Trường Đại học Khoa học Tự nhiên

Khoa Công nghệ Thông tin

Bộ môn Hệ thống Thông tin

Quản trị Cơ sở Dữ liệu

Chương III :
TRUY VẤN NÂNG CAO

GV : TUẤN NGUYÊN HOÀI ĐỨC

E-mail : tnhduc@fit.hcmus.edu.vn

Nội dung trình bày

- Khai báo biến và gán biến
- Cấu trúc điều khiển
- Thủ tục thường trú
- Kiểu dữ liệu cursor
- Hàm người dùng

Khai báo biến

- Cú pháp :
  - ***Declare*** *Var_name Datatype*
  - *Lưu ý tên biến : Tên biến phải bắt đầu bằng 1 ký tự @ và chỉ 1 mà thôi.*
- Ví dụ :
  - ***Declare*** *@MaSinhVien nvarchar(10)*
  - ***Declare*** *@TienLuong float*
  - ***Declare*** *@Sum float, @Count int*

Khai báo biến

- Tầm vực biến
  - ***Biến cục bộ có ý nghĩa trong một querry batch hay một thủ tục thường trú hoặc một hàm người dùng***
  - ***Biến hệ thống có ý nghĩa trên cả hệ thống. Tên của chúng bắt đầu bằng @@. Các biến này là read-only.***
  - ***Ví dụ biến hệ thống : @@fetch_status, @@rowcount, @@trancount…***

Lệnh gán

- **Set** TenBien = GiaTri
- **Set** TenBien = TenBien
- **Set** TenBien = BieuThuc
- **Set** TenBien = KetQuaTruyVan
- Ví dụ :

Set @MaLop = 'TH2001'

Set @SoSV = (select count(*) from SinhVien)

Set @MaLop = 'TH'+Year(@NgayTuyenSinh)

Câu truy vấn phải trả ra đúng 1 dòng và dòng đó phải có đúng 1 cột

Lệnh gán

- Cũng có thể gán giá trị cho biến bằng câu truy vấn thay vì chỉ thị **set**
- Ví dụ 1 :
  - ***SV(MaSV, HoTen, Tuoi)***
  - ***Select @Var2 = HoTen, @Var1 = Tuoi from SV where MaSV = 1***
- Ví dụ 2:
  - ***Declare @hoten nvarchar(40)***
  - ***set @hoten = (select hoten from sinhvien)***
  - ***print @hoten***

Kiểu dữ liệu phải tương ứng. Nếu câu truy vấn trả về nhiều dòng thì các biến chỉ nhận giá trị từ dòng đầu tiên

Subquery returned more than 1 value. This is not permitted when the subquery follows =, !=, <, <= , >, >= or when the subquery is used as an expression.

Nội dung trình bày

- Khai báo biến và gán biến
- Cấu trúc điều khiển
- Thủ tục thường trú
- Kiểu dữ liệu cursor
- Hàm người dùng

Cấu trúc điều khiển

- Cấu trúc if - Else

If ***logial expression***
[Begin]
    Code block
[End]
Else
[Begin]
    Code block
[End]

Có thể chứa các câu truy vấn phức tạp tùy ý

- Khai báo biến
- Các tính toán trên biến
- Các câu truy vấn phức tạp tùy ý
- …

Optional

Cấu trúc điều khiển

- Cấu trúc if - Else

*If **logial expression***
[Begin]
Code block
[End]
*[Else if **logial expression***
[Begin]
Code block
[End]
[,…n]]
Else
[Begin]
Code block
[End]

Có thể lặp lại nhiều lần tùy ý. Mô phỏng cấu trúc case

Cấu trúc điều khiển

- ***Ví dụ :***
  - ***HocPhan*** *(MaHP, TenHP, SiSo)*
  - ***DangKy*** *(MaSV, MaHP)*
  - **Viết lệnh để thêm một đăng ký mới cho sinh viên có mã số 001 vào học phần HP01 (giả sử học phần này đã tồn tại trong bảng HocPhan). Qui định sỉ số lớp cho mỗi học phần không quá 50 sv**

Cấu trúc điều khiển

- ***Ví dụ***

```
Declare @SiSo int
Select @SiSo = SiSo
From HocPhan Where MaHP= 'HP01'
If @SiSo < 50
Begin
        Insert into DANGKY(MaSV, MaHP)
        Values('001', 'HP01')
        Print N'Đăng ký thành công'
End
Else
        Print N'Học phần đã đủ SV'
```

Cấu trúc điều khiển

- Cấu trúc while

WHILE **Logical_expression**

[Begin]

{ sql_statement | statement_block }

[BREAK]

{ sql_statement | statement_block }

[CONTINUE]

[End]

Ngưng hẳn vòng lặp, thoát ra và thực hiện các lệnh kế tiếp sau vòng lặp

Ngưng hẳn lần lặp hiện hành, chuyển sang thực hiện ngay lần lặp kế tiếp

Cấu trúc điều khiển

- Ví dụ
  - *SinhVien(MaSV: int, HoTen: nvarchar(30))*
  - **Viết lệnh xác định một mã sinh viên mới theo qui định: mã sinh viên tăng dần, nếu có chỗ trống thì mã mới xác định sẽ chèn vào chỗ trống đó**
  - ***Chẳng hạn : 1,2,3,7 → mã sinh viên mới : 4***

Cấu trúc điều khiển

- Ví dụ

```
Declare @STT int
While exists (select * from SV
                    where MaSV = @STT)
    set @STT = @STT+1
Insert into SV(MaSV, HoTen)
values(@STT, 'Nguyen Van A')
```

Cấu trúc điều khiển

- Cấu trúc Case đơn giản

CASE **input_expression**

WHEN *when_expression*

THEN *result_expression*

[*...n*]

[**ELSE** *else_result_expression*]

END

Lệnh Case thường làm vế phải của lệnh gán, giá trị gán sẽ là *result_expression* khi mà *when_expression* được thỏa

Cấu trúc điều khiển

- Cấu trúc Case đơn giản
- Ví dụ :

```
SET @PLoai = CASE @loai
        WHEN 'pop_comp' THEN 'Popular Computing'
        WHEN 'mod_cook' THEN 'Modern Cooking'
        WHEN 'business' THEN 'Business'
        WHEN 'psychology' THEN 'Psychology'
        WHEN 'trad_cook' THEN 'Traditional Cooking'
        ELSE 'Not yet categorized'
END
```

Cấu trúc điều khiển

- ***Ví dụ :***
  - NHAN_VIEN(MaNV, HoTen, NgaySinh, CapBac, Phai)
    Cho biết những nhân viên đến tuổi về hưu (tuổi về hưu của nam là 60, của nữ là 55)

*Select * From NHAN_VIEN*

Where datediff(yy, NgaySinh, getdate())

> = ***Case Phai***

when 'Nam' then 60

when 'Nu' then 55

End

Cấu trúc điều khiển

- Cấu trúc Searched Case

CASE

WHEN *boolean_expression*

THEN *result_expression*

[...*n*]

[**ELSE** *else_result_expression*]

END

Lệnh Case thường làm vế phải của lệnh gán, giá trị gán sẽ là *result_expression khi mà boolean_expression được thỏa*

Cấu trúc điều khiển

- Cấu trúc Searched Case
- Ví dụ :

```
Set @DanhGiaMucGia = CASE
        WHEN @Gia IS NULL THEN 'Not yet priced'
        WHEN price < 10 THEN 'Very Reasonable Title'
        WHEN price >= 10 and price < 20 THEN 'Coffee Table'
        ELSE 'Expensive book!'
END
```

Cấu trúc điều khiển

- Ví dụ
  - ***Cho biết mã NV, họ tên và loại nhân viên (cấp bậc <=3:bình thường, cấp bậc = null: chưa xếp loại, còn lại: cấp cao)***

```
Select MaNV, HoTen, 'Loai' =
Case
when CapBac<=3 then 'Binh Thuong'
when CapBac is null then 'Chua xep loai'
else 'Cap Cao'
End
From NhanVien
```

Nội dung trình bày

- Khai báo biến và gán biến
- Cấu trúc điều khiển
- Thủ tục thường trú
- Kiểu dữ liệu cursor
- Hàm người dùng

Thủ tục thường trú

- Khái niệm
  - ***Công việc lập trình luôn đòi hỏi khả năng tái sử dụng mã lệnh.***
  - ***Không những vậy, các đoạn mã lệnh được tái sử dụng còn phải có tính uyển chuyển, xử lý linh động theo từng tình huống sử dụng.***
  - ***Từ đó xuất hiện khái niệm lập trình hướng thủ tục (functional). Các thủ tục được gọi lại và điều khiển thông qua hệ thống tham số***

Thủ tục thường trú

- Khái niệm
  - ***Việc lập trình trên CSDL cũng không ngoại lệ. Trong môi trường SQL Server, các thủ tục được gọi là Thủ tục thường trú (stored procedure – SP)***
  - ***Thường trú : Chỉ dịch 1 lần, từ đó lưu trữ bền vững trong CSDL, bền vững tựa như các table. Bất cứ khi nào cần, ta chỉ việc gọi thực hiện.***

Thủ tục thường trú

- Ý nghĩa
  - ***Tính tái sử dụng, uyển chuyển nhờ hệ thống tham số.***
  - ***Khi biên dịch SP, SQL Server tối ưu hóa nó sao cho thực thi hiệu quả nhất. Kết quả tối ưu hóa được lưu bền vững. Khi gọi thực thi không cần tối ưu hóa lại → lới gọi thủ tục tiết kiệm thời gian và tài nguyên hơn khối lệnh tương đương thân thủ tục.***

Thủ tục thường trú

- Ý nghĩa
  - ***Ứng dụng triển khai theo môi trường client – server. Client gửi lời gọi SP lên server thì chiếm dụng đường truyền ít hơn rất nhiều so với việc gửi khối lệnh tương đương thân hàm → tránh nghẽn đường truyền, giảm trì trệ.***
  - ***Đóng gói chỉ các thao tác cho phép trên CSDL vào các SP và quy định truy xuất DL phải thông qua SP. Ngoài ra còn có thể phân quyền trên SP.***

Thủ tục thường trú

- Ý nghĩa
  - ***SP giúp việc kết xuất báo biểu bằng Crystal Report trở nên đơn giản và hiệu quả hơn rất nhiều so với việc kết xuất trực tiếp từ các table và view.***

Thủ tục thường trú

- Cú pháp

Create ***{****proc* **|** *procedure****}*** ***proc_name***

Parameter DataType ***[****output****] [,…n]***

As

Code block

Go

Tên của stored procedure

Tên tham số (đặt như tên biến)

SP không có giá trị return, giá trị trả ra nếu có thì dùng một (hay một số) tham số output

Thân sủa SP, viết như thế nào là tùy váo từng bài toán cụ thể

Kiểu DL của tham số

Thủ tục thường trú

- Ví dụ : Xây dựng SP cho biết danh sách sinh viên của một lớp có mã cho trước

```
Create proc DS_Lop @MaLop varchar(10)
As
    Select SV.MaSV, SV.HoVaTen, SV.NgaySinh
    From SinhVien SV where SV.Lop = @MaLop
Go
```

Sử dụng tham số truyền vào

Thủ tục thường trú

- Ví dụ :
  - ***Xây dựng SP tính toán giá trị cho đơn hàng có mã cho trước với quan hệ DonHang như sau :***
  - ***DonHang(Ma,SoLuong,DonGia,ThueSuat,Chiet Khau,ThanhTien)***

Thủ tục thường trú

- Ví dụ

```
Create proc TongTien
@MaDH varchar(10) = 'DH001'
As
    Declare @ThanhTien float
    Declare @TienThue float
    Declare @TienChietKhau float
    Declare @DonGia float,@SoLuong int
    Set @SoLuong = (select SoLuong from DonHang
      where Ma = @MaDH)
```

Giá trị default

Thủ tục thường trú

- Ví dụ

```
Set @DonGia = (select DonGia from DonHang
  where Ma = @MaDH)
Set @TienThue = (select ThueSuat from DonHang
  where Ma = @MaDH)
Set @TienChietKhau = (select ChietKhau from
  DonHang where Ma = @MaDH)
Set @ThanhTien = @DonGia*@SoLuong
Set @TienThue = @ThanhTien*@TienThue/100
Set @ThanhTien = @ThanhTien + @TienThue
```

Thủ tục thường trú

- Ví dụ

```
Set @TienChietKhau =
  @ThanhTien*@TienChietKhau/100
Set @ThanhTien = @ThanhTien - @TienChietKhau
Update DonHang set ThanhTien = @ThanhTien
  where Ma = @MaDH
Go
```

Thủ tục thường trú

- Ví dụ :
  - ***Xây dựng SP tính điểm trung bình và xếp loại cho sinh viên có mã cho trước. Giả sử các quan hệ như sau :***
    - SinhVien(MaSV,HoVaTen,DTB,XepLoai,Lop)
    - MonHoc(MaMH,TenMH)
    - KetQua(MaMH,MaSV,LanThi,Diem)
  - ***Điểm thi chỉ tính lần thi sau cùng***
  - ***Xếp loại : Xuất sắc [9,10], Giỏi [8,8.9], Khá [7,7.9], Trung bình [5.0,6.9], Yếu [0,4.9]***
  - ***Kết quả xuất dạng tham số output, không ghi xuống CSDL.***

Thủ tục thường trú

Create proc XepLoaiSV @MaSV varchar(10),
@DTB float output, @XL nvarchar(20) output
As

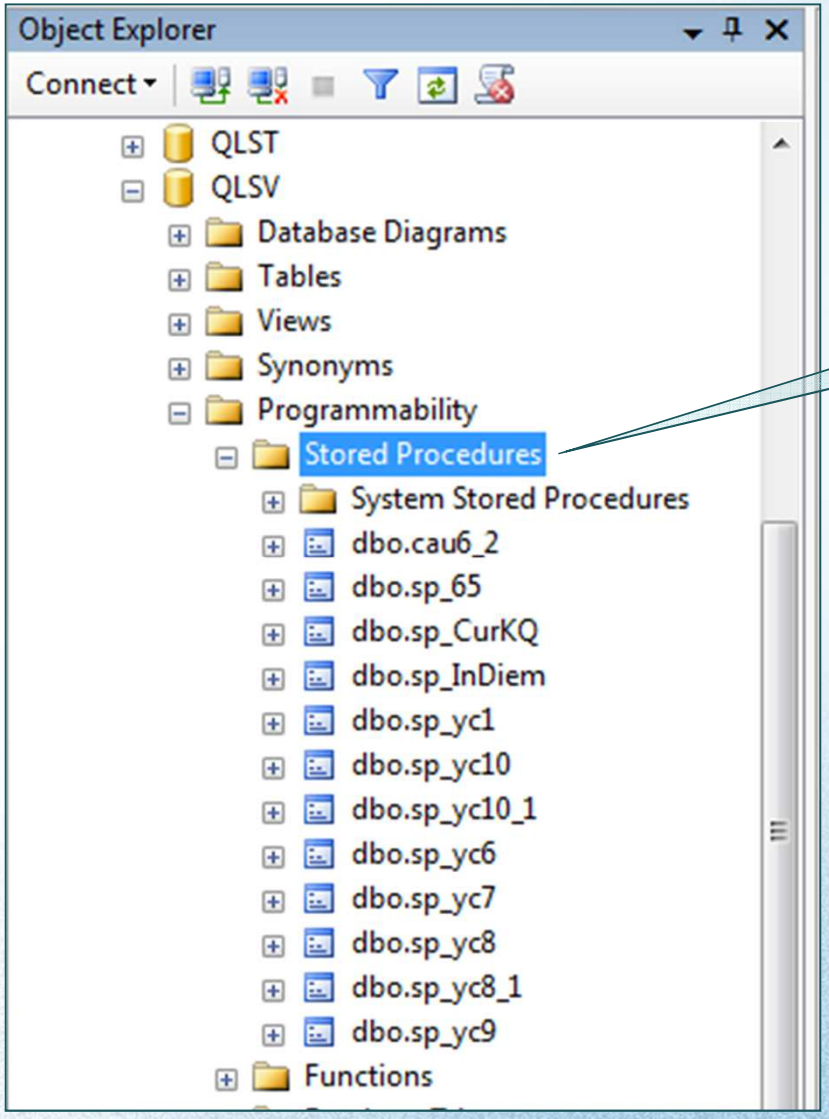
Tham số đầu ra

```
Set @DTB =
(Select avg(Diem) from KetQua Kq
Where MaSV = @MaSV and not exists
  (select * from KetQua Kq1 where Kq1.MaSV =
  @MaSV and Kq1.MaMH=Kq.MaMH and
  Kq1.LanThi > Kq.LanThi)
```

Thủ tục thường trú

```
If @DTB >= 9
    Set @XL = N'Xuất sắc'
Else if @DTB >= 8
    Set @XL = N'Giỏi'
Else if @DTB >= 7
    Set @XL = N'Khá'
Else if @DTB >= 5
    Set @XL = N'Trung bình'
Else
    Set @XL = N'Yếu'
Go
```

Thủ tục thường trú



Thư mục chứa thủ tục

Thủ tục thường trú

- Gọi thực hiện
  - ***Khi gọi thực hiện SP, dùng từ khóa Exec và cần truyền đủ tham số với kiểu DL và thứ tự chính xác như khai báo trong định nghĩa SP***
  - ***Tham số đầu vào có thể truyền* giá trị hằng *hay* biến *đã gán giá trị, không truyền một biểu thức***
  - ***Tham số đầu ra phải dùng* biến *và có từ khóa output, không cần khởi tạo giá trị SP sẽ điền giá trị tương ứng vào biến.***

Thủ tục thường trú

- Gọi thực hiện

```
Declare @MaSinhVien varchar(10)
Declare @DiemTB varchar(10)
Declare @XepLoai varchar(10)
Set @MaSinhVien = '0112357'
Exec XepLoaiSV @MaSinhVien,
   @DiemTB output,@XepLoai output
Exec XepLoaiSV '0213478',
   @DiemTB output,@XepLoai output
```

Tham số là biến đã gán giá trị '0112357'

Tham số là gía trị hằng

Các tham số output

Thủ tục thường trú

- Sửa thủ tục thường trú
  - *Alter procedure **Tên_TTục ...***
- Xóa thủ tục thường trú
  - *Drop procedure **Tên_TTục***
  - ***Ví dụ :***

    Drop procedure XepLoaiSV

Định nghĩa lại danh sách tham số và thân thủ tục (Giống khi tạo SP)

Thủ tục thường trú

- Bẫy lỗi
  - ***Begin Try***
    - Đoạn code có thể phát sinh lỗi
    - …
  - ***End Try***
  - ***Begin Catch***
    - Đoạn code xử lý lỗi phát sinh
    - …
  - ***End Catch***

Chỉ có với phiên Bản SQL Server 2005 Trở lên

Error_Severity()
Error_Number()
Error_Message()
Error_State()

Thủ tục thường trú

- Thủ tục nhận tham số table
  - ***Đặt vấn đề***
    - Trình ứng cụng 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, trình ứng dụng không thể trực tiếp insert DL vào các table mà phải thông qua SP.
    - Làm sao truyền DL trên form vào SP ?
  - ***Giải pháp***
    - Dùng thủ tục có tham số là table (chỉ có từ phiên bản 2008)

Thủ tục thường trú

- Thủ tục nhận tham số table
  - ***Ví dụ***
    - Cho các bảng DL như sau :

      DonHang(maDH,tenDH,ngayLap,tongTien)

      CTDonHang(maDH,maSP,SLDat,DonGiaDat, thanhTien)
    - B1 : Tạo kiểu DL bảng tạm để chứa các CTDonHang từ form truyền xuống.

```
CREATE TYPE CTDHTam AS TABLE
(DH varchar(10),SP varchar(10), SL float,
DG float, TT float)
GO
```

Thủ tục thường trú

- Thủ tục nhận tham số table
  - ***Ví dụ***
    - B2 : Viết thủ tục nhận tham số là kiểu bảng tạm CTDHTam để đổ DL từ bảng tạm này vào bảng thật CTDonHang.

```
Create proc usp_NhapDH
@maDH varchar(10), @tenDH nvarchar(50),
@ngay Date, @CT CDHTemp ReadOnly
As
    Declare @TT float
    Set @TT = (select sum(SL*DG) from @CT)
    Insert into DonHang
      values(@maDH,@tenDH,@ngay,@TT)
    Update @CT set DH=@maDH,TT=SL*DG
    Insert into CTDonHang select* from @CT
GO
```

Thủ tục thường trú

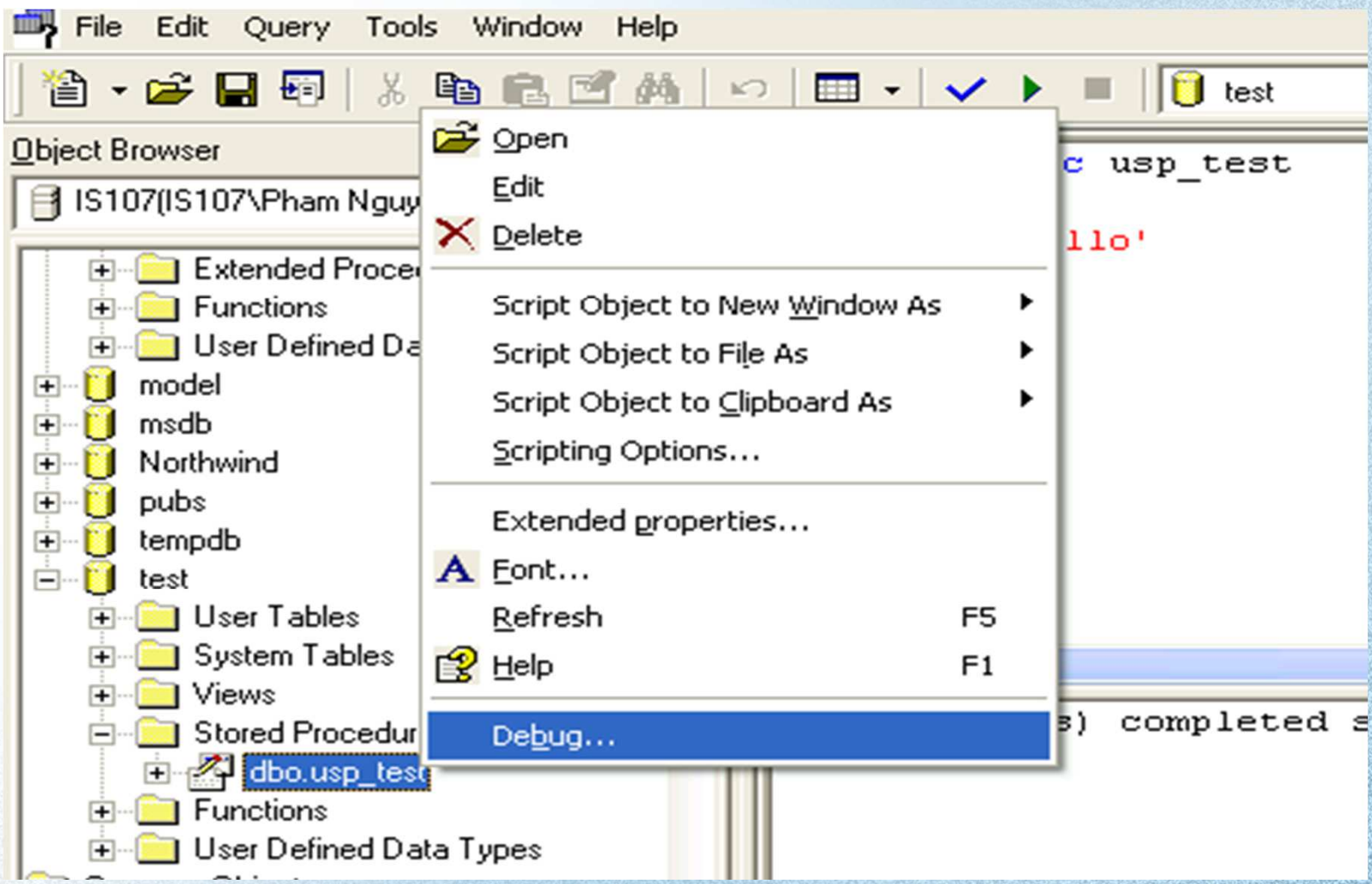
- Thủ tục nhận tham số table
  - ***Ví dụ***
    - B3 : Sử dụng thủ tục vừa viết

```
Declare @CT as CTDHTemp
Insert into @CT values(...)
Insert...
EXEC usp_NhapDH
'DH001',N'Đơn hàng cty Tiến Thịnh',
'2012-9-19', @CT
GO
```

Thủ tục thường trú

- Ngoài các SP do người dùng định nghĩa, SQL Server còn cung cấp các SP xây dựng sẵn của hệ thống
- Tên bắt đầu bằng sp_, SP người dùng tránh đặt tên bắt đầu bằng sp_
- Các SP này cung cấp tiện ích như quản lý câu trúc dữ liệu (xem chương II), quản lý khóa và giao tác, kết buộc đối tượng…
- Sinh viên tìm hiểu thêm về các SP này trong Books on-line và các tài liệu tham khảo

Thủ tục thường trú



Nội dung trình bày

- Khai báo biến và gán biến
- Cấu trúc điều khiển
- Thủ tục thường trú
- Kiểu dữ liệu cursor
- Hàm người dùng

Cursor

- Khái niệm
  - ***Là một cấu trúc dữ liệu ánh xạ đến một tập các dòng dữ liệu là kết quả của một câu truy vấn (select)***
  - ***Cho phép duyệt tuần tự qua tập các dòng dữ liệu và đọc giá trị từng dòng.***
  - ***thể hiện của cursor là 1 biến, nhưng tên biến này không bắt đầu bằng '@'***

Cursor

- Khái niệm
  - ***Vị trí hiện hành của cursor có thể được dùng như điều kiện trong mệnh đề where của lệnh update hoặc delete***
    - cho phép cập nhật/xoá dữ liệu (dữ liệu thật sự trong CSDL) tương ứng với vị trí hiện hành của cursor

Cursor

- Khai báo : dùng cú pháp chuẩn SQL 92 hoặc T_SQL mở rộng
  - ***Cú pháp SQL 92 chuẩn:***

Declare cur_name [**Insensitive**] [**Scroll**] **Cursor**

For select_statement

[**For** {**Read only**| **Update** [**of** ColName [,…n]] }]

Nội dung cursor không thay đổi dù DL thật thay đổi

Duyệt theo đa chiều

Chỉ đọc, không dùng cursor để cập nhật dữ liệu

Cho phép dùng cursor cập nhật dữ liệu trên 1 số (hoặc tất cả) cột

Cursor

- ***Cú pháp T_SQL mở rộng***

Declare cursor_name **Cursor**
[**Local** | **Global**]
[**Forward_only**| **Scroll**]
[**Static**| **Dynamic**]
[**Read_only**]
For select_statement
[**For Update** [**of** column_name [,…n]]]

Cục bộ, chỉ dùng được trong 1 khối lệnh, 1 SP hay 1 hàm

Toàn cục, tồn tại trong suốt kết nối đến khi bị hủy

Chỉ duyệt tới theo 1 chiều

Thay đổi khi DL thật thay đổi

Intensitive

Cursor

Mặc định:

- ***Global***
- ***Forward_only***
- ***Read only hay "for update" tùy thuộc vào câu truy vấn***
- ***Dynamic***

Cursor

- Duyệt cursor : Dùng lệnh Fetch để duyệt tuần tự qua cursor

  Fetch

  [[*Next****|*** *Prior****|*** *First****|*** *Last****|***

  Absolute ***n|*** *Relative* ***n]***

  From ***Tên_cursor***

  [*Into* ***Tên_biến [,…n]]***

Mặc định, bắt buộc đối với cursor forward-only

Biến tạm dùng để chứa phần tử muốn lấy ra từ cursor

Biến hệ thống
@@fetch_status
Cho biết đã duyệt Hết cursor chưa :
- Chưa hết :
@@fetch_status=0
- Vừa Hết :
@@fetch_status=0
- Đã Hết :
@@fetch_status≠0

Cursor

- Trình tự sử dụng :
  - ***Khai báo cursor***
  - ***"Mở" cursor bằng lệnh :*** *Open* ***tên_Cursor***
  - ***Khai báo các biến tạm để chứa phần tử hiện hành (đang được xử lý) của cursor***
    - Các biến tạm phải cùng kiểu dữ liệu với các trường tương ứng của phần tử trong cursor.
    - Có n trường trong phần tử của cursor thì phải có đủ n biến tạm tương ứng

Cursor

- Fetch (next,…) cursor để chuyển đến vị trí phù hợp
  - ***Có thể đưa các giá trị của dòng hiện hành vào các biến thông qua mệnh đề into của lệnh fetch***
  - ***Nếu không có mệnh đề into, các giá trị của dòng hiện hành sẽ được hiển thị ra cửa sổ kết quả (result pane) sau lệnh fetch***
  - ***Có thể sử dụng vị trí hiện tại như là điều kiện cho mệnh đề where của câu delete/ update (nếu cursor không là read_only)***
- Lặp lại việc duyệt và sử dụng cursor, có thể sử dụng biến @@fetch_status để biết đã duyệt qua hết cursor hay chưa.

Cursor

- Đóng cursor bằng lệnh **Close** Tên_cursor
- Sau khi đóng, vẫn có thể mở lại nếu cursor chưa bị hủy
- Hủy cursor bằng lệnh deallocate

  *Deallocate **Tên_cursor***

Cursor

- Ví dụ :

  SINHVIEN (MaSV, HoTen, MaKhoa)

  KHOA(MaKhoa, TenKhoa)

  - **Ví dụ 1:** ***Duyệt và đọc giá trị từ cursor***

    Cập nhật lại giá trị MaSV = Viết tắt tên Khoa + MaSV hiện tại cho tất cả sinh viên

Cursor

```
declare cur_DSKhoa cursor
for select MaKhoa, TenKhoa from Khoa
open cur_DSKhoa
declare @MaKhoa int,
        @TenKhoa varchar(30), @TenTat varchar(5)
fetch next from cur_DSKhoa into @MaKhoa,
@TenKhoa
```

Cursor

```
while @@fetch_status = 0
begin
        -- xác định tên tắt của Khoa dựa vào @TenKhoa…
        update   SinhVien set MaSV = @TenTat+MaSV
        Where MaKhoa = @MaKhoa
        fetch next from cur_DSKhoa
        into @MaKhoa, @TenKhoa
end
Close  cur_DSKhoa
Deallocate cur_DSKhoa
```

Cursor

- ***Ví dụ 2: dùng cursor để xác định dòng cập nhật***

```
declare cur_DSKhoa cursor scroll
for select MaKhoa, TenKhoa from Khoa
open cur_DSKhoa
fetch absolute 2 from cur_DSKhoa
if (@@fetch_status = 0)
        update Khoa
                set TenKhoa = ‘aaa’
                where current of cur_DSKhoa
Close  cur_DSKhoa
Deallocate cur_DSKhoa
```

Cursor

- Biến Cursor :
  - ***Ta có thể khai báo một biến kiểu cursor và gán cho nó tham chiếu đến một cursor đang tồn tại.***
  - ***Biến cursor có thể được xem như là con trỏ cursor***
  - ***Ví dụ :***

```
Declare @cur_var cursor
set @cur_var = my_cur
```

Một cursor nào đó đã khai báo từ trước và hiện đang tồn tại

Hoặc:

```
Declare @cur_var cursor
set @cur_var = cursor for select_statement
```

  - ***Biến cursor là một biến cục bộ***
  - ***Biến cursor sau khi gán giá trị được sử dụng như một cursor thông thường.***

Cursor

Ví dụ 3 : Kết hợp cursor với SP

- Xây dựng SP tính điểm trung bình và xếp loại cho sinh viên thuộc lớp cho trước. Giả sử các quan hệ như sau :
  - ***SinhVien(MaSV,HoVaTen,DTB,XepLoai,Lop)***
  - ***MonHoc(MaMH,TenMH)***
  - ***KetQua(MaMH,MaSV,LanThi,Diem)***
- Điểm thi chỉ tính lần thi sau cùng
- Xếp loại : Xuất sắc [9,10], Giỏi [8,8.9], Khá [7,7.9], Trung bình [5.0,6.9], Yếu [0,4.9]
- Kết quả ghi xuống CSDL, đồng thời xuất ra tổng số sinh viên xếp loại giỏi của lớp đó.

Cursor

- Phân tích ví dụ :
  - ***Lớp cần xét có nhiều sinh viên, từng sinh viên cần được xử lý thông qua 3 bước :***
    - Tính điểm trung bình cho sinh viên, điểm trung bình phải là điểm của lần thi sau cùng. Có thể tái sử dụng thủ tục XepLoaiSV
    - Dựa vào điểm trung bình của sinh viên để xác định xếp loại.
    - Cập nhật điểm và xếp loại vào bảng sinh viên
  - ***Mọi sinh viên đều lặp lại 3 bước trên***

Cursor

- Phân tích ví dụ : Như vậy ta thấy
  - ***Cần xử lý nhiều phần tử (các sinh viên).***
  - ***Mỗi phần tử xử lý tương đối phức tạp (truy vấn,tính toán, gọi thủ tục khác, điều kiện rẽ nhánh,cập nhật dữ liệu…)***
  - ***Cách xử lý các phần tử là như nhau***
- → Sử dụng cursor là thích hợp
  - ***Cursor chứa các sinh viên của lớp cần xét, chỉ cần chứa mã sinh viên là được***

Cursor

- Bắt đầu xây dựng SP:

```
Create proc XepLoaiSVLop
@Lop varchar(10), @SoSVGioi int output
As
    Declare @DTB float
    Declare @XepLoai nvarchar(20)
    Declare @MaSV varchar(10)
    Declare cur_SV cursor for
    (select MaSV from SinhVien where Lop = @Lop)
    Open cur_SV
```

Biến tạm dùng chứa phần tử hiện hành của cursor

Mở cursor

Cursor

```
Fetch next from cur_SV into @MaSV
While @@fetch_status=0
Begin
    Exec XepLoaiSV @MaSV,@DTB output,@XepLoai
      output
    Update SinhVien set DTB =@DTB,XepLoai=@XepLoai
    Where MaSV = @MaSV
    Fetch next from cur_SV into @MaSV
End
Close cur_SV
Deallocate cur_SV
Set @SoSVGioi = (slecte count(*) from SinhVien
Where Lop=@Lop and XepLoai = N‘Giỏi’)
Go
```

Lấy phần tử đầu tiên

Lấy phần tử tiếp theo

Đóng cursor

Giải phóng bộ nhớ chiếm bởi cursor

Nội dung trình bày

- Khai báo biến và gán biến
- Cấu trúc điều khiển
- Thủ tục thường trú
- Kiểu dữ liệu cursor
- Hàm người dùng

Hàm người dùng

- Hàm người dùng (function) cũng giống như SP
  - ***Là mã lệnh có thể tái sử dụng***
  - ***Chấp nhận các tham số input***
  - ***Dịch một lần và từ đó có thể gọi khi cần***
- Khác SP
  - ***Có giá trị trả về (một và chỉ một giá trị trả về)***
  - ***Không chấp nhận tham số output***

Hàm người dùng

- Lưu ý : Trong thân hàm không được sử dụng các hàm hệ thống bất định (Built-in nondeterministic functions), bao gồm :
  - ***GETDATE***
  - ***GETUTCDATE***
  - ***NEWID***
  - ***RAND***
  - ***TEXTPTR***
  - ***@@TOTAL_ERRORS, @@CPU_BUSY, @@TOTAL_READ, @@IDLE, @@TOTAL_WRITE, @@CONNECTIONS ...***

Hàm người dùng

- Có thể xem hàm người dùng thuộc về 3 loại tùy theo giá trị trả về của nó :
  - ***Giá trị trả về là kiểu dữ liệu cơ sở (int, varchar, float, datetime…)***
  - ***Giá trị trả về là Table có được từ một câu truy vấn (thuộc nhánh Table value function)***
  - ***Giá trị trả về là table mà dữ liệu có được nhờ tích lũy dần sau một chuỗi thao tác xử lý và insert (thuộc nhánh Table value function).***

Hàm người dùng

- Giá trị trả về là kiểu dữ liệu cơ sở (int, varchar, float, datetime…)

Create function ***func_name***

(parameter dataType [=default][,…n])

returns ***dataType***

As

Begin

Code block

Return {value | variable | expression}

End

Giá trị mặc định tham số

Kiểu DL trả về

Khai báo biến, xử lý tính toán, truy vấn …

Trả giá trị ra

Hàm người dùng

- Ví dụ

```
Create function SoLonNhat
(@a int,@b int,@c int) returns int
As
Begin
    declare @max int
    set @max = @a
    if @b > max set @max = @b
    if @c > max set @max = @c
    return @max
End
```

Dù không có tham số cũng phải ghi cặp ngoặc rỗng

Dù thân function chỉ có 1 lệnh cũng phải đặt giữa **Begin** và **End**

Hàm người dùng

- Giá trị trả về là Table có được từ một câu truy vấn

Create function ***func_name***

(parameter dataType [= default][,…n])

returns ***Table***

As

Return (Select… From… Where……)

Go

Kiểu DL trả về là **Table**

Thân function luôn chỉ có 1 lệnh, <u>không</u> được đặt giữa **Begin** và **End**

Hàm người dùng

- Ví dụ :

```
Create function DanhSachMatHang
(@MaDonHang varchar(10)) returns Table
As
    Return
    (Select MH.TenHang,MH.DonGia
    From ChiTietDH CT,MatHang MH
    Where CT.MaDH = @MaDonHang
    and CT.MaMH = MH.MaMH)
Go
```

Hàm người dùng

- Giá trị trả về là table mà dữ liệu có được nhờ tích lũy dần sau một chuỗi thao tác xử lý và insert.

```
Create function func_name
(parameter dataType [= default][,…n])
returns TempTab_name
Table(Table_definition)
As
Begin
    Code block
    Return
End
```

Tên của Table sẽ trả về

Định nghĩa các cột và khóa chính cho Table sẽ trả về

Tính toán và insert dần dần dữ liệu vào Table sẽ trả về

Hàm người dùng

- Ví dụ

```
Create function DanhSachLop
returns @DS
Table(MaLop varchar(10),SoSV int)
As
Begin
        Declare cur_L cursor for select Ma from Lop
        Declare @Ma varchar(10)
        Open cur_L
        Fetch next from cur_L into @Ma
        While @@fetch_status=0
        Begin
            ….
        End
        Close cur_L
        Deallcocate cur_L
        return
End
```

```
Insert into @DS
values
(@Ma,
(select count(*) from SinhVien
where Lop=@Ma))
Fetch next from cur_L into @Ma
```

Hàm người dùng

- Xóa hàm người dùng
  - *Drop function* ***Tên_Hàm_Cần_Xóa***
  - ***Ví dụ :***

    Drop function DanhSachMatHang

Hàm người dùng

- Chỉ thị Cross Apply
  - ***Dùng trong câu truy vấn, "kết" table trả về bởi hàm (dạng 2 và 3) với các table khác trong câu truy vấn thông qua các tham số của hàm.***
  - ***Ví dụ :***
    - Giả sử có hàm trả về @n nhan viên lương cao nhất của một phòng ban có mã cho trước :

```
Create function NVLC
(@MaPhg varchar(10),@n int)
returns table As return
Select top(@n) * from NhanVien
Where Phg = @MaPhg
Order by Luong
```

Chỉ có với phiên Bản SQL Server 2005 Trở lên

Hàm người dùng

- Chỉ thị Cross Apply
  - ***Ví dụ :***
    - Giả sử cần truy vấn tên phòng, tên trưởng phòng kèm theo 5 nhân viên lương cao nhất của mỗi phòng, trong 5 NV ấy chỉ in những NV có lương trên 300’000đ

Select P.tenphong,TP.hoVaTen

From PhongBan P, NhanVien TP

Cross apply NVLC(P.MaPhg,5) as Temp

where TP.MaNV = P.TrgPhg

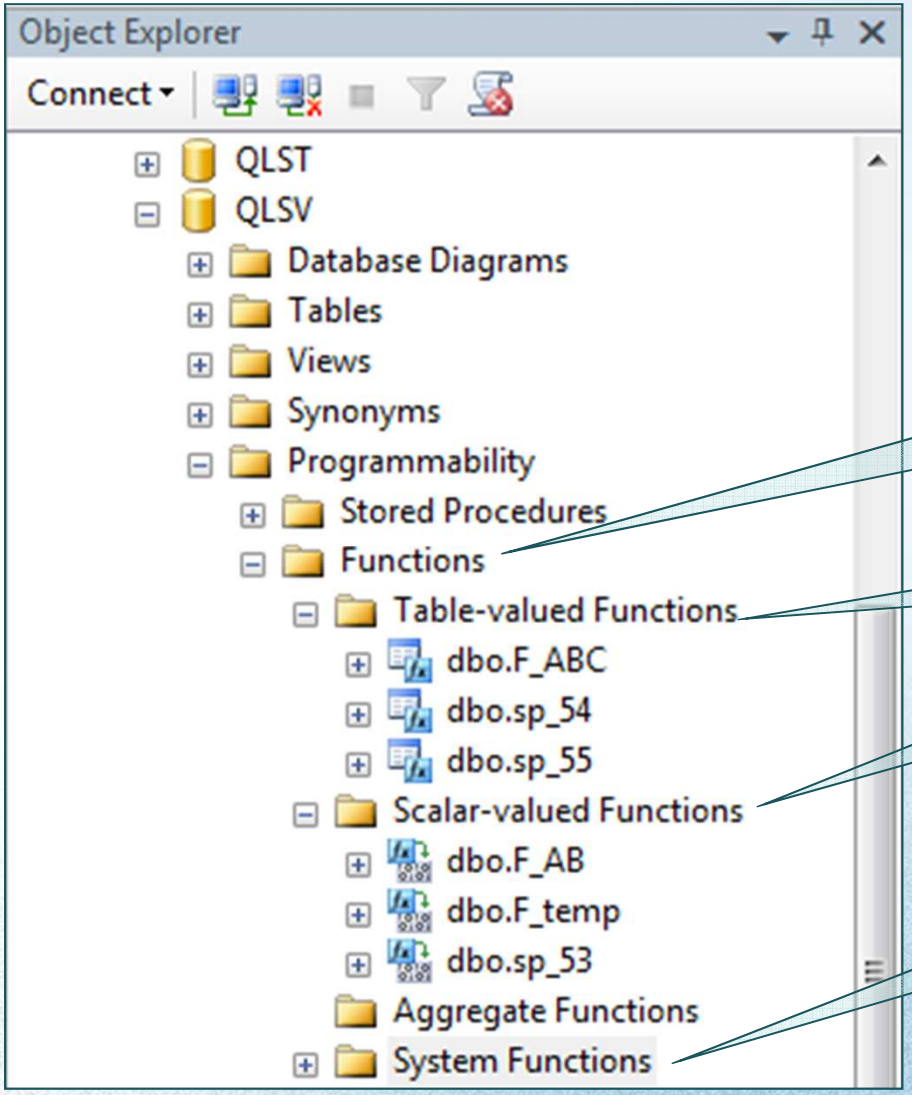
and Temp.Luong > 300’000

Hàm người dùng

- Các hàm người dùng được sử dụng trong câu truy vấn, trong biểu thức… phù hợp kiểu dữ liệu trả về của nó
- Ví dụ:
  - ***Select dbo.SoLonNhat(87,6,120)***
  - ***Select * from DanhSachMatHang('DH007')***

Hàm người dùng

- Ngoài các hàm do người dùng định nghĩa, SQL Server còn cung cấp các hàm xây dựng sẵn của hệ thống
- Các hàm này cung cấp tiện ích như xử lý chuỗi, xử lý thời gian, xử lý số học…
- Sinh viên tìm hiểu thêm về các hàm này trong Books on-line và các tài liệu tham khảo



Thư mục chứa hàm người dùng

Loại 2, 3

Loại 1

Thư mục chứa hàm hệ thống

Hết Chương III