CTT102 – Cơ sở dữ liêu

Stored Procedure và Function

Tóm tắt nội dung bài thực hành:

Lập trình trên môi trường SQL Server với Stored Procedure và Function

Bộ môn **Hệ thống thông tin**

Khoa Công nghệ thông tin

ĐH Khoa học tự nhiên TP HCM

MỤC LỤC

STORED PROCEDURE VÀ FUNCTION

1 Mục tiêu và tóm tắt nội dung

Sau khi hoàn thành bài thực hành này sinh viên sẽ biết được:

- Kỹ thuật viết hàm và thủ tục trong SQL Server cơ bản

2 Hướng dẫn chi tiết

2.1 Stored Procedure (Đọc là Stored Procedure hoặc Procedure)

2.1.1 Giới thiệu

Khi chúng ta tạo một ứng dụng với Microsoft SQL Server, ngôn ngữ lập trình Transact-SQL là ngôn ngữ chính giao tiếp giữa ứng dụng và database của SQL Server. Khi chúng ta tạo các chương trình bằng Transact-SQL, hai phương pháp chính có thể dùng để lưu trữ và thực thi cho các chương trình là:

- Chúng ta có thể lưu trữ các chương trình cục bộ và tạo các ứng dụng để gởi các lệnh đến SQL Server và xử lý các kết quả,
- Chúng ta có thể lưu trữ những chương trình như các stored procedure trong SQL Server và tạo ứng dụng để gọi thực thi các stored procedure và xử lý các kết quả.

Đặc tính của **Stored-procedure** trong SQL Server :

- Chấp nhận những tham số vào và trả về những giá trị được chứa trong các tham số ra để gọi những thủ tục hoặc xử lý theo lô.
- Chứa các lệnh của chương trình để thực hiện các xử lý trong database, bao gồm cả lệnh gọi các thủ tục khác thực thi.
- Trả về các trạng thái giá trị để gọi những thủ tục hoặc thực hiện các xử lý theo lô để cho biết việc thực hiện thành công hay thất bại, nếu thất bại thì lý do vì sao thất bại.

Ta có thể dùng Transact–SQL **EXCUTE** để thực thi các stored procedure. Stored procedure khác với các hàm xử lý là giá trị trả về của chúng không chứa trong tên và chúng không được sử dụng trực tiếp trong biểu thức.

Stored procedure có những thuận lợi so với các chương trình Transact-SQL lưu trữ cục bộ là:

- **Stored procedure cho phép điều chỉnh chương trình cho phù hợp**: Chúng ta có chỉ tạo stored procedure một lần và lưu trữ trong database một lần, trong chương trình chúng ta có thể gọi nó với số lần bất kỳ. Stored procedure có thể được chỉ rõ do một người nào đó tạo ra và sự thay đổi của chúng hoàn toàn độc lập với source code của chương trình.
- **Stored procedure cho phép thực thi nhanh hơn**: nếu sự xử lý yêu cầu một đoạn source code Transact – SQL khá lớn hoặc việc thực thi mang tính lặp đi lặp lại thì stored procedure thực hiện nhanh hơn việc thực hiện hàng loạt các lệnh Transact-SQL. Chúng được phân tích cú pháp và tối ưu hóa trong lần thực thi đầu tiên và một phiên bản dịch của chúng trong đó sẽ được lưu trong bộ nhớ để sử dụng cho lần sau, nghĩa là trong những lần thực hiện sau chúng không cần phải phân tích cú pháp và tối ưu lại, mà chúng sẽ sử dụng kết quả đã được biên dịch trong lần đầu tiên.
- **Stored procedure có thể làm giảm bớt vấn đề kẹt đường truyền mạng**: giả sử một xử lý mà có sử dụng hàng trăm lệnh của Transact-SQL và việc thực hiện thông qua từng dòng lệnh đơn, như vậy việc thực thông qua stored procedure sẽ tốt hơn, vì nếu không khi thực hiện chúng ta phải gởi hàng trăm lệnh đó lên mạng và điều này sẽ dẫn đến tình trạng kẹt mạng.
- **Stored procedure có thể sử dụng trong vấn đề bảo mật của máy**: vì người sử dụng có thể được phân cấp những quyền để sử dụng các stored procedure này, thậm chí họ không được phép thực thi trực tiếp những stored procedure này.

2.1.2 Cú pháp

Một Stored procedure được định nghĩa gồm những thành phần chính sau:

- Tên của stored procedure
- Các tham số
- Thân của stored procedure: bao gồm các lệnh của Transact-SQL dùng để thực thi procedure.

Một stored procedure được tạo bằng lệnh Create Procedure, và có thể thay đổi bằng cách dùng lệnh Alter Procedure, và có thể xóa bằng cách dùng lệnh Drop Procedure trong lập lệnh của Transact – SQL

Tạo procedure:

```
CREATE PROCEDURE procedure_name
        {@parameter data_type input/output }/*các biến tham số vào ra*/
AS
        [khai báo các biến cho xử lý]
        {Các câu lệnh transact-sql}
        RETURN value -- Stored procedure có thể trả về giá trị hoặc không
```

Một số lưu ý khi viết stored procedure:

```
-- 1. Ghi chú 1, một dòng
/* 2. Ghi chú 2
Nhiều dòng  */
/*3. Khai báo biến*/
DECLARE @parameter_name data_type
/*4. Gán giá trị cho biến*/
SET @parameter_name=value
SELECT @parameter_name=column FROM ...
/*5. In thông báo ra màn hình*/
print N'Chuỗi thông báo unicode'
/*6. Thông báo lỗi */
raiserror (N'Nội dung thông báo lỗi ', 16, 1)
/*7. Lệnh rẽ nhánh */
```

```
IF (điều kiện-có thể sử dụng câu truy vấn con và từ khoá EXISTS)
BEGIN
        {Các lệnh nếu thoả điều kiện / nếu chỉ có 1 lệnh thì không cần BEGIN ... END}
END
/*8. Lệnh rẽ nhánh có ELSE  */
IF (điều kiện-có thể sử dụng câu truy vấn con và từ khoá EXISTS)
BEGIN
        {Các lệnh nếu thoả điều kiện / nếu chỉ có 1 lệnh thì không cần BEGIN ... END}
END
ELSE
BEGIN
        {Các lệnh nếu không thoả điều kiện / nếu chỉ có 1 lệnh thì không cần BEGIN ... END}
END
/*9. Vòng lặp WHILE (Lưu ý: Không có vòng lặp FOR) */
WHILE ( điều kiện )
BEGIN
        {Các lệnh nếu thoả điều kiện / nếu chỉ có 1 lệnh thì không cần BEGIN ... END}
END
```

Biên dịch: Chọn **đúng** đoạn mã lệnh Tạo stored-procedure → F5

Thực thi procedure:

```
EXEC procedure_name --Stored-proc không tham số
EXEC procedure_name Para1_value, Para2_value, ... --Stored-proc có tham số

Lấy giá trị trả về của stored procedure:
EXEC @bien = procedure_name --Stored-proc không tham số
EXEC @bien = procedure_name Para1_value, Para2_value, ... --Stored-proc có tham số
```

2.1.3 Ví dụ

Ví dụ 1: Viết stored procedure tính tổng 2 số:

```
--Tạo
CREATE PROCEDURE sp_Tong @So1 int, @So2 int, @Tong int out
AS
        SET @Tong = @So1 + @So2;
--Biên dịch stored-procedure→F5
--Kiểm tra
Declare @Sum int
Exec sp_Tong 1, -2, @Sum out
Print @Sum
```

Ví dụ 2: Viết stored procedure tính tổng các số chẵn từ m→n:

```
--Tạo
CREATE PROCEDURE sp_TongChanMN @m int, @n int
AS
        Declare @tong int
        Declare @i  int
        Set @tong = 0
        Set @i = @m
        While (@i < @n)
        BEGIN
                IF (@i % 2 = 0)
                        SET @tong = @tong + @i
                SET @i = @i + 1

        END
        Print  @tong
--Biên dịch stored-procedure→F5
```

```
--Kiểm tra
Exec sp_TongChanMN 1, 20
```

Ví dụ 3: Viết stored procedure kiểm tra sự tồn tại của giáo viên theo mã:

```
--Tạo
CREATE PROCEDURE sp_KiemTraGVTonTai @MaGV char(9)
AS
        IF ( EXISTS (SELECT * FROM GIAOVIEN WHERE MAGV=@MAGV) )
                Print N'Giáo viên tồn tại'
        ELSE
                Print N'Không tồn tại giáo viên ! ' + @MaGV
--Biên dịch stored-procedure→F5
--Kiểm tra
Exec sp_KiemTraGVTonTai '001'
```

Ví dụ 4: Viết stored procedure xuất ra danh sách giáo viên của một bộ môn:

```
--Tạo
CREATE PROCEDURE sp_DanhSachGiaoVien @MaBM char(9)
AS
        SELECT * FROM GIAOVIEN WHERE MABM=@MaBM
--Biên dịch stored-procedure→F5
--Kiểm tra
Exec sp_DanhSachGiaoVien  'HTTT'
```

2.2 Function

2.2.1 Giới thiệu

Trong SQL Server ta có thể viết hàm và lấy giá trị trả về. Các dạng hàm có thể viết như sau :

- Hàm trả về giá trị vô hướng (scalar value) : varchar, int, ...
- Hàm trả về giá trị là bảng tạm (inline table-valued) : table

2.2.2 Cú pháp

Tạo hàm:

```
CREATE FUNTION function_name ( [@parameter_name parameter_data_type] )
RETURNS [return Data-type] /*Returns có 's' */
AS Begin
        return ([scalar value])
End
```

Tạo hàm trả về bảng (table):

```
CREATE FUNTION function_name ( [@parameter_name parameter_data_type] )
RETURNS table
AS
        return [select command]
```

2.2.3 Ví dụ

Ví dụ 5: Viết hàm tính tuổi dựa vào ngày sinh

```
-- Tạo hàm
CREATE FUNTION fTinhTuoi ( @ngaysinh datetime  )
RETURNS int
AS
BEGIN
```

```
        return year(getdate()) – year(@ngaysinh)
END
-- Biên dịch: F5
-- Kiểm tra
print dbo.fTinhTuoi('1/1/2000')
-- hoặc
SELECT MAGV, dbo. fTinhTuoi (NgaySinh)
FROM GIAOVIEN
```

Ví dụ 6: Viết hàm lấy danh sách giáo viên bộ môn HTTT

```
-- Tạo hàm
CREATE FUNTION fDSGV_HTTT ( )
RETURNS table
AS
        return (SELECT * FROM GIAO VIEN WHERE MABM='HTTT')
-- Biên dịch: F5
-- Kiểm tra
SELECT * FROM  dbo.fDSGV_HTTT ( )
```

3 Bài tập tại lớp

Yêu cầu:

Viết các stored procedure sau:

a. In ra câu chào "Hello World !!!".
b. In ra tổng 2 số.
c. Tính tổng 2 số (sử dụng biến output để lưu kết quả trả về).
d. In ra tổng 3 số (Sử dụng lại stored procedure Tính tổng 2 số).
e. In ra tổng các số nguyên từ m đến n.
f. Kiểm tra 1 số nguyên có phải là số nguyên tố hay không.

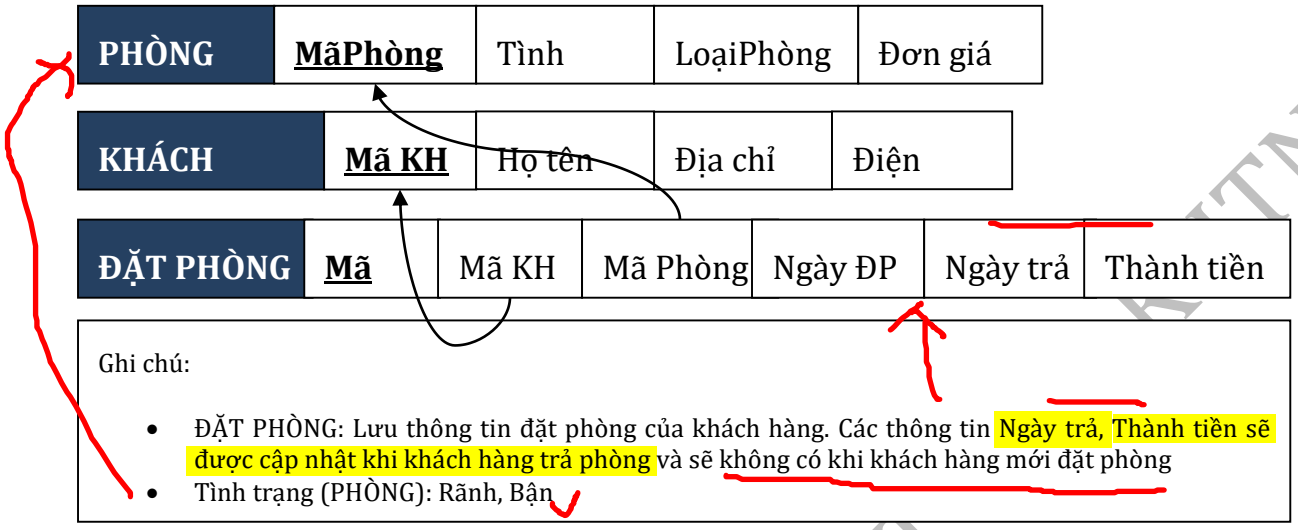
g. In ra tổng các số nguyên tố trong đoạn m, n.
h. Tính ước chung lớn nhất của 2 số nguyên.
i. Tính bội chung nhỏ nhất của 2 số nguyên.

Viết các stored procedure sau:

j. Xuất ra toàn bộ danh sách giáo viên.
k. Tính số lượng đề tài mà một giáo viên đang thực hiện.
l. In thông tin chi tiết của một giáo viên(sử dụng lệnh print): Thông tin cá nhân, Số lượng đề tài tham gia, Số lượng thân nhân của giáo viên đó.
m. Kiểm tra xem một giáo viên có tồn tại hay không (dựa vào MAGV).
n. Kiểm tra quy định của một giáo viên: Chỉ được thực hiện các đề tài mà bộ môn của giáo viên đó làm chủ nhiệm.
o. Thực hiện thêm một phân công cho giáo viên thực hiện một công việc của đề tài:
    - Kiểm tra thông tin đầu vào hợp lệ: giáo viên phải tồn tại, công việc phải tồn tại, thời gian tham gia phải >0
    - Kiểm tra quy định ở câu n.
p. Thực hiện xoá một giáo viên theo mã. Nếu giáo viên có thông tin liên quan (Có thân nhân, có làm đề tài, ...) thì báo lỗi.
q. In ra danh sách giáo viên của một phòng ban nào đó cùng với số lượng đề tài mà giáo viên tham gia, số thân nhân, số giáo viên mà giáo viên đó quản lý nếu có, ...
r. Kiểm tra quy định của 2 giáo viên a, b: Nếu a là trưởng bộ môn c của b thì lương của a phải cao hơn lương của b. (a, b: mã giáo viên)
s. Thêm một giáo viên: Kiểm tra các quy định: Không trùng tên, tuổi > 18, lương > 0
t. Mã giáo viên được xác định tự động theo quy tắc: Nếu đã có giáo viên 001, 002, 003 thì MAGV của giáo viên mới sẽ là 004. Nếu đã có giáo viên 001, 002, 005 thì MAGV của giáo viên mới là 003.

4 Bài tập về nhà

Cho lược đồ CSDL:



Ghi chú:

- ĐẶT PHÒNG: Lưu thông tin đặt phòng của khách hàng. Các thông tin Ngày trả, Thành tiền sẽ được cập nhật khi khách hàng trả phòng và sẽ không có khi khách hàng mới đặt phòng
- Tình trạng (PHÒNG): Rãnh, Bận

Yêu cầu:

1. Viết stored procedure sau:

Tên stored procedure: **spDatPhong**

Nội dung: ghi nhận thông tin đặt phòng của khách hàng xuống cơ sở dữ liệu.

Tham số yêu cầu: mã khách hàng (@makh), mã phòng (@maphong), ngày đặt phòng (@ngaydat).

Lưu ý: Mã đặt phòng là số nguyên và phải phát sinh tự động theo cách sau: mã đặt phòng phát sinh = mã đặt phòng lớn nhất + 1.

Các yêu cầu và kiểm tra và tính toán:

- Kiểm tra mã khách hàng phải hợp lệ (phải xuất hiện trong bảng KHÁCH HÀNG)
- Kiểm tra mã phòng hợp lệ (phải xuất hiện trong bảng PHÒNG)
- Chỉ được đặt phòng khi tình trạng của phòng là "**Rảnh**"
- Nếu các kiểm tra hợp lệ thì ghi nhận thông tin đặt phòng xuống CSDL (Ngày trả và thành tiền của khi đặt phòng là NULL)
- Sau khi đặt phòng thành công thì phải cập nhật tình trạng của phòng là "**Bận**"

Yêu cầu khác: **Phải có diễn giải bằng lời của các bước thực hiện**.

2. Stored procedure/function

Tên stored procedure: **spTraPhong**

Nội dung: ghi nhận thông tin trả phòng của khách hàng xuống cơ sở dữ liệu.

Tham số yêu cầu: mã đặt phòng (@madp), mã khách hàng (@makh)

Các yêu cầu về kiểm tra và tính toán:

- Kiểm tra tính hợp lệ của mã đặt phòng, mã khách hàng: Hợp lệ nếu khách hàng có thực hiện việc đặt phòng.
- Ngày trả phòng chính là ngày hiện hành.
- Tiền thanh toán được tính theo công thức: Tien = Số ngày mượn x đơn giá của phòng.
- Phải thực hiện việc cập nhật tình trạng của phòng là "**Rảnh**" sau khi ghi nhận thông tin trả phòng.

Yêu cầu khác: **Phải có diễn giải bằng lời của các bước thực hiện**.

Thời lượng: 03 giờ.

HẾT