TUẦN 3 – THỦ TỤC THƯỜNG TRÚ (STORED PROCEDURE)

I. HƯỚNG DẪN

a. Định nghĩa và các tính chất của thủ tục thường trú – stored procedure

Định nghĩa: tập hợp các lệnh được đóng gói và lưu trữ lại ở database và được người dùng gọi thực thi khi cần.

Ưu điểm:

- Tăng tốc độ truy vấn
- Giảm lưu lượng truyền dữ liệu đến server
- Linh hoạt, tái sử dụng
- Bảo mật

Thành phần:

- Tên thủ tục
- Tham số
  - Tham số đầu vào (input): chứa các giá trị do người dùng cung cấp cho thủ tục khi gọi thủ tục.
  - Tham số đầu ra (output): chứa kết quả trả ra cho người dùng.
- Thân thủ tục – nội dung thực thi của thủ tục

b. Khai báo, sử dụng biến

Biến cần được khai báo bằng lệnh declare và gán giá trị bằng lệnh set trước khi được sử dụng. Tất cả các biến do người dùng định nghĩa trong T-SQL đều phải bắt đầu bằng @.

```
DECLARE @bien1 KDL, @bien2 KDL, …
SET @bien      = gia_tri -- gán biến bằng một giá trị cụ thể, ví dụ: @ms = '1102'
               = @bien2 -- gán biến bằng biến khác
               = (SELECT A FROM BANG, …) -- gán biến bằng kết quả của 1 câu truy vấn
```

Người dùng có thể gán cùng lúc nhiều giá trị trả ra từ câu truy vấn vào các biến khác nhau.

```
SELECT @bien1 = A, @bien2 = B FROM BANG, …
```

Lưu ý:

- Khi dùng lệnh SET để gán kết quả từ một câu truy vấn cho biến thì câu truy vấn chỉ được trả về tối đa 1 dòng 1 cột, cột được chọn trả ra cũng phải cùng miền giá trị với biến.
- Khi dùng lệnh SELECT để gán nhiều giá trị vào các biến khác nhau từ kết quả của câu truy vấn thì miền giá trị của các biến phải tương ứng giống nhau với miền giá trị các cột thuộc tính trả ra từ câu truy vấn được gán vào biến. Ngoài ra, câu truy vấn chỉ được trả về 1 dòng duy nhất.

c. Tạo thủ tục mới

```
CREATE PROC sp_TenProc @ts1 KDL [in | out], @ts2 KDL [in | out], …
AS
BEGIN
        T_SQL
END
GO
```

Lưu ý:

- Không cần thêm từ khóa in khi khai báo tham số đầu vào.
- Bắt buộc phải có từ khóa out khi khai báo tham số đầu ra.
- Một thủ tục có thể có nhiều tham số đầu vào hoặc đầu ra hoặc không có tham số nào.
- Trong thủ tục có thể sử dụng **lệnh return** để **trả về một số nguyên**.

d. Chỉnh sửa, xóa thủ tục

Chỉnh sửa:

```
ALTER PROC sp_TenProc @ts1 KDL [in | out], @ts2 KDL [in | out], …
AS
BEGIN
        T_SQL
END
GO
```

Xóa:

```
DROP PROC sp_TenProc
```

e. Cấu trúc điều khiển

Điều kiện rẽ nhánh:

```
IF điều_kiện
BEGIN
        lệnh_1
        ...
END
```

```
IF điều_kiện
BEGIN
        lệnh_1 ...
END
ELSE
BEGIN
        lệnh_1 ...
END
```

```
CASE giá_trị
   WHEN ... THEN ...
   WHEN ... THEN ...
   WHEN ... THEN ...
END
```

Vòng lặp:

```
WHEN điều_kiện
BEGIN
        lệnh_1
        …
END
```

f. Xuất dữ liệu

- Dùng lệnh select: dữ liệu xuất ra dạng bảng.
- Dùng lệnh print: dữ liệu xuất ra dạng chuỗi thông điệp.

g. Thực thi thủ tục

Truyền tham số tường minh:
EXEC **ten_thu_tuc @ts1 = @bien1** [in | out], **@ts2 = @bien2** [in | out], **…**

Truyền tham số không tường minh:
EXEC **ten_thu_tuc @bien1** [in | out], **@bien2** [in | out], **…**

Bắt giá trị do thủ tục trả về bằng lệnh return:
EXEC **@kq = ten_thu_tuc @bien1, @bien2, …**

Lưu ý:

- Truyền tham số tường minh không cần quan tâm thứ tự khai báo các tham số lúc tạo thủ tục.
- Truyền tham số không tường minh thì các biến truyền vào sẽ lần lượt được gán vào các tham số của thủ tục theo đúng thứ tự khai báo các tham số khi tạo thủ tục, do đó cần truyền chính xác thứ tự các biến.
- Biến được gán vào tham số phải cùng miền giá trị với tham số.
- Thủ tục chỉ có thể trả về số nguyên.

Ví dụ 1: Viết stored procedure in ra "Hello World".

```
create proc sp_xinChao
as
begin
    print 'Hello World'
end
go
--Thuc thi
exec sp_xinChao
```

Ví dụ 2: Viết stored procedure nhận vào tên và in ra "Hello " + tên.

```
create proc sp_xinChaoTen @ten nvarchar(20)
as
begin
    print 'Hello' + space(1) + @ten
end
go

exec sp_xinChaoTen 'LyLy'
```

Ví dụ 3: Viết stored procedure kiểm tra số a là chẵn hay lẻ, trả về 0 nếu a chẵn và trả về 1 nếu a lẻ.

Cách 1: Sử dụng lệnh return trả về kết quả kiểm tra

```
create proc sp_kiemTraChanLe @a int
as
begin
    if @a%2 = 0
        return 0
    else
        return 1
end
go

declare @kq int
--truyen tham so khong tuong minh khi thuc thi
exec @kq = sp_kiemTraChanLe 3
print @kq

--truyen tham so tuong minh khi thuc thi
exec @kq = sp_kiemTraChanLe @a = 3
print @kq
```

Cách 2: Sử dụng tham số output chứa kết quả kiểm tra

```
create proc sp_kiemTraChanLe @a int, @kq int out
as
begin
    if @a%2 = 0
        set @kq = 0
    else
        set @kq = 1
end
go

declare @kq int
--truyen tham so khong tuong minh khi thuc thi
exec sp_kiemTraChanLe 3, @kq out
print @kq
```

Ví dụ 4: Viết stored procedure nhận vào n, m và in ra các số lẻ nằm trong đoạn n, m. Yêu cầu sử dụng lại stored procedure ở ví dụ 4.

```
create proc sp_inSoLe @n int, @m int
as
begin
    declare @i int, @kq int
    set @i = @n
    while @i <= @m
    begin
        exec sp_kiemtrachanle @i, @kq out
        if @kq = 1 --so le
            print @i
        set @i = @i + 1
    end
end
go
```

Có thể dùng lệnh:
exec @kq = sp_kiemtrachanle @i
Trong trường hợp kết quả kiểm tra được trả về qua lệnh return trong thân thủ tục

II. BÀI TẬP

Viết stored procedure thực hiện các yêu cầu sau:

1. Nhận vào hai số a, b và trả về tổng a, b.
2. Nhận vào hai số a, b và trả về hiệu a, b.
3. Nhận vào hai số a, b và trả về tích a, b.
4. Nhận vào hai số a, b và trả về thương a, b.
5. Nhận vào hai số a, b và trả về số dư của phép chia a cho b.
6. Nhận vào hai số a, b và i.
    - Nếu i là 1 trả về tổng a, b
    - Nếu i là 2 trả về hiệu a, b
    - Nếu i là 3 trả về tích a, b
    - Nếu i là 4 trả về thương a, b
    - Nếu i là 5 trả về số dư phép chia a cho b

    Yêu cầu: sử dụng lại stored procedure ở câu 1 đến câu 5 và dùng case để xét giá trị i.

7. Nhận vào n, m và trả về tổng các giá trị nằm trong đoạn n, m *(dùng tham số output)*.
8. Nhận vào năm n, kiểm tra xem n có phải năm nhuận không. Nếu n là năm nhuận trả về 1 còn không phải trả về 0 *(dùng tham số output)*.
9. Nhận vào năm n, m, đếm xem có bao nhiêu năm nhuận trong đoạn n đến m *(dùng tham số output)*.
10. Nhận vào n và trả ra giá trị n!, biết rằng n! = 1*2*3*…*n
11. Nhận vào ngày a (kiểu date hoặc datetime) cho biết tháng của ngày a có bao nhiêu ngày. *Ví dụ: nếu ngày a là ngày 15/02/2000 thì trả về 28 là số ngày của tháng 2 trong năm 2000.*
12. Nhận vào n, kiểm tra xem n có phải số nguyên tố không. Nếu là số nguyên tố trả về 1 còn không trả về 0.
13. Nhận vào n, m và trả về tích các số nguyên tố nằm trong đoạn n, m *(dùng tham số output)*. Lưu ý: số nguyên tố là số có hai ước chung là 1 và chính nó, ví dụ: 13, 17, ...
14. Nhận vào n, kiểm tra xem n có phải số chính phương không. Nếu là số chính phương trả về 1 còn không trả về 0. Lưu ý: số chính phương là bình phương của một số khác, ví dụ: 4, 9, ...
15. Nhận vào n, m và trả về tổng các số chính phương nằm trong đoạn n, m *(dùng tham số output)*.