**Import và export data trong SQL Server**

Trong bài này chúng ta sẽ tìm hiểu một số cách import và export data trong SQL Server. Sau đó sẽ bàn qua các loại Stored Procedure và Cursor.   
  
**Sử dụng bcp và BULK INSERT để import data**  
  
**bcp** là một command prompt dùng để import hay export data từ một data file (Text file hay Excel File) vào SQL Server hay ngược lại. Thường khi muốn chuyển một số lượng lớn data từ một database system khác như Oracle, DB2...sang SQL Server trước hết ta sẽ export data ra một text file sau đó import vào SQL Server dùng bcp command. Một trường hợp thông dụng hơn là ta export data từ SQL Server sang một Microsoft Excel file và Excel file này có thể là input cho một program hay một database system khác.  
  
Chúng ta cũng có thể chuyển data vào SQL Server dùng câu lệnh **BULK INSERT**. Tuy nhiên BULK INSERT **chỉ có thể import data** vào trong SQL Server chứ không thể export data ra một data file như bcp.  
  
Ðể có thể insert data vào SQL Server Database, data file phải có dạng bảng nghĩa là có cấu trúc hàng và cột. Chú ý khi data được bulk copy (copy hàng loạt dùng bcp hay BULK INSERT) vào một table trong SQL Server thì table đó phải tồn tại và data được cộng thêm vào (append). Ngược lại khi export data ra một data file thì một file mới sẽ được tạo ra hoặc data file sẽ bị overwrite nếu nó tồn tại.  
  
Cú pháp đầy đủ của lệnh bcp có thể xem trong SQL Server Books Online. Ở đây chỉ trình bày một số ví dụ đơn giản về cách sử dụng bcp command và BULK INSERT.  
  
**Ví dụ 1**: Giả sử bạn muốn export data từ table Orders trong PracticeDB (đây là database được tạo ra trong bài tập số 1 ) ra một text file trong đó các cột được phân cách bằng dấu ";". Bạn có thể làm như sau: mở DOS command prompt và đánh vào dòng lệnh sau:

PHP Code:

bcp PracticeDB..Orders out c:Orders.txt -c –T –t;

Trong ví dụ trên ta muốn bulk copy table Orders ra một text file trong đó :

***out****: copy data từ table hay view ra một data file (c:\Orders.txt). Ngược lại ta có thể dùng switch****in****để import data từ text file vào SQL Server.****-c****: bulk copy dùng kiểu dữ liệu Character (Char) (nếu không chỉ rõ thì SQL Server sẽ dùng "TAB" character (\t) để phân định các cột và dùng new line character (\n) để phân định các hàng như các giá trị default).****-t;****: dấu ";" đi sau switch "t" cho biết ta muốn dùng ";" để phân định các cột (nếu không sẽ dùng giá trị mặc định như trên)****-T****: dùng (NT) Trust connection để kết nối với database. Nghĩa là nếu user đã authenticated (cho phép) vào được Windows system thì đương nhiên được sử dụng SQL Server mà không cần dùng thêm username và password nào khác.*

**Ví dụ 2**: Thay vì copy toàn bộ table ta có thể dùng query để select một phần data và export ra text file như sau:

PHP Code:

bcp "Select \* From practiceDB..Orders" queryout c:Orders.txt -c -SVinhtai -Usa -Pabc

Trong ví dụ này ta select toàn bộ data trong Orders table ra một text file dùng query và SQL Server authentication.

***queryout****: cho biết đây là một query chứ không phải là table.****-S****: tên của SQL Server (hay tên của một Instance)****-U****: SQL user name dùng để log on****-P****: password dùng để log on.*

**Ví dụ 3** : dùng BULK INSERT để bulk copy data từ text file vào SQL Server database. Mở Query Analyser (BULK INSERT là một T-SQL command chứ không phải là một command prompt utility) và đánh vào các dòng sau :

PHP Code:

BULK INSERT PracticeDB..Orders FROM 'c:\Orders.txt ' WITH (DATAFILETYPE = 'CHAR')

Trong ví dụ trên DATAFILETYPE= 'CHAR' cho biết data được chứa dạng Char data type. Nêú muốn dùng data type dạng unicode thì dùng 'WIDECHAR'  
  
Chú ý: Các switch trong **bcp** command là case-sensitive. Nghĩa là chữ hoa và chữ thường sẽ có ý nghĩa khác nhau.  
  
Distributed Queries  
  
Ðôi khi chúng ta muốn select data từ những database system khác như MS Access, Oracle, DB2... hay thậm chí từ một SQL Server khác ta cần phải dùng distributed query. SQL Server sẽ dùng kỹ thuật OLEDB và các API để chuyển các query này tới các database system khác. Có 2 cách để truy cập vào các database system khác là dùng LINKED SERVER và Ad Hoc Computer Name.  
  
**Linked Server:**  
  
Linked Server là một server ảo được dùng để truy cập vào các database system khác. Một khi đã setup thì ta có thể query data dùng four-part name : linked\_server\_name.catalog.schema.object\_name . Trong đó catalog thường tương đương với database name, Schema tương đương với database owner và object\_name tương đương với table hay view.  
  
Ví dụ: Giả sử ta setup một Linked Server vào Access database "PracticeDB.mdb" trong đó các table đều tương tự như PracticeDB database trong SQL Server (được tạo ra trong phần bài tập số 1).  
  
Mở **Enterprise Manager -> Chọn node Security của local server -> Right-Click lên node Linked Server chọn New Linked Server**. Sau đó nhập vào tên của Linked Server *LinkedPracticeDB*, trong phần **Provider Name** chọn *Microsoft Jet 4.0 OLEDB Provider*. Trong phần **Data Source** nhập vào vị trí của Access database (C:\PracticeDB.mdb) và click OK.  
  
Ta sẽ có Linked Server tên LinkedPracticeDB xuất hiện dưới phần Security/Linked Server. Giả sử ta muốn select data từ Linked Server này ta có thể dùng Query Analyser như sau:

PHP Code:

Select \* from LinkedPracticeDB...Customers

Trong ví dụ trên ta dùng tên của Linked Server và theo sau là ba chấm (vì để truy cập vào database ta phải dùng four-part name nhưng trong trường hợp này ta dùng default value nên không cần cho biết tên của Catalog và Schema nhưng phải dùng dấu chấm để phân biệt từng phần).  
  
Ngoài cách trên ta có thể dùng pass-through query với **OPENQUERY** function như sau:

PHP Code:

Select \* from OPENQUERY(LinkedPracticeDB,'Select \* from Customers')

Trong ví dụ trên ta thấy function OPENQUERY sẽ trả về một data set và có thể nằm sau keyword FROM như một table. Khi dùng OPENQUERY function ta cần cho biết tên của Linked Server và query mà ta muốn thực hiện.  
  
Lưu ý: function trong SQL Server được dùng tương tự như là stored procedure.  
  
**Ad Hoc Computer Name**  
  
Ngoài cách dùng Linked Server như đã trình bày ở trên ta có thể dùng ad hoc computer name (ad hoc nghĩa là lâm thời, tạm thời). Nghĩa là đối với những database system mà ta thường xuyên query thì dùng Linked Server còn đối với những query lâu lâu mới dùng đến thì ta có thể select data bằng **OPENROWSET** hay **OPENDATASOURCE** functions  
  
Ví dụ: ta cũng sẽ select data từ Access database như trên dùng **OPENROWSET**

PHP Code:

Select \* from OPENROWSET('Microsoft.jet.oledb.4.0','C:\PracticeDB.mdb';

Trong ví dụ trên khi dùng OPENROWSET ta cần phải đưa vào tất cả những thông tin cần thiết để connect vào database như tên của Provider, vị trí của file, username, password (trường hợp này không có password) và tên của table mà ta muốn select. Mỗi lần ta thực thi câu lệnh trên SQL Server đều kiểm tra security trong khi đó nếu dùng Linked Server thì chỉ kiểm ta một lần mà thôi. OPENROWSET tương tự như OPENQUERY ở chỗ nó trả về một rowset và có thể đặt vào vị trí của một table trong câu lệnh query.  
  
Ngoài cách dùng trên ta cũng có thể dùng OPENDATASOURCE để query như sau:

PHP Code:

                  Select \* from OPENDATASOURCE('Microsoft.jet.oledb.4.0',  
                                                                  'Data Source = C:\PracticeDB.mdb; User ID = Admin; Password = ')  
                                                                  ...Customers

Trong ví dụ trên ta thấy OPENDATASOURCE trả về một phần của four-part name (nghĩa là tương đương với tên của Linked Server) cho nên ta phải dùng thêm ba dấu chấm.  
  
**Cursors**  
  
Nếu giải thích một cách ngắn gọn thì cursor tương tự như recordset hay dataset trong programming. Nghĩa là ta select một số data vào memory sau đó có thể lần lượt làm việc với từng record bằng cách Move Next...  
  
Có 3 loại cursors là Transact- SQL Cursors, API Cursors và Client Cursors. Trong đó Transact-SQL và API thuộc loại Server Cursors nghĩa là cursors được load lên và làm việc bên phía server. Trong khuôn khổ bài học này ta chỉ nghiên cứu Transact-SQL cursors.  
  
Transact-SQL cursors được tạo ra trên server bằng các câu lệnh Transact-SQL và chủ yếu được dùng trong stored procedures và triggers. Trước hết hãy xem qua một ví dụ về cursor:

PHP Code:

DECLARE @au\_lname varchar(40), @au\_fname varchar(20)

Trong ví dụ ở trên ta sẽ select LastName và FirstName từ Employees table của Northwind database và load vào Employee\_Cursor sau đó lần lượt in tên của các employee ra màn hình. Ðể làm việc với một cursor ta cần theo các bước sau:

1. Dùng câu lệnh DECLARE CURSOR để khai báo một cursor. Khi khai báo ta cũng phải cho biết câu lệnh SELECTsẽ được thực hiện để lấy data.
2. Dùng câu lệnh OPEN để đưa data lên memory (populate data). Ðây chính là lúc thực hiện câu lệnh SELECT vốn được khai báo ở trên.
3. Dùng câu lệnh FETCH để lấy từng hàng data từ record set. Cụ thể là ta phải gọi câu lệnh FETCH nhiều lần. FETCH tương tự như lệnh Move trong ADO recordset ở chỗ nó có thể di chuyển tới lui bằng câu lệnh FETCH FIRST, FETCH NEXT, FETCH PRIOR, FETCH LAST, FETCH ABSOLUTE n, FETCH RELATIVE n nhưng khác ở chỗ là nó lấy data bỏ vào trong variable (FETCH...FROM...INTO variable\_name). Thông thường ta FETCH data trước sau đó loop cho tới record cuối của Cursor bằng vòng lặp WHILE bằng cách kiểm tra global variable @@FETCH\_STATUS (=0 nghĩa là thành công).
4. Khi ta viếng thăm từng record ta có thể UPDATE hay DELETE tùy theo nhu cầu (trong thí dụ này chỉ dùng lệnh PRINT)
5. Dùng câu lệnh CLOSE để đóng cursor. Một số tài nguyên (memory resource) sẽ được giải phóng nhưng cursor vẫn còn được khai báo và có thể OPEN trở lại.
6. Dùng câu lệnh DEALLOCATE để phóng thích hoàn toàn các tài nguyên dành cho cursor (kể cả tên của cursor).

Lưu ý là trong ví dụ ở trên trước khi dùng Cursor ta cũng declare một số variable (@au\_fname và @au\_lname) để chứa các giá trị lấy được từ cursor. Bạn có thể dùng Query Analyzer để chạy thử ví dụ trên.  
  
**Stored Procedures**  
  
Trong những bài học trước đây khi dùng Query Analyzer chúng ta có thể đặt tên và save các nhóm câu lệnh SQL vào một file dưới dạng script để có thể sử dụng trở lại sau này. Tuy nhiên thay vì save vào text file ta có thể save vào trong SQL Server dưới dạng Stored Procedure.**Stored Procedure là một nhóm câu lệnh Transact-SQL đã được compiled (biên dịch) và chứa trong SQL Server dưới một tên nào đó và được xử lý như một đơn vị** (chứ không phải nhiều câu SQL riêng lẻ).  
**Ưu Ðiểm Của Stored Procedure**  
  
Stored Procedure có một số ưu điểm chính như sau:

* **Performance** : Khi thực thi một câu lệnh SQL thì SQL Server phải kiểm tra permission xem user gởi câu lệnh đó có được phép thực hiện câu lệnh hay không đồng thời kiểm tra cú pháp rồi mới tạo ra một execute plan và thực thi. Nếu có nhiều câu lệnh như vậy gởi qua network có thể làm giảm đi tốc độ làm việc của server. SQL Server sẽ làm việc hiệu quả hơn nếu dùng stored procedure vì người gởi chỉ gởi một câu lệnh đơn và SQL Server chỉ kiểm tra một lần sau đó tạo ra một execute plan và thực thi. Nếu stored procedure được gọi nhiều lần thì execute plan có thể được sử dụng lại nên sẽ làm việc nhanh hơn. Ngoài ra cú pháp của các câu lệnh SQL đã được SQL Sever kiểm tra trước khi save nên nó không cần kiểm lại khi thực thi.
* **Programming Framework** : Một khi stored procedure được tạo ra nó có thể được sử dụng lại. Ðiều này sẽ làm cho việc bảo trì (maintainability) dễ dàng hơn do việc tách rời giữa business rules (tức là những logic thể hiện bên trong stored procedure) và database. Ví dụ nếu có một sự thay đổi nào đó về mặt logic thì ta chỉ việc thay đổi code bên trong stored procedure mà thôi. Những ứng dụng dùng stored procedure này có thể sẽ không cần phải thay đổi mà vẫn tương thích với business rule mới. Cũng giống như các ngôn ngữ lập trình khác stored procedure cho phép ta đưa vào các input parameters (tham số) và trả về các output parameters đồng thời nó cũng có khả năng gọi các stored procedure khác.
* **Security** : Giả sử chúng ta muốn giới hạn việc truy xuất dữ liệu trực tiếp của một user nào đó vào một số tables, ta có thể viết một stored procedure để truy xuất dữ liệu và chỉ cho phép user đó được sử dụng stored procedure đã viết sẵn mà thôi chứ không thể "đụng" đến các tables đó một cách trực tiếp. Ngoài ra stored procedure có thể được encrypt (mã hóa) để tăng cường tính bảo mật.

**Các Loại Stored Procedure**  
  
Stored procedure có thể được chia thành 5 nhóm như sau:

1. **System Stored Prcedure** : Là những stored procedure chứa trong Master database và thường bắt đầu bằng tiếp đầu ngữ **sp\_** . Các stored procedure này thuộc loại built-in và chủ yếu dùng trong việc quản lý database (administration) và security. Ví dụ bạn có thể kiểm tra tất cả các processes đang được sử dụng bởi user DomainName\Administrators bạn có thể dùng sp\_who @loginame='DomainName\Administrators'. Có hàng trăm system stored procedure trong SQL Server. Bạn có thể xem chi tiết trong SQL Server Books Online.
2. **Local Stored Procedure** : Ðây là loại thường dùng nhất. Chúng được chứa trong user database và thường được viết để thực hiện một công việc nào đó. Thông thường người ta nói đến stored procedure là nói đến loại này. Local stored procedure thường được viết bởi DBA hoặc programmer. Chúng ta sẽ bàn về cách tạo stored prcedure loại này trong phần kế tiếp.
3. **Temporary Stored Procedure** : Là những stored procedure tương tự như local stored procedure nhưng chỉ tồn tại cho đến khi connection đã tạo ra chúng bị đóng lại hoặc SQL Server shutdown. Các stored procedure này được tạo ra trên TempDB của SQL Server nên chúng sẽ bị delete khi connection tạo ra chúng bị cắt đứt hay khi SQL Server down. **Temporary stored procedure được chia làm 3 loại** : **local** (bắt đầu bằng #), **global** (bắt đầu bằng ##) và stored procedure được **tạo ra trực tiếp trên TempDB**. Loại local chỉ được sử dụng bởi connection đã tạo ra chúng và bị xóa khi disconnect, còn loại global có thể được sử dụng bởi bất kỳ connection nào. Permission cho loại global là dành cho mọi người (public) và không thể thay đổi. Loại stored procedure được tạo trực tiếp trên TempDB khác với 2 loại trên ở chỗ ta **có thể set permission**, chúng **tồn tại kể cả sau khi connection tạo ra chúng bị cắt đứt** và **chỉ biến mất khi SQL Server shut down**.
4. **Extended Stored Procedure** : Ðây là một loại stored procedure sử dụng một chương trình ngoại vi (external program) vốn được compiled thành một DLL để mở rộng chức năng hoạt động của SQL Server. Loại này thường bắt đầu bằng tiếp đầu ngữ **xp\_** .Ví dụ, xp\_sendmail dùng đề gởi mail cho một người nào đó hay xp\_cmdshell dùng để chạy một DOS command... Ví dụ xp\_cmdshell 'dir c:\' . Nhiều loại extend stored procedure được xem như system stored procedure và ngược lại.
5. **Remote Stored Procedure** : Những stored procedure gọi stored procedure ở server khác.

**Viết Stored Procedure**  
  
Tên và những thông tin về Stored Procedure khi được tạo ra sẽ chứa trong SysObjects table còn phần text của nó chứa trong SysComments table. Vì Stored Procedure cũng được xem như một object nên ta cũng có thể dùng các lệnh như CREATE, ALTER, DROP để tạo mới, thay đổi hay xóa bỏ một stored procedure. Chúng ta hãy xem một ví dụ sau về Stored Procedure: Ðể tạo một stored procedure bạn có thể dùng Enterprise Manager **click lên trên Stored Procedure -> New Stored Procedure...**. Trong ví dụ này ta sẽ tạo ra một stored procedure để insert một new order vào Orders table trong Practice DB. Ðể insert một order vào database ta cần đưa vào một số input như OrderID, ProductName (order món hàng nào) và CustomerName (ai order). Sau đó ta trả về kết quả cho biết việc insert đó có thành công hay không. Result = 0 là insert thành công.

PHP Code:

@OrderID smallint,

Ðể tạo ra một stored procedure ta dùng lệnh CREATE PROCEDURE theo sau là tên của nó (nếu là temporary stored procedure thì thêm dấu # trước tên của procedure. Nếu muốn encrypt thì dùng WITH ENCRYPTION trước chữ AS) và các input hoặc ouput parameters. Nếu là output thì thêm keyword OUTPUT đằng sau parameter. Ta có thể cho giá trị default cùng lúc với khai báo data type của parameter. Kể từ sau chữ AS là phần body của stored procedure.   
  
Trong ví dụ ở trên trước hết ta khai báo một biến @CustomerID sau đó bắt đầu một transaction bằng BEGIN TRANSACTION (toàn bộ công việc insert này được thực hiện trong một Transaction nghĩa là hoặc là insert thành công hoặc là không làm gì cả- all or nothing). Trước hết ta kiểm tra xem người khách hàng là người mới hay cũ. Nếu là người mới thì ta "tiện tay" insert vào Customers table luôn còn nếu không thì chỉ insert vào Orders table mà thôi. Nếu là người customer mới ta lấy CustomerID lớn nhất từ Customers table bỏ vào biến @CustomerID và sau đó tăng lên một đơn vị dùng cho việc Insert Customer ở dòng kế tiếp.  
  
Sau khi insert người customer mới ta tiếp tục công việc bằng việc kiểm tra xem Order muốn insert có tồn tại chưa (vì nếu order đã tồn tại thì khi insert SQL Server sẽ báo lỗi do OrderID là Primary key). Nếu như order trên vì lý do nào đó đã có trong DB thì ta roll back và trả kết quả =1 còn nếu không thì ta insert một order mới vào và commit transaction với kết quả trả về =0.  
  
Tương tự như vậy nếu người customer đã tồn tại (sau chữ else đầu tiên) thì ta chỉ việc insert order giống như trên. Trong mọi trường hợp kể trên ta đều in ra kết quả và return.  
  
Ví dụ trên đây chỉ mang tính học hỏi còn trên thực tế database có thể phức tạp hơn nhiều nên việc viết stored procedure đòi hỏi kiến thức vững chắc về SQL và kỹ năng về programming.