TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT
KHOA KỸ THUẬT CÔNG NGHỆ

HỆ QUẢN TRỊ CSDL

Chương 4

STORED PROCEDURE AND FUNCTION
(Thủ tục thường trú và hàm người dùng)

Nội dung

- Giới thiệu về T-SQL
- Stored Procedure là gì?
- Xây dựng một Stored Procedure
- Thực thi một Stored Procedure
- Function - Hàm người dùng

T-sql là gì?

- **Transact-SQL** (còn gọi là **T-SQL**) là một ngôn ngữ lập trình **database** hướng thủ tục độc quyền của **Microsoft** sử dụng trong **SQL Server**.
- Là ngôn ngữ SQL mở rộng dựa trên SQL chuẩn của ISO và ANSI được sử dụng trong SQL Server

- ISO: International Organization for Standardization
- ANSI: American National Standards Institute

3

Tìm hiểu các thành phần của T-sql

T-SQL được chia làm 3 nhóm:

- DDL: Ngôn ngữ định nghĩa dữ liệu
  - Create
  - Alter
  - Drop
- DCL: Ngôn ngữ điều khiển dữ liệu
  - Giant
  - Denny
  - Revoke
- DML: Ngôn ngữ thao tác dữ liệu
  - Select, Insert
  - Update, Delete

4

Cấu trúc khối lệnh trong T-SQL

- **T-SQL** tổ chức theo từng khối lệnh:
  - Một khối lệnh có thể lồng bên trong một khối lệnh khác
  - Một khối lệnh bắt đầu bởi **BEGIN** và kết thúc bởi **END**
  - Bên trong khối lệnh có nhiều lệnh

Cấu trúc khối lệnh:

```
1  BEGIN
2     -- Khai báo biến
3     -- Các câu lệnh T-SQL
4  END;
```


Biến trong sql server

- Khái niệm
  - Có 2 loại là biến cục bộ và biến hệ thống
  - **Biến cục bộ** là do người dung tự định nghĩa, có ý nghĩa trong một query batch, một thủ tục thường trú hoặc một hàm người dùng
  - **Biến hệ thống** do SQL SERVER định nghĩa sẵn, có phạm vi sử dụng trên toàn bộ hệ thống. Tên biến hệ thống bắt đầu bằng @@. Các biến này là **read-only**.

Biến trong sql server (tt)

SQL Server cung cấp 2 loại biến

❖ Biến toàn cục: Bắt đầu bằng @@

Ví dụ: @@CONNECTIONS, @@CPU_BUSY, @@ERROR…

❖ Biến cục bộ: Bắt đầu bằng @

Ví dụ: USE QLSV

-- Khai báo biến cục bộ:

Cú pháp: **Declare @Tên biến <Kiểu Dữ Liệu>**

Ví dụ:

- ✓Declare @Masv nvarchar(5)
- ✓Declare @Hocbong float
- ✓Declare @Sum float, @Count int

7

Biến trong sql server (tt)

❖ Gán giá trị cho biến:

- ➢ Set Ten_bien = Gia_tri
- ➢ Set Ten_bien = Ten_bien
- ➢ Set Ten_bien = Bieu_thuc
- ➢ Set Ten_bien = Ket_qua_truy_van
- ➢ Ví dụ:
  - ✓Set @Ma_lop = 'D15PM02'
  - ✓Set @Ma_lop = 'DH' + Year(@Ngay_tuyen)
  - ✓Set @Si_so_lop = (SELECT count(*) FROM SV)
- ➢ Mỗi từ khóa SET chỉ dùng gán cho một biến

8

Biến trong sql server (tt)

- Gán giá trị cho biến:
  - Có thể gán giá trị cho biến bằng mệnh đề SELECT
  - Ví dụ 1:
    - ✓Select @Ma_lop = 'D15PM02', @Si_so =30
  - Ví dụ 2:
    - ✓Select @Si_so =SiSo From Lop Where Malop ='D15PM02'
    - ✓Select @max= max(diemlan1), @min=min(diemlan1) from DIEMTHI where mahocphan='sql'

9

Biến trong sql server (tt)

- Gán giá trị cho biến:
  - Lưu ý:
    - ✓Khi gán giá trị cho biến thì giá trị phải cùng kiểu dữ liệu với biến
    - ✓Giá trị gán cho biến là kết quả câu truy vấn thì kết quả của câu truy vấn phải trả về đúng 1 giá tri.
  - Ví dụ:
    - ✓Set @Ho_ten = (Select HoTen From sinhvien) → lỗi: "Subquery returned more than 1 value. This is not permitted…"

10

Biến trong sql server (tt)

- In kết ra màn hình:

```
DECLARE  @KQ  INT
SET @KQ= 100
PRINT N'Kết quả là: '
PRINT @KQ
PRINT N'Kết quả là' + Convert(NVARCHAR(10),@KQ)
```

(Có thể dùng **SELECT** thay cho **PRINT)**

11

Toán tử

- Các loại toán tử
  - Số học: *, /, %, - , +
  - So sánh: =, <>, >, >=, <, <=
  - Nối chuỗi: +
  - Logic: AND, OR, NOT

12

Toán tử (tt)

- Ví dụ:

```
BEGIN
DECLARE @a INT, @b INT, @c FLOAT
SET @a=5
SET @b=4
SET @c=@a+@b
PRINT N'Tổng của a và b là:'+convert(nvarchar(40),@c)
SET @c=@a-@b
PRINT N'Hiệu của a và b là:'+convert(nvarchar(40),@c)
SET @c=@a*@b
PRINT N'Tích của a và b là:'+convert(nvarchar(40),@c)
SET @c=@a/@b
PRINT N'Thương của a và b là:'+convert(nvarchar(40),@c)
```

Messages

```
Tổng của a và b là:9
Hiệu của a và b là:1
Tích của a và b là:20
Thương của a và b là:1
```

Một số hàm sử dụng trong sql

- MAX, MIN, SUM, COUNT, AVG, ROUND
- LEN, LEFT, RIGHT, SUBSTRING, UPPER, LOWER
- DAY, MONTH, YEAR
- Hàm GETDATE: Lấy ngày hiện hành
  - Ví dụ:

```
SELECT GETDATE() AS [Ngày giờ hiện tại]
      , CONVERT (date, GETDATE()) AS [Ngày hiện tại]
      , CONVERT (time, GETDATE()) AS [Giờ hiện tại]
```

	Ngày giờ hiện tại	Ngày hiện tại	Giờ hiện tại
1	2009-09-29 06:43:32.233	2009-09-29	06:43:32.2330000

14

Một số hàm sử dụng trong sql

- Hàm DATEPART(Định dạng,'Tháng/Ngày/Năm'): Lấy 1 phần (ngày, tháng, năm, ngày trong tuần, ngày trong năm) của ngày
  - Ví dụ:

```
SELECT DATEPART(yyyy,'02/10/1982') AS N'Năm',
       DATEPART(mm,'02/10/1982') AS N'Tháng',
       DATEPART(dd,'02/10/1982') AS N'Ngày',
       DATEPART(DAYOFYEAR,'02/10/1982') AS N'Ngày trong năm',
       DATEPART(WEEKDAY,'02/10/1982') AS N'Ngày thứ'
```

Năm	Tháng	Ngày	Ngày trong năm	Ngày thứ
1982	2	10	41	4

Một số hàm sử dụng trong sql

- Hàm DATEDIFF(Định dạng, 'ngày bắt đầu', 'ngày kết thúc'): Lấy hiệu giữa hai mốc thời gian (ngày kết thúc) và (ngày bắt đầu)
  - Ví dụ:

```
Select DATEDIFF (dd,'2017-10-20', '2017-10-25') as
N'số ngày'
```

số ngày
5

Cấu trúc lệnh

- Cấu trúc If … Else
- Cấu trúc While
- Phát biểu Continue
- Phát biểu Break
- Phát biểu Return
- Cấu trúc Case
- Cấu trúc Try...Catch

17

Cấu trúc điều khiển

- **IF…ELSE**

IF Biểu thức

{câu lệnh hoặc nhóm lệnh được thực thi}

ELSE

{câu lệnh hoặc nhóm lệnh được thực thi}

Lưu ý:

- Trong Cấu trúc If...Else, nếu có từ hai lệnh trở lên thì phải đặt giữa hai từ khóa Begin và End.

18

Cấu trúc điều khiển

❖ **Ví dụ:**

```
-----IF...ELSE
DECLARE @gt char(1), @gioitinh varchar(20)
SET @gt='N'
    IF @gt='N'
        SET @gioitinh='Nam'
    ELSE
        SET @gioitinh='Nữ'
    PRINT @gioitinh
```

Messages

Nam

19

Cấu trúc điều khiển

❖ **CASE…WHEN**

❖ **Dạng 1:**

CASE <Biểu thức điều kiện>

WHEN <Giá trị 1> THEN <Kết quả 1>

WHEN <Giá trị 2> THEN <Kết quả 2>

…

ELSE <Kết quả khác>

END

20

CASE…WHEN

❖ **Ví dụ 1:**

```
DECLARE @t int, @thang nvarchar(10)
SET @t = 7
SET @thang=
    CASE @t
        WHEN 2 THEN N'hai'
        WHEN 3 THEN N'ba'
        WHEN 4 THEN N'tư'
        WHEN 5 THEN N'năm'
        WHEN 6 THEN N'sáu'
        WHEN 7 THEN N'bảy'
    END
SELECT @thang AS N'Tháng'
```

Results | Messages

Tháng
bảy

21

CASE…WHEN (tt)

❖ Ví dụ 2:

- Nhan_vien(MaNV,HoTen,NgaySinh, Phai). Cho biết những nhân viên đến tuổi về hưu biết rằng tuổi về hưu của nam là 60, nữ là 55)

```
Select * From Nhan_vien
Where datediff(yy,Ngaysinh,Getdate())
    >=      CASE Phai
                WHEN 'Nam' THEN 60
                WHEN 'Nu' THEN 55
            END
```

22

CASE…WHEN (tt)

❖ **Dạng 2:**

CASE

WHEN <biểu thức điều kiện 1 THEN <Kết quả 1>
WHEN <biểu thức điều kiện 2 THEN <Kết quả 2>
…
WHEN <biểu thức điều kiện n THEN <Kết quả n>
ELSE <Kết quả khác>

END

23

CASE…WHEN (tt)

❖ **Ví dụ:** Liệt kê thông tin nhập hàng bao gồm: Số phiếu nhập, Ngày nhập, tên vật tư, số lượng, nhận xét. Trong đó: nếu số lượng nhập >=1000 thì ghi là nhập nhiều, từ 500 trở lên thì ghi là nhập đủ, các trường hợp còn lại ghi nhập ít.

```
SELECT Sophieunhap,Ngaynhap,Tenvattu,Soluongnhap,Nhanxet=
(CASE
   WHEN Soluongnhap>=1000 THEN N'Nhập nhiều'
   WHEN Soluongnhap>=500 THEN N'Nhập đủ'
   ELSE N'Nhập ít'
 END)
FROM Vattu,Nhapvattu
WHERE Vattu.Masovt=Nhapvattu.Masovt
ORDER BY Soluongnhap DESC
```

Cấu trúc while

- **WHILE**

  WHILE Biểu thức

  Câu lệnh hoặc nhóm lệnh được thực thi

- Chú ý: Nếu trong các cấu trúc điều khiển là một nhóm lệnh thì phải đặt nhóm lệnh đó trong cặp từ khóa BEGIN … END

25

Cấu trúc while

- **Ví dụ 1:**

```
-----WHILE
DECLARE @n INT
SET @n=5
WHILE @n<10
BEGIN
    PRINT @N
    SET @n=@n+1
END
```

Messages

```
5
6
7
8
9
```

26

Cấu trúc while (tt)

- **Ví dụ 2:**
- Cho **SV(Masv:int, HoTen:nvarchar(30))**

- Yêu cầu chèn vào bảng sinh viên một sinh viên mới sao cho mã sinh viên của sinh viên chèn vào đảm bảo tính liên tục từ thấp đến cao
- Chẳng hạn ta có MaSV: 1,2,3,4,8,9,… → Sinh viên mới sẽ có mã là 5

27

Cấu trúc while (tt)

- **Ví dụ 2:**
- Cho **SV(Masv:int, HoTen:nvarchar(30))**

```
…..
Declare @count int
Set @count = 1
While exists (select * From SV   Where MaSV = @count)
set @count = @count +1
Insert into SV(MaSV, HoTen)
Values (@count, N'Lê Như Ngọc')
```

28

Cấu trúc while (tt)

- BREAK: Thoát khỏi vòng lặp WHILE
- RETURN: Chúng ta có thể dùng RETURN bất kỳ thời điểm nào để thoát khỏi thủ tục. Các phát biểu sau RETURN sẽ không được thực thi.
- CONTINOUS: Các câu lệnh thực thi sau từ khóa CONTINOUS trong WHILE điều được bỏ qua để chuyển qua vòng lặp mới.

29

Cấu trúc Try...Catch

- Cấu trúc Try...Catch trong SQL Server được dùng để bắt lỗi tương tự như trong C# và C++.
- Một nhóm các lệnh được đặt trong khối Try, nếu có một lỗi xuất hiện bên trong khối Try thì điều khiển được gởi đến một nhóm lệnh khác được đặt trong một khối Catch.
- Chỉ có từ phiên bản SQL Server 2005.

30

Cấu trúc Try...Catch (tt)

- Cấu trúc Try… Catch có cấu trúc như sau:

```
BEGIN TRY
    { <Các câu lệnh SQL>}
END TRY
BEGIN CATCH
    { <Các câu lệnh SQL>}
END CATCH
[ ; ]
```

31

Cấu trúc Try...Catch

- Một số thông tin về lỗi:
  - ERROR_NUMBER(): Trả về mã số lỗi.
  - ERROR_SEVERITY(): Trả về mức độ của lỗi.
  - ERROR_STATE(): Mã trạng thái của lỗi.
  - ERROR_PROCEDURE(): Trả về tên của thủ tục hay trigger xuất hiện lỗi.
  - ERROR_LINE(): Trả về số dòng bên trong thủ tục xuất hiện lỗi.
  - ERROR_MESSAGE(): Trả về dòng văn bản thông báo lỗi một cách đầy đủ.
- Các hàm này trả về Null nếu nó được gọi bên ngoài của khối Catch.

32

Cấu trúc Try...Catch (tt)

- Ví dụ: Điều khiển lỗi trong chia hai số

```
declare @thuong float, @sobichia float, @sochia float
Select @sobichia = 3, @sochia = 0
begin try
        set @thuong=@sobichia/@sochia
        print @thuong
end try
begin catch
        SELECT
                ERROR_NUMBER() AS ErrorNumber,
                ERROR_SEVERITY() AS ErrorSeverity,
                ERROR_STATE() AS ErrorState,
                ERROR_PROCEDURE() AS ErrorProcedure,
                ERROR_LINE() AS ErrorLine,
                ERROR_MESSAGE() AS ErrorMessage;
        Print N'Số chia bằng không'
end catch
```

33

Cấu trúc Try...Catch (tt)

- Ví dụ: Khi thực thi thủ tục trên: phepchia 3 , 0.Kết quả trình bày trong ngăn Results và ngăn Messages như sau:

Results | Messages

	ErrorNumber	ErrorSeverity	ErrorState	ErrorProcedure	ErrorLine	ErrorMessage
1	8134	16	1	phepchia	6	Divide by zero error encountered.

Results | Messages

```
(1 row(s) affected)
Số chia bằng không
```

34

Thủ tục lưu trữ - (Stored Procedure)

1. Giới thiệu
2. Phân loại thủ tục
3. Tạo thủ tục
4. Lời gọi thủ tục
5. Giá trị trả về của tham số trong thủ tục
6. Phát biểu điều khiển
7. Tham số với giá trị mặc định
8. Thủ tục với tham số Table
9. Sửa đổi thủ tục
10. Xoá thủ tục

35

Stored Procedure là gì?

- Là một nhóm các câu lệnh T-SQL đã được biên dịch từ trước (pre-compiled).
- Thực hiện một nhiệm vụ cụ thể
- Một Stored Procedure
  - có thể không chứa hoặc chứa nhiều tham số truyền vào;
  - có thể trả giá trị về thông qua tham số truyền cho thủ tục.

36

Stored Procedure là gì?

- Có 2 dạng Stored Procedure:
  - System Stored Procedure
  - User-defined Stored Procedure
- Các System Stored Procedure có sẵn khi chúng ta cài đặt SQL Server. Tất cả các System Stored Procedure đều bắt đầu bằng tiền tố sp_

37

Tại sao dùng stored procedure?

- Tăng tốc độ thực thi
- Mô đun hóa, dễ dàng gọi lại
- Dễ dàng nâng cấp, bảo mật.

38

Phân loại

Trong SQL Server có 3 nhóm thủ tục nội tại sau:

- **Nhóm thứ nhất** là do người dùng tạo ra. Gồm hai loại:
  - Loại thủ tục nội tại được người dùng tạo ra và lưu vào CSDL. Chúng chứa các phát biểu T-SQL.
  - Loại thứ hai được khai báo và tạo ra bằng ngôn ngữ lập trình .NET.

39

Phân loại (tt)

- **Nhóm thứ hai** là thủ tục nội tại hệ thống thực hiện các chức năng quản trị CSDL thường dùng. Các thủ tục này chứa trong CSDL Resource.
  - Danh sách các thủ tục nội tại hệ thống hiển thị trong ngăn System Stored Procedure
  - Thủ tục nội tại trong CSDL Resource luôn có tên với **tiền tố là sp_**. Do đó bạn không nên đặt tên thủ tục nội tại do mình tạo ra bằng tiền tố này.
- **Nhóm thứ ba** là thủ tục nội tại hệ thống mở rộng. Loại này cũng được lưu trong CSDL Resouce nhưng có tên bắt đầu với xp_.

40

Tạo một sroted procedure

Cú pháp:

CREATE PROC[EDURE] <tên thủ tục> [(<DSách tham số>)]
AS
BEGIN
    [DECLARE <biến cục bộ >]
    <Các câu lệnh của thủ tục>
END

41

Tạo một sroted procedure (tt)

CREATE PROC[EDURE] <tên thủ tục> [(<DSách tham số>)]
AS
[BEGIN]
    [DECLARE <biến cục bộ >]
    <Các câu lệnh của thủ tục>
[END]

Ví dụ 1:

```
CREATE PROCEDURE TinhTong (@a int, @b int)
AS
Begin
    Declare @kq int
    Set @kq = @a + @b
    Print @kq
End
```

42

Tạo một sroted procedure (tt)

```
CREATE PROC[EDURE] <tên thủ tục> [(<DSách tham số>)]
AS
[BEGIN]
    [DECLARE <biến cục bộ >]
    <Các câu lệnh của thủ tục>
[END]
```

Ví dụ 2:

```
CREATE PROCEDURE sp_NVP (@mp int)
AS
SELECT * FROM NhanVien WHERE Maphong=@mp
```

43

Tạo một sroted procedure (tt)

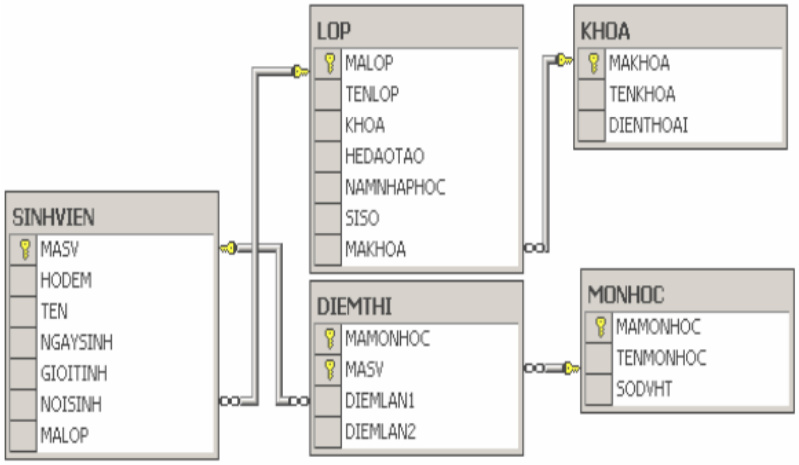
❖ **Ví dụ 3**: Viết thủ tục in ra tổng của các số từ 1..n.

```
CREATE PROCEDURE Tinhtong (@n int)
AS
    Declare @tong int, @i int
    Select @tong=0, @i=0
    While @i<=@n
        Begin
            Set @tong=@tong+@i
            Set @i=@i+1
        End
    Print @tong
GO
```

44

Tạo một sroted procedure (tt)

❖ **Ví dụ 4**: Cho CSDL quản lý điểm thi như



45

Tạo một sroted procedure (tt)

- Giả sử ta cần thực hiện một chuỗi các thao tác trên cơ sở dữ liệu
  - 1. Bổ sung thêm môn học *cơ sở dữ liệu* có mã CST*005* và số đơn vị học trình là *5* vào bảng MONHOC
  - 2. Lên danh sách nhập điểm thi môn *cơ sở dữ liệu* cho các sinh viên học lớp có mã *CDT002* (bổ sung thêm vào bảng DIEMTHI các bản ghi với cột MAMONHOC nhận giá trị *CST005*, cột MASV nhận giá trị lần lượt là mã các sinh viên học lớp có mã *CDT002* và các cột điểm là NULL).

46

Tạo một sroted procedure (tt)

- Theo cách thông thường ta sẽ viết 2 lệnh như sau:

```
INSERT INTO monhoc
        VALUES('CST005', N'Cơ sở dữ liệu',5)
```

-
```
INSERT INTO diemthi(mamh,masv)
        SELECT 'CST005', masv
                FROM sinhvien
                WHERE malop = 'CDT002'
```

Đây là mã môn học, đã cho trước nên ghi cụ thể ra luôn

- Lưu ý: ở đây bạn có cú pháp câu lệnh chèn dữ liệu vào một bảng có tên banga với dữ liệu lấy từ bảng có tên bangb khác như sau:

```
INSERT INTO banga (cot1, cot2)
        SELECT cot1, cot2 FROM bangb
```

47

Tạo một sroted procedure (tt)

- Thay vì phải viết 2 câu lệnh như trên, ta có thể định nghĩa một thủ tục với các tham số sau @mamh, @tenmh, @sodvht, @malop để nhập dữ liệu cho một môn học bất kỳ và một lớp bất kỳ do người dùng nhập vào khi sử dụng thủ tục.

48

Tạo một sroted procedure (tt)

```
CREATE PROC sp_LenDanhSachDiem(
        @mamh   NVARCHAR(10),
        @tenmh   NVARCHAR(50),
        @sodvht          SMALLINT,
        @malop   CHAR(4))
AS
BEGIN
    INSERT INTO monhoc
       VALUES(@mamh,@tenmh,@sodvht)
    INSERT INTO diemthi(mamh,masv)
       SELECT @mamh,masv
       FROM sinhvien
       WHERE malop=@malop
  END
```

- Khi thủ tục trên đã được tạo ra, thực hiện được hai yêu cầu trên qua lời gọi thủ tục:
  - **sp_LenDanhSachDiem 'CST005','Cơ sở dữ liệu',5,'L002'**

49

Thực thi stored procedure

- Thực thi thủ tục bằng lời gọi thủ tục có dạng:
  Tên_thủ_tục [danh_sách_các_đối_số]

- Lời gọi thủ tục được thực hiện bên trong một thủ tục khác, bên trong một trigger hay kết hợp với các câu lệnh SQL khác, sử dụng cú pháp :
  EXECUTE|EXEC tên_thủ_tục [danh_sách_các_đối_số]

50

Thực thi stored procedure (tt)

Cú pháp:
EXEC[UTE] <tên_thủ_tục> [<danh sách các đối số>]

Số lượng các đối và thứ tự của chúng phải phù hợp với số lượng và thứ tự của các tham số hình thức

Ví dụ 1: **EXEC sp_NVP 2**

Ví dụ 2: **EXECUTE sp_LenDanhSachDiem 'CST005','Cơ sở dữ liệu',5, 'CT002'**

Nếu truyền các đối không theo thứ tự thì tất cả các đối đều phải viết dưới dạng:

@tên_tham_số = giá_trị

51

Tham số trong stored procedure

- Tham số nhập
  - Là tham số phải truyền giá trị vào khi gọi thủ tục
- Tham số xuất (giữ lại giá trị của đối số sau khi kết thúc thủ tục)
  - ✓ Là tham số nhận giá trị mà thủ tục cần trả về,
  - ✓ Cách xây dựng tham số xuất:

    @ <Tên_tham_số> <Kiểu_dữ_liệu> OUT[PUT]
- Và trong lời gọi thủ tục, sau đối số được truyền cho thủ tục, bạn cũng phải chỉ định thêm từ khoá OUTPUT (hoặc OUT)

52

Tham số trong stored procedure (tt)

Ví dụ:

- CREATE PROCEDURE spCong_hai_so (@a INT,@b INT, @c INT)
  AS
  SELECT @c=@a+@b
  GO
- Và lời gọi thực thi
  DECLARE @tong INT
  SELECT @tong=0
  EXECUTE spCong_hai_so 100,200,@tong
  Print @tong

- CREATE PROCEDURE spCong_hai_so (@a INT,@b INT, @c INT OUT)
  AS
  SELECT @c=@a+@b
  GO
- Và lời gọi thực thi:
  DECLARE @tong INT
  SELECT @tong=0
  EXECUTE spCong_hai_so 100,200,@tong OUT
  Print @tong

53

Tham số với giá trị mặc định

- Giá trị mặc định sẽ được gán cho tham số trong trường hợp không truyền đối số cho tham số khi có lời gọi đến thủ tục.
- Cú pháp:
  - @tên_tham_số <kiểu_dữ_liệu> = giá_trị_mặc_định
- Ví dụ: Viết thủ tục spTestDefault như sau:

54

Tham số với giá trị mặc định (tt)

```
CREATE PROC spTestDefault(@tenlop NVARCHAR(30) =NULL,
@noisinh NVARCHAR(100) ='Huế')
AS
        IF @tenlop IS NULL
                SELECT hodem,ten  FROM sinhvien INNER JOIN lop
        ON sinhvien.malop=lop.malop
                WHERE noisinh like ('%'+@noisinh)
        ELSE
                SELECT hodem,ten FROM sinhvien INNER JOIN lop
        ON sinhvien.malop=lop.malop
                WHERE noisinh like ('%'+@noisinh) AND
                tenlop=@tenlop
GO
```


Tham số với giá trị mặc định (tt)

- Thực hiện các lời gọi thủ tục với các mục đích khác nhau như sau:
  - Cho biết họ tên của các sinh viên sinh tại Huế:
    - ✓spTestDefault
  - Cho biết họ tên của các sinh viên lớp dữ liệu 2 sinh tại Huế:
    - ✓spTestDefault @tenlop=N'dữ liệu 2'

Tham số với giá trị mặc định (tt)

- Thực hiện các lời gọi thủ tục với các mục đích khác nhau như sau:
  - Cho biết họ tên của các sinh viên sinh tại Quảng nam
    - ✓spTestDefault @noisinh=N'Quảng nam'
  - Cho biết họ tên của các sinh viên lớp đồ họa 3 sinh tại Đà Nẵng:
    - ✓spTestTefault @tenlop=N'Đồ họa 3', @noisinh=N'Đà Nẵng'

57

Tham số Table

Ví dụ 1:

```
CREATE PROC usp_StoreTable
AS
BEGIN
    DECLARE @table TABLE (  MaGV NCHAR(10),
                            HoTen nvarchar(50))
    INSERT @table ( MaGV, HoTen )
    SELECT  MaGV, TenGV
    FROM    dbo.GIAOVIEN

    SELECT * FROM @table
END
```

58

Tham số Table (tt)

- Vấn đề:
  - Ứng dụng cho người dùng nhập đơn hàng trên Form (bao gồm thông tin chung và thông tin riêng của từng chi tiết đơn hàng).
  - Vì tính an ninh dữ liệu, ứng dụng không thể trực tiếp insert dữ liệu vào các table mà phải thông qua các stored procedure (SP)
  - Làm sao truyền dữ liệu trên form vào SP?
- Giải pháp:
  - Dùng SP có tham số là table (chỉ có từ phiên bản SQL Server 2008)

59

Tham số Table (tt)

- **Ví dụ 2:**
  - Cho lược đồ dữ liệu như sau:
    - ✓**Hoadon(MaHD,MaKH,NgayLap,TongTien)**
    - ✓**CTHOADON(MaHD,MaSP,SoLuong,DonGiaBan,ThanhTien)**
  - **Bước 1: Tạo kiểu dữ liệu bảng tạm để chứa các đơn hàng từ Form truyền xuống:**

```
CREATE TYPE tbTamCTHD AS TABLE
( HD varchar(10),
  SP varchar(10),
  SL float, DG float,
  TT float )
```

60

Tham số Table (tt)

- **Bước 2: Viết thủ tục nhận tham số là kiểu dữ liệu bảng tạm tbTamCTHD để đổ dữ liệu từ bảng tạm vào bảng thực sự tên CSDL**:

```
CREATE PROC uspNhapHD (@MaHD varchar(10), @Makh
varchar(10), @Ngay Date, @CT tbTamCTHD ReadOnly)
AS
  Declare @tt float
  Set @tt = (Select Sum(SL*DG) From @CT)
  Insert Into HoaDon
        Values @MaHD, @MaKH, @Ngay, @tt
  Update @CT Set HD = @MaHD, TT = SL * DG
  Insert Into CTHOADON
        Select * From @CT
GO
```


Tham số Table (tt)

- **Bước 3: Sử dụng thủ tục vừa viết**

```
Declare @CT tbTamCTHD
Insert Into @CT Values (…)
Insert …
EXEC uspNhapHD ‘HD001’, ‘KH001’, ‘2015-8-31’, @CT
GO
```

Thay đổi, xóa stored procedure

- Thay đổi: Alter Proc <Tên thủ tục>
- Xóa: Drop Proc <Tên thủ tục>

63

HÀM

- Hàm là đối tượng cơ sở dữ liệu tương tự như thủ tục.
- Điểm khác biệt giữa hàm và thủ tục:
  - Hàm trả về một giá trị thông qua tên hàm còn thủ tục thì không.

64

HÀM

Hàm gồm 3 loại:

- Hàm trả về một giá trị có kiểu cơ sở như int, varchar, float, datetime, …
- Hàm trả về một table (tương tự view), thường trong làm là một câu truy vấn lấy dữ liệu.
- Hàm trả về một table được định nghĩa cấu trúc ngay trong hàm và dữ liệu có được nhờ thực hiện một dãy các thao tác insert

65

HÀM TRẢ VỀ MỘT GIÁ TRỊ

Cú pháp

```
CREATE FUNCTION tên_hàm ([danh_sách_tham_số])
RETURNS (kiểu_của_giá_trị_trả_về)
AS
BEGIN
        Thân hàm
        Return <giá trị trả về>

END
```

66

Ví dụ
Hàm trả về một giá trị

```
--Tạo hàm fTuoi
Create function fTuoi (@ns int)
Returns int
As
Begin
        return year(getdate()) - @ns
end
go

--Biên dịch hàm với F5
--Kiểm tra thử hàm
print dbo.fTuoi(1982) --phải có dbo.
```

67

Ví dụ
Hàm trả về một giá trị

```
CREATE FUNCTION thu (@ngay DATETIME)
RETURNS NVARCHAR(10)
AS BEGIN
DECLARE @tt NVARCHAR(10)
SELECT @tt=
  CASE DATEPART(DW,@ngay)
      WHEN 1 THEN N'Chủ nhật'
      WHEN 2 THEN N'Thứ hai'
      WHEN 3 THEN N'Thứ ba'
      WHEN 4 THEN N'Thứ tư'
      WHEN 5 THEN N'Thứ năm'
      WHEN 6 THEN N'Thứ sáu'
      ELSE N'Thứ bảy'
  END
RETURN (@tt)/* Giá Trị trả về của hàm*/
END
```

select dbo.thu('27/09/2017')

68

Ví dụ
Hàm trả về một giá trị

❖ Viết hàm tính số ngày của một tháng.

69

Hàm trả về một bảng (tương tự view)

Cú pháp

```
CREATE FUNCTION <tên_hàm> ([danh sách tham số]) RETURNS TABLE
AS
RETURN (câu lệnh select)
```

Chú ý:

- Kiểu trả về của hàm được chỉ định bởi mệnh đề RETURNS TABLE.
- Trong phần thân của hàm chỉ có duy nhất một câu lệnh RETURN xác định giá trị trả về của hàm thông qua duy nhất một câu lệnh SELECT (không sử dụng bất kỳ câu lệnh nào khác trong phần thân thân hàm)

70

Ví dụ
hàm trả về một bảng

```
CREATE FUNCTION FXemThongTinNCC(@dc nvarchar(30))
RETURNS TABLE
AS
RETURN
    SELECT cc.MaCC, cc.TenCC, cc.von, cc.Diachi,
           ch.MaH, Soluong*Dongia as thanhtien
    FROM Chuyenhang ch join NhaCC cc
                        On ch.MaCC =cc.MaCC
    WHERE cc.Diachi=@dc
--gọi hàm
Select * From FXemThongTinncc('TP HCM')

-- Khi gọi hàm có kiểu trả về là 1 bảng ta dùng cú
pháp SELECT * FROM Tên_Hàm
```


Ví dụ
Hàm trả về một bảng

VD 2: Viết hàm xem điểm trung bình của một sinh viên bất kỳ

```
CREATE FUNCTION XEMDTB( @MaSV int)
RETURNS TABLE
AS
RETURN
  SELECT student.StudentID, StudentName, AVG(mark) AS DTB
  FROM Student join Mark on Student.StudentID = Mark.StudentID
  WHERE Mark.StudentID=@MaSV
  GROUP BY student.StudentID, StudentName
```

- --GỌI HÀM
- SELECT * FROM dbo.XEMDTB(4)

Hàm trả về một bảng được định nghĩa trong hàm

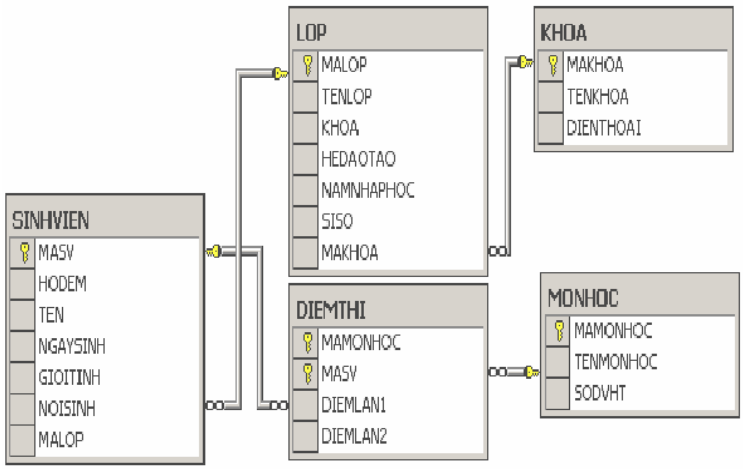
Cú pháp:

```
CREATE FUNCTION <tên_hàm> ([danh sách tham số])
RETURNS <@biến bảng> TABLE (Định nghĩa bảng)
AS
BEGIN
        Các câu lệnh trong thân hàm
RETURN
END
```

73

Ví dụ: hàm trả về một bảng được định nghĩa trong hàm

- Ví dụ: Cho CSDL quản lý điểm thi như sau:



74

Ví dụ: hàm trả về một bảng được định nghĩa trong hàm

```
CREATE FUNCTION Func_Tongsv(@khoa SMALLINT) RETURNS
  @bangthongke TABLE (
    makhoa      NVARCHAR(5),
    tenkhoa     NVARCHAR(50),
    tongsosv    INT
  ) AS
  BEGIN
    IF @khoa=0
      INSERT INTO @bangthongke
      SELECT khoa.makhoa,tenkhoa,COUNT(masv)
        FROM (khoa INNER JOIN lop
        ON khoa.makhoa=lop.makhoa)  INNER JOIN sinhvien
        ON lop.malop=sinhvien.malop
        GROUP BY khoa.makhoa,tenkhoa
```

75

Ví dụ: hàm trả về một bảng được định nghĩa trong hàm

```
ELSE
  INSERT INTO @bangthongke
        SELECT khoa.makhoa,tenkhoa,COUNT(masv)
        FROM (khoa INNER JOIN lop
                ON khoa.makhoa=lop.makhoa)
                        INNER JOIN sinhvien
                                ON lop.malop=sinhvien.malop
        WHERE khoa=@khoa
        GROUP BY khoa.makhoa,tenkhoa
    RETURN /*Trả kết quả về cho hàm*/
  END
```

76

Ví dụ: hàm trả về một bảng được định nghĩa trong hàm

- Gọi hàm:
  - SELECT * FROM Func_TongSV(5) → cho biết tổng số sinh viên khóa 5 của mỗi khoa
  - SELECT * FROM Func_TongSV(0) → cho biết tổng số sinh viên (tất cả các khóa) của mỗi khoa

77

Tài liệu tham khảo

- Slide Bài giảng Hệ quản trị Cơ sở dữ liệu, khoa CNTT Đại học Thủ Dầu Một
- Mike Chapple, Microsoft SQL Server 2012 for Dummies, Wiley, 2013.
- Tạ Thị Thu Phượng, Hệ quản trị cơ sở dữ liệu (Bài giảng tóm tắt), Đại học Đà Lạt.

Phone: 0650. 3834930 **Website**: www.fit.tdmu.edu.vn