceDetails và sử dụng phát biểu SELECT để xem dữ liệu như ví dụ 7-39.

Ví dụ 7-39: Khai báo tạo dữ liệu mẫu để xóa

```
INSERT INTO SalesInvoiceDetailsForBackup
SELECT * FROM SalesInvoiceDetails
GO
Select * FROM SalesInvoiceDetailsForBackup
GO
```

Khi thực thi phát biểu SQL trong ví dụ trên, chúng ta sẽ có danh sách mẫu tin như hình 7-23.



	Ordinal	InvoiceNo	ProductID	Quantity	Price	Discount	VAT Tax Rate	VAT Tax Amount	InvoiceAmount
1	1	SI00000001	P00001	100	10000	10	10	100000	900000
2	1	SI00000002	P00003	100	10000	100000	10	100000	900000
3	1	SI00000003	P00002	100	10000	10	10	100000	900000
4	2	SI00000001	P00002	200	10000	10	10	200000	1800000
5	2	SI00000002	P00002	200	10000	200000	10	200000	1800000
6	2	SI00000003	P00003	300	10000	10	10	300000	2700000
7	3	SI00000004	P00004	100	10000	10	10	100000	900000

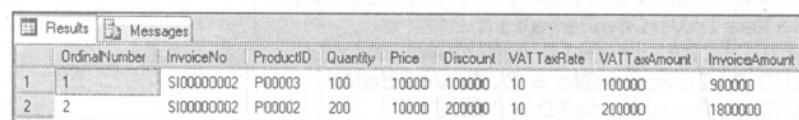
Hình 7-23: Tạo dữ liệu để xóa.

Để xóa những mẫu tin có giá trị trong cột Discount lớn hơn 1000, bạn khai báo phát biểu DELETE và sử dụng mệnh đề OUTPUT để xem danh sách mẫu tin đã bị xóa như ví dụ 7-40.

Ví dụ 7-40: Khai báo xóa mẫu tin với mệnh đề OUTPUT

```
DELETE FROM SalesInvoiceDetailsForBackup
OUTPUT DELETED.*
WHERE Discount > 1000
GO
```

Khi thực thi phát biểu trong ví dụ trên, bạn sẽ nhận được kết quả trả về như hình 7-24.



	OrdinalNumber	InvoiceNo	ProductID	Quantity	Price	Discount	VAT Tax Rate	VAT Tax Amount	InvoiceAmount
1	1	SI00000002	P00003	100	10000	100000	10	100000	900000
2	2	SI00000002	P00002	200	10000	200000	10	200000	1800000

Hình 7-24: Sử dụng mệnh đề OUTPUT.

Chương 7: Phát biểu T-SQL cơ bản dang xử lý dữ liệu

263 M

Chú ý: Khi làm việc với cơ sở dữ liệu SQL Server 2000, bạn thường sử dụng phát biểu INSERT để thêm mẫu tin từ bảng A sang bảng B để sao lưu trước khi xóa chúng khỏi bảng B.

Khi làm việc với cơ sở dữ liệu SQL Server 2005, bạn có thể sử dụng mệnh đề OUTPUT để sao chép những mẩu tin vừa xóa thành công vào bảng khác.

Để thực hiện ý định này, trước tiên bạn phải bảo đảm tồn tại mẫu tin trong bảng SalesInvoiceDetailsForBackup để xóa.

Kế đến, bạn xóa danh sách mẫu tin tồn tại trong bảng PurchaseInvoiceDetailsForBackup, bảng này dùng để lưu những mẫu tin xóa trong bảng SalesInvoiceDetailsForBackup.

Sau đó, khai báo phát biểu DELETE với mệnh đề OUTPUT và INTO như ví dụ 7-41.

Ví dụ 7-41: Khai báo sử dụng mêm đề INTO

```
DELETE FROM SalesInvoiceDetailsForBackup  
OUTPUT DELETED.*  
INTO PurchaseInvoiceDetailsForBackup  
GO
```

Khi thực thi phát biểu **DELETE** trên, bạn có thể kiểm tra mẫu tin thêm vào bảng **PurchaseInvoiceDetailsForBackup** bằng cách thực thi phát biểu **SELECT**, kết quả trình bày như hình 7-25.

	OrdinalNumber	InvoiceNo	ProductID	Quantity	Price	Discount	VATTaxRate	VATTaxAmount	PurchaseAmount
1	1	S100000001	P00001	100	10000	20	10	100000	900000
2	1	S100000002	P00003	100	10000	100000	10	100000	900000
3	1	S100000003	P00002	100	10000	20	10	100000	900000
4	2	S100000001	P00002	200	10000	20	10	200000	1800000
5	2	S100000002	P00002	200	10000	200000	10	200000	1800000
6	2	S100000003	P00003	300	10000	20	10	300000	2700000
7	3	S100000004	P00004	100	10000	20	10	100000	900000

Hình 7-25: Lưu mẫu tin đã xóa.

3.5. Phát biểu DELETE với mệnh đề WITH

Tương tự như những phát biểu DML khác, bạn cũng có thể sử dụng mệnh đề WITH với phát biểu DELETE bằng khai báo tương tự như ví dụ 7-42.

Ví dụ 7-42: Khai báo sử dụng mảng để WITH

WITH RecordsForDelete AS

 264

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

```
(SELECT * FROM PurchaseInvoiceDetailsForBackup  
WHERE Discount>1000  
)  
DELETE FROM RecordsToDelete  
GO
```

Giả sử, chúng ta có danh sách mẫu tin trong bảng PurchaseInvoiceDetailsForBackup như hình 7-26.

	OrdinalNumber	InvoiceNo	ProductID	Quantity	Price	Discount	VATTaxRate	VATTaxAmount	PurchaseAmount
1	1	S100000001	P00001	100	10000	20	10	100000	900000
2	1	S100000002	P00003	100	10000	100000	10	100000	900000
3	1	S100000003	P00002	100	10000	20	10	100000	900000
4	2	S100000001	P00002	200	10000	20	10	200000	1800000
5	2	S100000002	P00002	200	10000	200000	10	200000	1800000
6	2	S100000003	P00003	300	10000	20	10	300000	2700000
7	3	S100000004	P00004	100	10000	20	10	100000	900000

Hình 7-26: Danh sách mẫu tin trong bảng PurchaseInvoiceDetailsForBackup.

Thực thi phát biểu DELETE với mệnh đề WITH vừa trình bày ở trên thì kết quả trình bày như hình 7-27.

Messages									
id	text	from	to	date	subject	status	type	size	path
1	Test message 1	user1	user2	2023-10-01 10:00:00	Test Subject 1	Pending	Text	100B	C:\Users\user1\Documents\test1.txt
2	Test message 2	user2	user1	2023-10-01 10:15:00	Test Subject 2	Delivered	Text	100B	C:\Users\user2\Documents\test2.txt

Hình 7-27: Xóa mẫu tin.

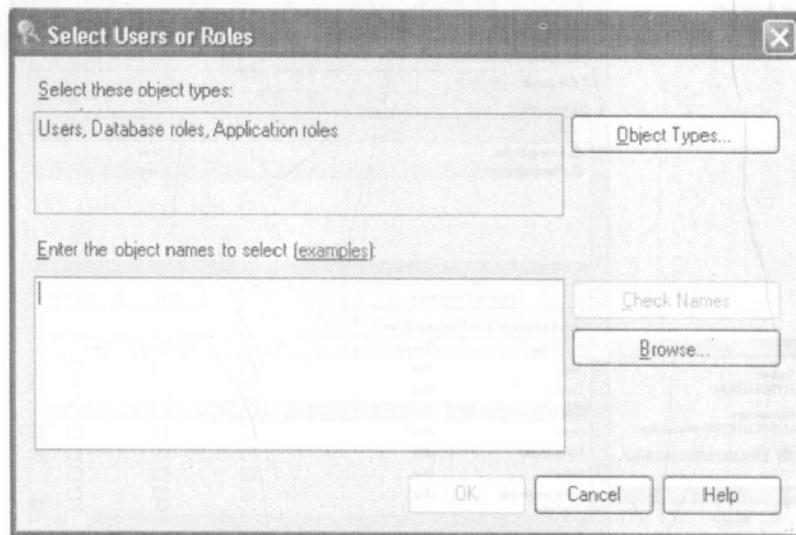
Bằng cách thực thi phát biểu SELECT để kiểm tra danh sách mẫu tin đã bị xóa như hình 7-28.

	OrdinalNumber	InvoiceNo	ProductID	Quantity	Price	Discount	VATTaxRate	VATTaxAmount	PurchaseAmount
1	1	S100000001	P00001	100	10000	20	10	100000	900000
2	1	S100000003	P00002	100	10000	20	10	100000	900000
3	2	S100000001	P00002	200	10000	20	10	200000	1800000
4	2	S100000003	P00003	300	10000	20	10	300000	2700000
5	3	S100000004	P00004	100	10000	20	10	100000	900000

Hình 7-28: Kiểm tra mẫu tin đã xóa.

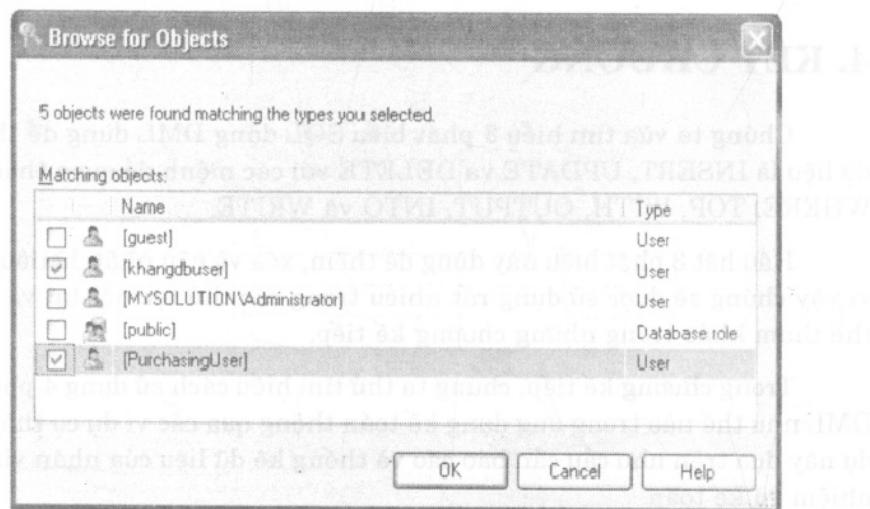
Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu 265

Chú ý: Để thực thi phát biểu DELETE hay xóa mẫu tin trong bảng, người sử dụng cần có quyền DELETE. Chẳng hạn, nếu bạn muốn gán quyền xóa mẫu tin bằng MS cho bảng Products, bạn chọn vào nút Add và cửa sổ xuất hiện như hình 7-29.



Hình 7-29: Thêm tài khoản người sử dụng.

Nhấn nút Browse cửa sổ liệt kê danh sách người sử dụng xuất hiện như hình 7-30.

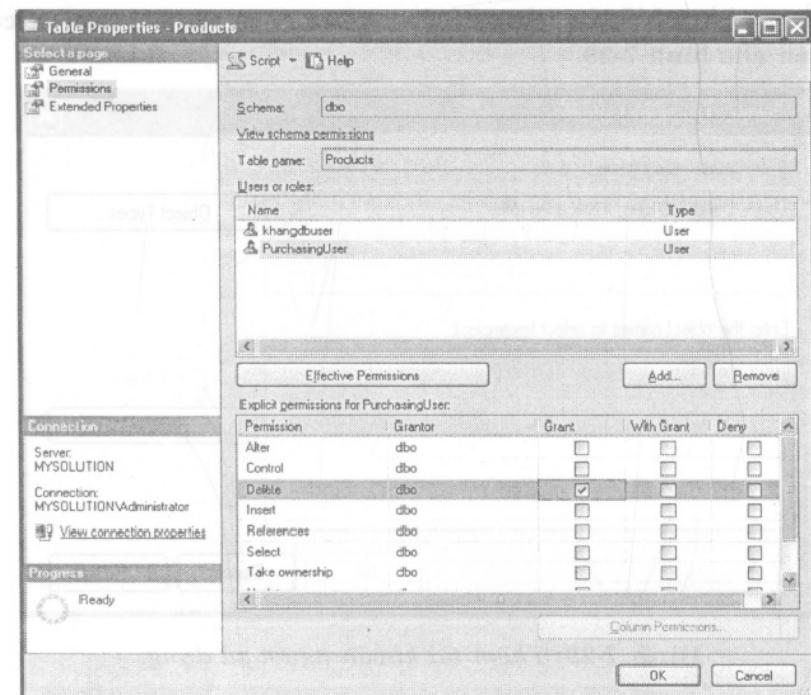


Hình 7-30: Chọn tên người sử dụng.

M® 266

Chương 7: Phát biểu T-SQL cơ bản dạng xử lý dữ liệu

Nhấn nút OK, rồi chọn vào CheckBox ứng với quyền DELETE như hình 7-31.



Hình 7-31: Cấp quyền DELETE.

4. KẾT CHƯƠNG

Chúng ta vừa tìm hiểu 3 phát biểu SQL dạng DML dùng để thay đổi dữ liệu là INSERT, UPDATE và DELETE với các mệnh đề quen thuộc như: WHERE, TOP, WITH, OUTPUT, INTO và WRITE.

Hầu hết 3 phát biểu này dùng để thêm, xóa và cập nhật dữ liệu, chính vì vậy chúng sẽ được sử dụng rất nhiều trong các thủ tục nội tại và bạn có thể tham khảo trong những chương kế tiếp.

Trong chương kế tiếp, chúng ta thử tìm hiểu cách sử dụng 4 phát biểu DML như thế nào trong ứng dụng kế toán thông qua các ví dụ cụ thể, các ví dụ này dựa trên nhu cầu cần báo cáo và thống kê dữ liệu của nhân viên làm nhiệm vụ kế toán.

Ứng dụng:

SỬ DỤNG PHÁT BIỂU SQL TRONG ỨNG DỤNG

Tóm tắt ứng dụng

Chúng ta đã tìm hiểu cấu trúc cơ sở dữ liệu trong chương 2, cách làm việc với trình điều khiển cơ sở dữ liệu Microsoft SQL Server Management Studio trong chương 3 rồi tìm hiểu thành phần chính của cơ sở dữ liệu trong chương 4.

Tiếp theo sau chương 4, bạn đã tìm hiểu phát biểu CREATE và ALTER trong chương 5 và phát biểu SELECT trong chương 6.

Sau khi tìm hiểu phát biểu SELECT với hầu hết các mệnh đề liên quan, bạn lại tiếp tục tìm hiểu các phát biểu INSERT, UPDATE và DELETE trong chương 7.

Trong chương này, chúng ta sẽ tìm hiểu cách sử dụng phát biểu SELECT trong ứng dụng kế toán dính kèm thông qua từng yêu cầu cụ thể.

Các vấn đề chính sẽ được đề cập:

- ✓ Phân kế toán công nợ phải thu.
- ✓ Phân kế toán công nợ phải trả.
- ✓ Phân kế toán tổng hợp.
- ✓ Phân kế toán xuất nhập tồn.

1. PHÂN KẾ TOÁN CÔNG NỢ PHẢI THU

Như trình bày trong những chương trước, phát biểu SELECT được sử dụng nhiều nhất so với phát biểu INSERT, UPDATE và DELETE trong ứng

dụng, chúng ta sẽ sử dụng phát biểu này với các yêu cầu của người làm quản lý cũng như người làm kế toán.

Để thực hiện phát biểu SELECT cho phần công nợ phải thu, trước tiên chúng ta giả sử dữ liệu đang tồn tại của bảng liên quan trong cơ sở dữ liệu.

Giả sử, chúng ta có danh sách mẫu tin trong bảng Customers với ví dụ UD-1.

Ví dụ UD-1: Danh sách khách hàng

```
SELECT CustomerID, CompanyNameInVietnamese  
FROM Customers  
GO
```

Danh sách khách hàng trong bảng Customers trình bày như hình UD-1.

	CustomerID	CompanyNameInVietnamese
1	A0001	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam
2	A0002	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam
3	A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam
4	A0004	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam
5	A0005	Công ty Cổ phần Suzumi Vietnam
6	A0006	Tập đoàn UCIA USA
7	A0007	Công ty Đa quốc gia UFCA

Hình UD-1: Danh sách khách hàng.

Liên quan đến phần doanh thu bán hàng, chúng ta cũng xem danh sách loại hóa đơn với phát biểu như ví dụ UD-2.

Ví dụ UD-2: Liệt kê danh sách loại hóa đơn bán hàng

```
SELECT * FROM SalesInvoiceTypes  
GO
```

Khi thực thi phát biểu trên, danh sách loại hóa đơn được liệt kê như hình UD-2.

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

269

The screenshot shows a table titled 'Results' with three columns: 'InvoiceTypeID', 'InvoiceTypeNamelnVietnamese', 'Invoice...', and 'ShowInv...'. There are three rows of data:

InvoiceTypeID	InvoiceTypeNamelnVietnamese	Invoice...	ShowInv...
1	SI1 Hoá đơn bán hàng	NULL	1
2	SI2 Hoá đơn bán hàng là Bán thành phẩm	NULL	1
3	SI3 Hoá đơn bán hàng khuyến mãi	NULL	1

Hình UD-2: Danh sách loại hóa đơn bán hàng.

Chúng ta cũng có thể liệt kê danh sách mẫu tin trong bảng SalesInvoices bằng phát biểu trong ví dụ UD-3.

Ví dụ UD-3: Liệt kê danh sách hóa đơn bán hàng

```
SELECT * FROM SalesInvoices
GO
```

Khi thực thi phát biểu trên, danh sách hóa đơn bán hàng trình bày như hình UD-3.

The screenshot shows a table titled 'Results' with six columns: 'InvoiceNo', 'InvoiceBatch...', 'DueDate', 'InvoiceType...', 'CustomerID', and 'CustomerID'. There are 13 rows of data:

InvoiceNo	InvoiceBatch...	DueDate	InvoiceType...	CustomerID
SI00000001	SBI001	2007-10-10 00:00:00	SI1	A0001
SI00000002	SBI001	2007-10-11 00:00:00	SI1	A0002
SI00000003	SBI001	2007-10-12 00:00:00	SI1	A0003
SI00000004	SBI001	2007-10-13 00:00:00	SI1	A0001
SI00000005	SBI002	2007-10-14 00:00:00	SI1	A0004
SI00000006	SBI002	2007-10-14 00:00:00	SI1	A0005
SI00000007	SBI003	2007-10-17 00:00:00	SI1	A0006
SI00000008	SBI003	2007-10-17 00:00:00	SI1	A0007
SI00000009	SBI004	2007-10-18 00:00:00	SI1	A0008
SI00000010	SBI005	2007-10-19 00:00:00	SI1	A0001
SI00000011	SBI005	2007-10-19 00:00:00	SI1	A0002
SI00000012	SBI006	2007-10-20 00:00:00	SI1	A0001
SI00000013	SBI006	2007-10-20 00:00:00	SI1	A0005

Hình UD-3: Danh sách hóa đơn bán hàng.

Tương tự như vậy, chúng ta cũng có danh sách mẫu tin chi tiết hóa đơn bán hàng trong bảng SalesInvoiceDetails bằng ví dụ UD-4.

Ví dụ UD-4: Liệt kê danh sách hóa đơn bán hàng chi tiết

```
SELECT * FROM SalesInvoiceDetails
GO
```

M® 270

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

Chú ý: Do không gian trình bày của cửa sổ Query có giới hạn, một số mẫu tin không thể nhìn thấy trên hình trình bày, thay vào đó bạn có thể kiểm tra dữ liệu trong cơ sở dữ liệu đính kèm.

Thực thi phát biểu trên, bạn có thể tìm thấy danh sách mẫu tin trong bảng SalesInvoiceDetails như hình UD-4.

	OrdinalNumber	InvoiceNo	ProductID	Quantity	Price	Discount	VATRate
1	1	S100000001	P00001	100	10000	50000	10
2	1	S100000002	P00003	100	10000	100000	10
3	1	S100000003	P00002	100	10000	50000	10
4	1	S100000004	P00001	150	15000	0	10
5	1	S100000005	P00001	250	15000	50000	10
6	1	S100000006	P00002	165	12500	0	10
7	1	S100000007	P00005	125	13500	60000	10
8	1	S100000008	P00006	35	14500	30000	10
9	1	S100000009	P00001	15	12500	0	10
10	1	S100000010	P00002	35	12500	30000	10
11	1	S100000011	P00002	25	10500	0	10
12	1	S100000012	P00002	25	10500	30000	10
13	1	S100000013	P00004	5	12500	30000	10
14	2	S100000001	P00002	200	10000	50000	10
15	2	S100000002	P00002	200	10000	0	10
16	2	S100000003	P00003	300	10000	50000	10
17	2	S100000004	P00004	120	12000	0	10
18	2	S100000005	P00003	120	12500	55000	10
19	2	S100000007	P00006	25	14500	10000	10
20	2	S100000008	P00003	35	12500	30000	10
21	2	S100000009	P00003	35	12500	15000	10
22	2	S100000010	P00003	35	12500	30000	10
23	2	S100000011	P00006	25	12500	0	10
24	2	S100000012	P00004	5	12500	30000	10
25	2	S100000013	P00005	5	12500	30000	10
26	3	S100000009	P00004	5	12500	15000	10
27	3	S100000010	P00004	35	12500	30000	10
28	3	S100000012	P00006	5	12500	20000	10

Query executed successfully.

Hình UD-4: Danh sách mẫu tin chi tiết hóa đơn bán hàng.

1.1. Doanh thu bán hàng theo thời gian

Với yêu cầu thống kê doanh thu bán hàng theo ngày hiện hành, bạn sử dụng phát biểu SELECT với mệnh đề INNER JOIN, GROUP BY theo cột dữ liệu DueDate của hai bảng dữ liệu SalesInvoices và SalesInvoiceDetails.

1.1.1. Doanh thu bán hàng theo ngày

Bằng cách sử dụng mệnh đề WHERE hay INNER JOIN và GROUP BY để lấy ra dữ liệu từ hai bảng SalesInvoices và SalesInvoiceDetails như ví dụ UD-5.

Ví dụ UD-5: Doanh thu theo ngày trong tháng

```
SELECT DueDate,
--Sử dụng hàm SUM
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
--Sử dụng mệnh đề WHERE thay vì mệnh đề INNER JOIN
FROM SalesInvoices S, SalesInvoiceDetails D
WHERE S.InvoiceNo = D.InvoiceNo
--Sử dụng mệnh đề GROUP BY
GROUP BY DueDate
GO
```

Khi thực thi phát biểu trên, danh sách mẫu tin được trích lọc và tính toán theo thời gian từ hai bảng SalesInvoices và SalesInvoiceDetails trình bày như hình UD-5.

	DueDate	SalesAmount
1	2007-10-10 00:00:00	3200000.000000
2	2007-10-11 00:00:00	3200000.000000
3	2007-10-12 00:00:00	4300000.000000
4	2007-10-13 00:00:00	4059000.000000
5	2007-10-14 00:00:00	2268750.000000
6	2007-10-14 10:00:00	5670000.000000
7	2007-10-17 00:00:00	3164500.000000
8	2007-10-18 00:00:00	726250.000000
9	2007-10-19 00:00:00	2052250.000000
10	2007-10-20 00:00:00	426400.000000

Hình UD-5: Doanh thu theo thời gian.

Chú ý: Nếu cột DueDate xuất hiện giá trị thời gian, mệnh đề GROUP BY sẽ không thể nhóm theo ngày, bạn có thể tìm thấy hai mẫu tin cùng ngày 2007-10-14 nhưng khác giờ trong hình UD-5.

Để nhóm dữ liệu theo ngày, bạn có thể sử dụng hàm CONVERT để tính theo ngày như ví dụ UD-6.

Ví dụ UD-6: Khai báo sử dụng hàm CONVERT

--Sử dụng hàm CONVERT

```
-Sử dụng hàm CONVERT  
SELECT Convert(char(10), DueDate, 103) AS SalesDate,  
SUM(Quantity*Price*(1+VATRate/100)-Discount) As  
SalesAmount
```

--Sử dụng mệnh đề WHERE thay vì mệnh đề INNER JOIN

```
--Sử dụng menu de MySQL này và menu de MySQL bên  
FROM SalesInvoices S, SalesInvoiceDetails D  
WHERE S.InvoiceNo = D.InvoiceNo
```

--Sử dụng mệnh đề GROUP BY với hàm CONVERT

GROUP BY Convert (char(10) ,DueDate, 103)
GO

Khi thực thi phát biểu trên, danh sách mẫu tin được trích lọc và tính toán theo ngày từ hai bảng SalesInvoices và SalesInvoiceDetails trình bày như hình UD-6.

	SalesDate	SalesAmount
1	10/10/2007	3200000.000000
2	11/10/2007	3200000.000000
3	12/10/2007	4300000.000000
4	13/10/2007	4059000.000000
5	14/10/2007	7938750.000000
6	17/10/2007	3164500.000000
7	18/10/2007	726250.000000
8	19/10/2007	2052250.000000
9	20/10/2007	426400.000000

Hình UD-6: Doanh thu theo ngày.

1.1.2. Doanh thu bán hàng theo tuần hiện tại

Bằng cách sử dụng hàm DATEPART để tính tuần cho ngày của hóa đơn và hàm GETDATE để tính tuần cho ngày hiện hành, bạn có thể liệt kê danh sách doanh thu bán hàng trong tuần hiện hành như ví dụ UD-7.

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

273

Ví dụ UD-7: Doanh thu bán hàng theo tuần hiện hành

```
SELECT Convert(char(10), DueDate, 103) AS SalesDate,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
FROM SalesInvoices S, SalesInvoiceDetails D
WHERE S.InvoiceNo = D.InvoiceNo
```

--Sử dụng hàm DATEPART và GETDATE
AND DATEPART(ww, DueDate) = DATEPART(ww, GETDATE())
GROUP BY Convert(char(10), DueDate, 103)
GO

Khi thực thi phát biểu trên, doanh thu bán hàng trong tuần hiện hành sẽ trình bày như hình UD-7.

	SalesDate	SalesAmount
1	10/10/2007	3200000.000000
2	11/10/2007	3200000.000000
3	12/10/2007	4300000.000000
4	13/10/2007	4059000.000000

Hình UD-7: Doanh thu bán hàng trong tuần hiện hành.

1.1.3. Doanh thu bán hàng theo tuần

Tương tự như trên, bạn có thể liệt kê danh sách doanh thu bán hàng theo tuần trong tháng bằng cách sử dụng hàm DATEPART như ví dụ UD-8.

Ví dụ UD-8: Doanh thu bán hàng theo tuần trong tháng

--Sử dụng hàm DATEPART
SELECT DATEPART(ww, DueDate) AS SalesWeek,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
FROM SalesInvoices S, SalesInvoiceDetails D
WHERE S.InvoiceNo = D.InvoiceNo

--Sử dụng hàm DATEPART với mệnh đề GROUP BY
GROUP BY DATEPART(ww, DueDate)
GO

Khi thực thi phát biểu trên, doanh thu bán hàng theo tuần trong tháng sẽ trình bày như hình UD-8.

Chú ý: Dữ liệu trong các bảng SalesInvoiceBatch, SalesInvoices và SalesInvoiceDetails chỉ chứa nghiệp vụ trong tháng hiện hành, do chúng ta sử dụng cơ chế đóng sổ (Close Month).

M® 274**Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng**

	SalesWeek	SalesAmount
1	41	14759000.000000
2	42	14308150.000000

Hình UD-8: Doanh thu bán hàng theo tuần.**1.2. Doanh thu bán hàng theo hóa đơn**

Nếu như bạn sử dụng mệnh đề GROUP BY dựa vào cột DueDate để liệt kê doanh thu bán hàng theo thời gian, trong trường hợp liệt kê doanh thu bán hàng theo từng mã số hóa đơn thì bạn khai báo ví dụ UD-9.

Ví dụ UD-9: Liệt kê doanh thu bán hàng theo mã hóa đơn

```
--Liệt kê hai cột InvoiceNo, DueDate
SELECT S.InvoiceNo, DueDate,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
FROM SalesInvoices S, SalesInvoiceDetails D
WHERE S.InvoiceNo = D.InvoiceNo

--Sử dụng mệnh đề GROUP BY theo hai cột InvoiceNo, DueDate
GROUP BY S.InvoiceNo, DueDate
GO
```

Khi thực thi phát biểu trên, doanh thu bán hàng theo mã hóa đơn sẽ trình bày như hình UD-9.

	InvoiceNo	DueDate	SalesAmount
1	S100000001	2007-10-10 00:00:00	3200000.000000
2	S100000002	2007-10-11 00:00:00	3200000.000000
3	S100000003	2007-10-12 00:00:00	4300000.000000
4	S100000004	2007-10-13 00:00:00	4059000.000000
5	S100000005	2007-10-14 10:00:00	5670000.000000
6	S100000006	2007-10-14 00:00:00	2268750.000000
7	S100000007	2007-10-17 00:00:00	2185000.000000
8	S100000008	2007-10-17 00:00:00	979500.000000
9	S100000009	2007-10-18 00:00:00	726250.000000
10	S100000010	2007-10-19 00:00:00	1419750.000000
11	S100000011	2007-10-19 00:00:00	632500.000000
12	S100000012	2007-10-20 00:00:00	297500.000000
13	S100000013	2007-10-20 00:00:00	128900.000000

Hình UD-9: Doanh thu bán hàng theo mã hóa đơn.

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

275 M®

1.3. Doanh thu bán hàng theo sản phẩm

Báo cáo doanh thu theo sản phẩm cho phép nhà quản lý biết được sản phẩm nào có kết quả kinh doanh tốt nhất.

Dể làm điều này, trước tiên bạn liệt kê xem danh sách mẫu tin trong bảng Products bằng phát biểu như ví dụ `UD-10`.

Ví dụ UD-10: Danh sách sản phẩm

```
SELECT ProductID, CategoryID,
ProductNameInVietnamese
FROM Products
GO
```

Khi thực thi phát biểu trên, danh sách sản phẩm được trình bày như hình UD-10.

	ProductID	CategoryID	ProductNameInVietnamese
1	P00001	1	Túi xách
2	P00002	1	Túi xách dùng cho học sinh nữ
3	P00003	1	Túi xách dùng cho học sinh nam
4	P00004	1	Túi áo mưa
5	P00005	2	Túi xách dùng cho Máy tính
6	P00006	2	Túi xách dùng cho Điện thoại di động
7	P00007	2	Túi xách dùng cho PC

Hình UD-10: Danh sách sản phẩm.

Kế đến, với cách sử dụng mệnh đề GROUP BY như những ví dụ trên, bạn có thể sử dụng mệnh đề này để nhóm dữ liệu giữa hai bảng Products và SalesInvoiceDetails như ví dụ UD-11.

Ví dụ UD-11: Doanh thu bán hàng theo sản phẩm

--Liệt kê ba cột dữ liệu

```
    Lấy ra các giá trị  
SELECT P.ProductID, ProductNameInVietnamese,  
SUM(Quantity*Price*(1+VATRate/100)-Discount) As  
SalesAmount  
FROM Products P, SalesInvoiceDetails D  
WHERE P.ProductID = D.ProductID
```

--Sử dụng mệnh đề GROUP BY

```
GROUP BY P.ProductID, ProductNameInVietnamese  
GO
```

M® 276

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

Khi thực thi phát biểu trên, doanh thu bán hàng theo sản phẩm trong tháng hiện hành sẽ trình bày như hình UD-11.

	ProductID	ProductNameInVietnamese	SalesAmount
1	P00001	Túi xách	7806250.000000
2	P00002	Túi xách dùng cho học sinh nữ	8667500.000000
3	P00003	Túi xách dùng cho học sinh nam	7213750.000000
4	P00004	Túi áo mưa	2245150.000000
5	P00005	Túi xách dùng cho Máy tính	1835000.000000
6	P00006	Túi xách dùng cho Điện thoại di động	1299500.000000

Hình UD-11: Doanh thu bán hàng theo sản phẩm.

Chú ý: Trong trường hợp muốn liệt kê toàn bộ danh sách sản phẩm, bạn phải sử dụng mệnh đề LEFT JOIN thay vì WHERE. Để làm điều này, bạn khai báo ví dụ trên thành ví dụ UD-12.

Ví dụ UD-12: Khai báo mêtô đề LEFT JOIN

```
SELECT P.ProductID, ProductNameInVietnamese,
       SUM(Quantity*Price*(1+VATRate/100)-Discount) As SalesAmount
```

--Sử dụng mệnh đề LEFT JOIN

```
--Sử dụng mệnh đề LEFT JOIN  
FROM Products P LEFT JOIN SalesInvoiceDetails D  
ON P.ProductID = D.ProductID
```

--Sử dụng mệnh đề GROUP BY

```
--Sử dụng lệnh ac GROUP BY  
GROUP BY P.ProductID, ProductNameInVietnamese  
GO
```

Khi thực thi phát biểu trên, doanh thu bán hàng theo sản phẩm trong tháng hiện hành sẽ trình bày như hình UD-12.

	ProductID	ProductNameInVietnamese	SalesAmount
1	P00001	Túi xách	7806250.000000
2	P00002	Túi xách dùng cho học sinh nữ	8667500.000000
3	P00003	Túi xách dùng cho học sinh nam	7213750.000000
4	P00004	Túi áo mưa	2245150.000000
5	P00005	Túi xách dùng cho Máy tính	1835000.000000
6	P00006	Túi xách dùng cho Điện thoại di động	1299500.000000
7	P00007	Túi xách dùng cho PC	NULL

Hình UD-12: Doanh thu theo tất cả sản phẩm.

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

277

Báo cáo doanh thu theo loại sản phẩm cho phép nhà quản lý biết được loại sản phẩm nào có kết quả kinh doanh tốt nhất.

Để làm điều này, trước tiên bạn liệt kê xem danh sách loại mẫu tin trong bảng Categories bằng phát biểu như ví dụ UD-13.

Ví dụ UD-13: Danh sách loại sản phẩm

```
SELECT CategoryID, CategoryNameInVietnamese
FROM Categories
GO
```

Khi thực thi phát biểu trên, danh sách loại sản phẩm được trình bày như hình UD-13.

	CategoryID	CategoryNameInVietnamese
1	1	Túi xách học sinh
2	2	Túi xách máy tính và điện thoại di động
3	4	Túi xách khác

Hình UD-13: Danh sách loại sản phẩm.

Trong trường hợp bạn muốn liệt kê doanh thu bán hàng theo loại sản phẩm thì khai báo như ví dụ UD-14.

Ví dụ UD-14: Doanh thu theo loại sản phẩm

```
SELECT C.CategoryID, CategoryNameInVietnamese,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
FROM Categories C, Products P, SalesInvoiceDetails D
```

-- Sử dụng mệnh đề WHERE với ba bảng dữ liệu

```
WHERE C.CategoryID = P.CategoryID
AND P.ProductID = D.ProductID
```

-- Sử dụng mệnh đề GROUP BY

```
GROUP BY C.CategoryID, CategoryNameInVietnamese
GO
```

Khi thực thi phát biểu trên, doanh thu bán hàng theo loại sản phẩm trong tháng hiện hành sẽ trình bày như hình UD-14.

Tương tự như trường hợp liệt kê tất cả sản phẩm, bạn có thể liệt kê mọi loại sản phẩm bằng cách sử dụng mệnh đề LEFT JOIN thay vì WHERE như ví dụ UD-15.

M® 278

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

The screenshot shows a results grid from SSMS with the following data:

	CategoryID	CategoryNameInVietnamese	SalesAmount
1	1	Túi xách học sinh	25932650.000000
2	2	Túi xách máy tính và điện thoại di động	3134500.000000

Hình UD-14: Doanh thu theo loại sản phẩm.**Ví dụ UD-15: Liệt kê tất cả loại sản phẩm**

```
SELECT C.CategoryID, CategoryNameInVietnamese,
       SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
```

--Sử dụng mệnh đề LEFT JOIN

```
FROM Categories C LEFT JOIN Products P
ON C.CategoryID = P.CategoryID
LEFT JOIN SalesInvoiceDetails D
ON P.ProductID = D.ProductID
```

--Sử dụng mệnh đề GROUP BY

```
GROUP BY C.CategoryID, CategoryNameInVietnamese
GO
```

Khi thực thi phát biểu trên, doanh thu bán hàng theo sản phẩm trong tháng hiện hành sẽ trình bày như hình UD-15.

The screenshot shows a results grid from SSMS with the following data:

	CategoryID	CategoryNameInVietnamese	SalesAmount
1	1	Túi xách học sinh	25932650.000000
2	2	Túi xách máy tính và điện thoại di động	3134500.000000
3	4	Túi xách khác	NULL

Hình UD-15: Liệt kê toàn bộ loại sản phẩm.**1.4. Doanh thu bán hàng theo khách hàng**

Nhằm xem xét khách hàng nào mua hàng với số tiền lớn, bạn có thể sử dụng phát biểu SELECT để liệt kê danh sách khách hàng cùng với số tiền mà họ đã mua hàng để làm báo cáo tài chính.

Tương tự như cách nhóm dữ liệu theo mã sản phẩm, bạn cũng có thể liệt kê doanh thu theo mã khách hàng bằng cách sử dụng mệnh đề INNER JOIN như ví dụ UD-16.

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

279



Ví dụ UD-16: Doanh thu theo khách hàng

```
SELECT C.CustomerID, CompanyNameInVietnamese,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As SalesAmount
```

--Sử dụng mệnh đề INNER JOIN

```
FROM Customers C INNER JOIN SalesInvoices S  
ON C.CustomerID = S.CustomerID  
INNER JOIN SalesInvoiceDetails D  
ON S.InvoiceNo = D.InvoiceNo
```

--Sử dụng mệnh đề GROUP BY

GROUP BY C.CustomerID, CompanyNameInVietnamese
GO

Khi thực thi phát biểu trên, doanh thu bán hàng theo khách hàng trong tháng hiện hành sẽ trình bày như hình UD-16.

	CustomerID	CompanyName Vietnamese	SalesAmount
1	A0001	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	8976250.000000
2	A0002	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	3832500.000000
3	A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	4300000.000000
4	A0004	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	5670000.000000
5	A0005	Công ty Cổ phần Suzumi Vietnam	2397650.000000
6	A0006	Tập đoàn UCIA USA	2185000.000000
7	A0007	Công ty Đa quốc gia UFCA	979500.000000
8	A0008	Công ty Cổ phần Rerui Vietnam	726250.000000

Hình UD-16: Doanh thu theo khách hàng.

Tuy nhiên, bạn cũng có thể liệt kê tất cả khách hàng có hay không phát sinh hóa đơn bán hàng bằng cách sử dụng mệnh đề LEFT JOIN như ví dụ UD-17.

Ví dụ UD-17: Doanh thu bán hàng theo khách hàng

```
SELECT C.CustomerID, CompanyNameInVietnamese,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As SalesAmount
```

--Sử dụng mệnh đề LEFT JOIN

FROM Customers C LEFT JOIN SalesInvoices S
ON C.CustomerID = S.CustomerID;

Sử dụng mảng để LEFT JOIN

---Sử dụng mệnh đề LEFT JOIN

Sử dụng mãnh dã GROUP BY là một lỗi SQL thường

 280

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

```
GROUP BY C.CustomerID, CompanyNameInVietnamese  
GO
```

Khi thực thi phát biểu trên, doanh thu bán hàng theo khách hàng trong tháng hiện hành sẽ trình bày như hình UD-17.

	CustomerID	CompanyNameInVietnamese	SalesAmount
1	A0001	Công ty Trách Nhiệm Hữu Hạn Microsoft Vietnam	8976250.000000
2	A0002	Công ty Trách Nhiệm Hữu Hạn IBM Vietnam	3832500.000000
3	A0003	Công ty Trách Nhiệm Hữu Hạn Kodaka Vietnam	4300000.000000
4	A0004	Công ty Trách Nhiệm Hữu Hạn E-Google Vietnam	5670000.000000
5	A0005	Công ty Cổ phần Suzuki Vietnam	2397650.000000
6	A0006	Tập đoàn UCIA USA	2185000.000000
7	A0007	Công ty Đa quốc gia UFCA	979500.000000
8	A0008	Công ty Cổ phần ReruitVietnam	726250.000000
9	A0009	Trung tâm giáo dục Vietnam	NULL

Hình UD-17: Doanh thu bán hàng theo khách hàng.

1.5. Doanh thu bán hàng theo tỉnh thành

Báo cáo doanh thu theo tỉnh thành là một trong những báo cáo giúp cho nhà quản lý biết được sản phẩm kinh doanh tốt theo vị trí địa lý.

Để làm điều này, trước tiên bạn có thể kiểm tra danh sách tỉnh thành bằng cách sử dụng phát biểu SELECT như ví dụ UD-18.

Ví dụ UD-18: Danh sách tỉnh thành

SELECT * FROM Provinces

GO

Khi thực thi phát biểu trên, danh sách tinh thành được trình bày như sau:

	ProvinceID	ProvinceName	CountryID
1	BDU	Bình Dương	VNA
2	DNA	Đồng Nai	VNA
3	HAN	Hà Nội	VNA
4	HCM	Hồ Chí Minh	VNA
5	TTH	Thừa Thiên - Huế	VNA

Hình UD-18: Danh sách tỉnh thành.

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

281 ®

Sau đó, bạn khai báo phát biểu SELECT với mệnh đề WHERE với bốn bảng dữ liệu Customers, SalesInvoices, SalesInvoiceDetails và Products như ví dụ UD-19.

Ví dụ UD-19: Doanh thu theo tỉnh thành

```
SELECT C.ProvinceID, P.ProductID,  
ProductNameInVietnamese,  
SUM(Quantity*Price*(1+VATRate/100)-Discount) As  
SalesAmount  
FROM Customers C, SalesInvoices S,  
Products P, SalesInvoiceDetails D
```

--Sử dụng mệnh đề WHERE
WHERE P.ProductID = D.ProductID
AND C.CustomerID = S.CustomerID
AND S.InvoiceNo = D.InvoiceNo

--Sử dụng mệnh đề GROUP BY
GROUP BY C.ProvinceID, P.ProductID,
ProductNameInVietnamese

--Sử dụng mệnh đề ORDER BY
ORDER BY SalesAmount DESC, ProductID ASC
GO

Khi thực thi phát biểu trên, doanh thu bán hàng theo tỉnh thành trong tháng hiện hành sẽ trình bày như hình UD-19

	ProvinceID	ProductID	ProductNameInVietnamese	SalesAmount
1	HCM	P00002	Túi xách dùng cho học sinh nữ	7617500.000000
2	HAN	P00003	Túi xách dùng cho học sinh nam	4845000.000000
3	HAN	P00001	Túi xách	4075000.000000
4	HCM	P00001	Túi xách	3731250.000000
5	HCM	P00003	Túi xách dùng cho học sinh nam	2368750.000000
6	HCM	P00004	Túi áo mưa	2245150.000000
7	DNA	P00005	Túi xách dùng cho Máy tính	1796250.000000
8	HAN	P00002	Túi xách dùng cho học sinh nữ	1050000.000000
9	HCM	P00006	Túi xách dùng cho Điện thoại di động	910750.000000
10	DNA	P00006	Túi xách dùng cho Điện thoại di động	388750.000000
11	HCM	P00005	Túi xách dùng cho Máy tính	38750.000000

Hình UD-19: Doanh thu bán hàng theo tỉnh thành.

Nếu muốn chỉ định tỉnh thành cụ thể là HCM, bạn có thể khai báo ví dụ UD-19 như ví dụ UD-20.

M® 282**Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng****Ví dụ UD-20: Doanh thu bán hàng tại HCM**

```

SELECT C.ProvinceID, P.ProductID,
ProductNameInVietnamese,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
FROM Customers C, SalesInvoices S,
Products P, SalesInvoiceDetails D
--Sử dụng mệnh đề WHERE
WHERE P.ProductID = D.ProductID
AND C.CustomerID = S.CustomerID
AND S.InvoiceNo = D.InvoiceNo
--Sử dụng phép toán AND
AND C.ProvinceID = 'HCM'
--Sử dụng mệnh đề GROUP BY
GROUP BY C.ProvinceID, P.ProductID,
ProductNameInVietnamese
--Sử dụng mệnh đề ORDER BY
ORDER BY SalesAmount DESC, ProductID ASC
GO

```

Khi thực thi phát biểu trên, doanh thu bán hàng theo tỉnh thành là HCM trong tháng hiện hành sẽ trình bày như hình UD-20.

ProvinceID	ProductID	ProductNameInVietnamese	SalesAmount
1 HCM	P00002	Túi xách dùng cho học sinh nữ	7617500.000000
2 HCM	P00001	Túi xách	3731250.000000
3 HCM	P00003	Túi xách dùng cho học sinh nam	2368750.000000
4 HCM	P00004	Túi áo mưa	2245150.000000
5 HCM	P00006	Túi xách dùng cho Điện thoại di động	910750.000000
6 HCM	P00005	Túi xách dùng cho Máy tính	38750.000000

Hình UD-20: Doanh thu bán hàng tại HCM.

Lưu ý: Bạn có thể sử dụng mệnh đề HAVING để trích lọc mẫu tin sau khi sử dụng GROUP BY như ví dụ UD-21.

Ví dụ UD-21: Khai báo mệnh đề HAVING

```

SELECT C.ProvinceID, P.ProductID,
ProductNameInVietnamese,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
FROM Customers C, SalesInvoices S,
Products P, SalesInvoiceDetails D
--Sử dụng mệnh đề WHERE
WHERE P.ProductID = D.ProductID

```

Ứng dụng: Sử dụng phát biểu SQL trong ứng dụng

283 M®

```

AND C.CustomerID = S.CustomerID
AND S.InvoiceNo = D.InvoiceNo
--Sử dụng mệnh đề GROUP BY
GROUP BY C.ProvinceID, P.ProductID,
ProductNameInVietnamese
--Sử dụng mệnh đề HAVING
HAVING ProvinceID = 'HCM'
--Sử dụng mệnh đề ORDER BY
ORDER BY SalesAmount DESC, ProductID ASC
GO

```

Khi thực thi phát biểu trên, doanh thu bán hàng theo tỉnh thành là HCM trong tháng hiện hành sẽ trình bày như hình UD-21.

	ProvinceID	ProductID	ProductNameInVietnamese	SalesAmount
1	HCM	P00002	Túi xách dùng cho học sinh nữ	7617500.000000
2	HCM	P00001	Túi xách	3731250.000000
3	HCM	P00003	Túi xách dùng cho học sinh nam	2368750.000000
4	HCM	P00004	Túi áo mưa	2245150.000000
5	HCM	P00006	Túi xách dùng cho Điện thoại di động	910750.000000
6	HCM	P00005	Túi xách dùng cho Máy tính	38750.000000

Hình UD-21: Doanh thu theo tỉnh thành là HCM.

1.6. N sản phẩm có doanh thu lớn nhất

Bất kỳ ngành nghề kinh doanh nào cũng cần biết số lượng cụ thể của sản phẩm bán chạy nhất theo tỉnh thành.

Chẳng hạn, bạn có thể liệt kê danh sách sản phẩm có doanh thu sắp xếp theo chiều giảm dần bằng cách khai báo như ví dụ UD-22.

Ví dụ UD-22: Danh sách sản phẩm và doanh thu bán hàng giảm dần

```

SELECT P.ProductID, ProductNameInVietnamese,
SUM(Quantity*Price*(1+VATRate/100)-Discount) As
SalesAmount
FROM Products P, SalesInvoiceDetails D
--Sử dụng mệnh đề WHERE
WHERE P.ProductID = D.ProductID
--Sử dụng mệnh đề GROUP BY
GROUP BY P.ProductID, ProductNameInVietnamese
--Sử dụng mệnh đề ORDER BY
ORDER BY SalesAmount DESC
GO

```

