CHƯƠNG 4: LẬP TRÌNH VỚI CSDL, PROCEDURE, FUNCTION

Giảng viên: Dương Quang Huy

NỘI DUNG

1. Khai báo và sử dụng “Biến”
2. Cấu trúc lệnh trong SQL Server
3. Các hàm cơ bản trong SQL Server
4. Sử dụng kiểu dữ liệu Cursor
5. Thủ tục (Stored Procedure)
6. Hàm (Function)

1. Khai báo và sử dụng “Biến”

❖ Biến cục bộ:

- ❑ Là biến do người lập trình khai báo, biến có thể được khai báo trong thủ tục nội tại, hàm.
- ❑ Cú pháp khai báo biến:

```
Declare @Tên_Biến <Kiểu_Dữ_Liệu>
```

- ❑ Tên Biến: Luôn bắt đầu bằng ký tự @, không có khoảng trắng, ký tự đặt biệt, ký tự số đứng đầu.
- ❑ Ví dụ: `Declare @SoLuong Int`

1. Khai báo và sử dụng “Biến”

- Gán giá trị cho biến:
  - Set @tên_biến=giátrị/hàm

  Hoặc
  - Select @tên_biến=giátrị/hàm

  Hoặc
  - Select @tên_Biến=Têncột/Hàm,...

    From ...

1. Khai báo và sử dụng “Biến”

- Ví dụ 1:

```
DECLARE @X int;
Set @X = 10;
Print N'X = '+Convert(Char(9),@X);
```

- Kết quả:

- Ví dụ 2:

```
DECLARE @NgayDH Datetime
SELECT @NgayDH = MAX(NGAYHD)
FROM HOADON
PRINT N'Ngày: '+Convert(Char(12),@NgayDH)
```

1. Khai báo và sử dụng "Biến"

❖ Biến hệ thống:

- ❑ Là biến do SQL Server cung cấp, cho biết trạng thái của hệ thống.
- ❑ Biến hệ thống luôn bắt đầu bằng hai ký tự @@ và chỉ đọc.
- ❑ Một số các biến thường dùng.

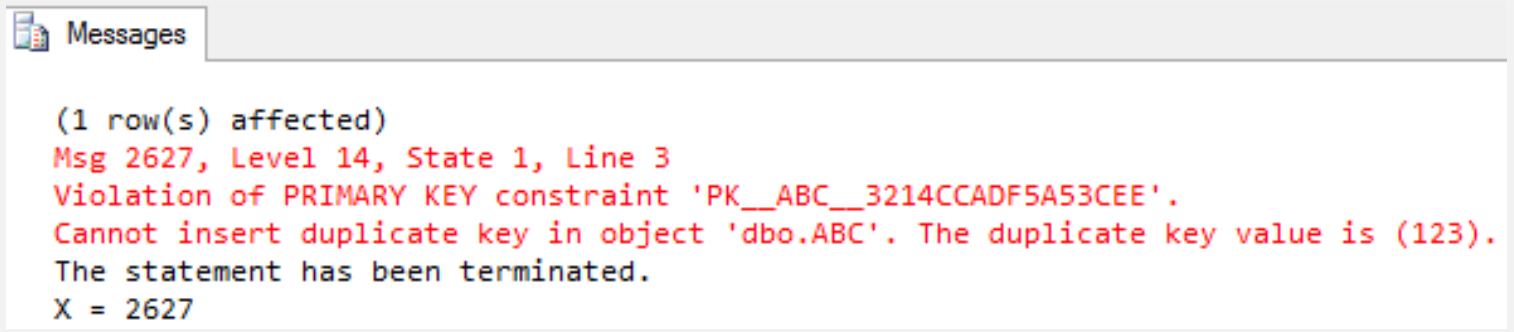
Tên Biến	Ý nghĩa
@@Error	Cho biết các lệnh trước đó có lỗi hay không, nếu có lỗi @@Error <> 0.
@@Rowcount	Cho biết số dòng bị tác động bởi câu truy vấn gần nhất.
@@Fetch_Status	Trả về <> 0 nếu lệnh duyệt CurSor gây lỗi.

1. Khai báo và sử dụng “Biến”

- Ví dụ:

```
Create Table ABC( MS int Primary key);
Declare @X int;
Insert into ABC Values(123);
Insert into ABC Values(123);
Set @X = @@ERROR;
Print N'X = '+Convert(Char(9),@X);
```

- Kết quả:



2. Cấu trúc lệnh trong SQLServer

❖ Lệnh điều khiển IF ... Else.

❑ Cú pháp:

```
If   <Điều kiện>
Begin
     <tập Lệnh 1>
End
Else
Begin
     <tập Lệnh 2>
End
```

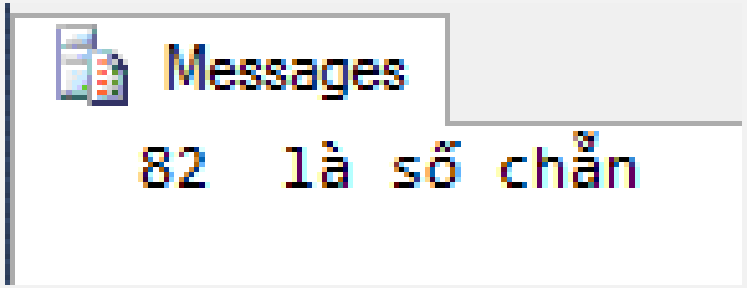
❑ Nếu tập hợp lệnh 1, tập lệnh 2 chỉ có 1 lệnh thì không cần Begin ... End.

2. Cấu trúc lệnh trong SQLServer

- Ví dụ: Kiểm tra số chẵn hay lẻ

```
Declare @x int
Set @x = RAND()*100
If @x % 2 = 0
    Print Convert(Char(3),@x)+N' là số chẵn'
Else
    Print Convert(Char(3),@x)+N' là số lẻ'
```

- Kết quả:



2. Cấu trúc lệnh trong SQLServer

❖ Lệnh điều khiển Case…When.

- ❑ Cho phép kiểm tra điều kiện và xuất thông tin theo từng trường hợp.
- ❑ Cú pháp:

```
Case <Biểu thức>
    When <giá trị> Then <biểu thức>
    When <giá trị> Then <biểu thức>
    ...
    [ELSE <biểu thức>]
End
```

2. Cấu trúc lệnh trong SQLServer

- Ví dụ:

```
Declare @So int, @kq nvarchar(10)
Set @So = RAND()*10;
Set @kq = Case @So
            When 0 Then N'Không'
            When 1 Then N'Một'
            When 2 Then N'Hai'
            When 3 Then N'Ba'
            When 4 Then N'Bốn'
            When 5 Then N'Năm'
            When 6 Then N'Sáu'
            When 7 Then N'Bảy'
            When 8 Then N'Tám'
            Else N'Chín'
          End;
Print @kq
```

2. Cấu trúc lệnh trong SQLServer

❖ Cấu trúc lặp While.

- ❑ Cú pháp:

```
While <Biểu thức Điều kiện>
Begin
    <Tập lệnh>
End
```

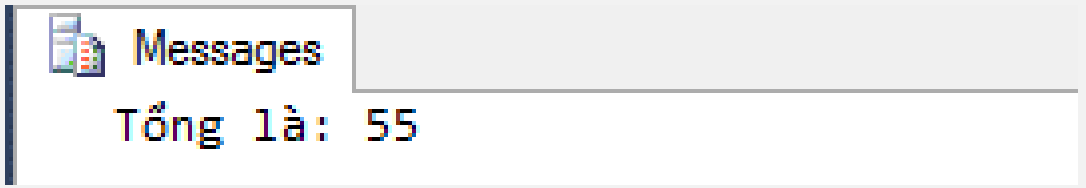
- ❑ Tập lệnh sẽ được thực hiện đến khi biểu thức điều kiện trả về False.
- ❑ Có thể dùng lệnh Break để thoát khỏi vòng lặp.

2. Cấu trúc lệnh trong SQLServer

- Ví dụ:

```
Declare @t int, @i int
Set @t = 0
Set @i = 1;
While (@i <= 10)
Begin
    Set @t = @t + @i
    Set @i = @i + 1
End
Print N'Tổng là: '+Convert(Char(3),@t)
```

- Kết quả:



3. Các hàm cơ bản trong SQL Server

- Thường dùng để chuyển dữ liệu từ số,ngày sang chuỗi.
- **Hàm Cast:** Chuyển một kiểu dữ liệu sang kiểu bất kỳ.

❑ Cú pháp:

```
Cast(Biểu_Thức as Kiểu_Dữ Liệu)
```

❑ Ví dụ:

```
Declare @t int
Set @t = 123 + 321
Print N'Tổng là: '+Cast(@t as char(9))
```

❑ Kết quả:

3. Các hàm cơ bản trong SQL Server

- **Hàm Convert:** Tương tự Cast, thường chuyển từ ngày sang chuỗi có định dạng.
  - Cú pháp:

```
Convert(KiểuDữLiệu,BiểuThức[,ĐịnhDạng])
```

  - Ví dụ 1:

```
Declare @t int
Set @t = 123 + 321
Print N'Tổng là: '+Convert(char(9),@t)
```

  - Kết quả:

3. Các hàm cơ bản trong SQL Server

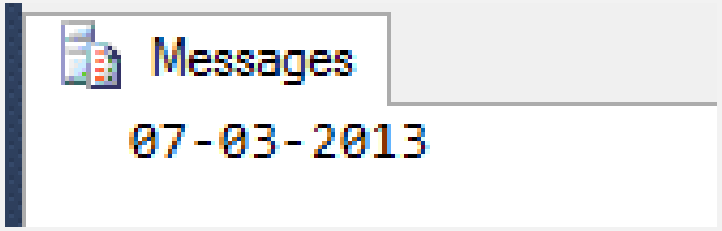
- Một số định dạng thường dùng:

Định dạng năm(YY)	Định dạng năm(YYYY)	Hiển thị dữ liệu
1	101	Mm/dd/yyyy
3	103	Dd/mm/yyyy
5	105	Dd-mm-yyyy
12	112	Yyyymmdd

- Ví dụ 2:

```
Print Convert(char(10),Getdate(),105)
```

- Kết quả:



3. Các hàm cơ bản trong SQL Server

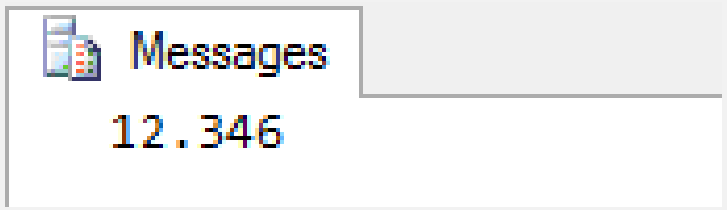
❖ **Hàm Str**: Chuyển số sang chuỗi có định dạng.

❑ Cú pháp:

```
Str(Số_Thực,Số_Ký_Tự[,Số_Lẻ])
```

❑ Ví dụ:

```
Declare @so float
Set @so = 12.3456
Print Str(@so,6,3)
```



❑ Chú ý: Khi nối chuỗi với số, ngày ta phải chuyển các giá trị này sang chuỗi, sau đó sử dụng dấu “+” để nối.

3. Các hàm cơ bản trong SQL Server

❖ Các hàm trên kiểu chuỗi.

❑ Hàm ASCII(string)

- ➢ Hàm trả về mã ASCII của ký tự đầu tiên bên trái của chuỗi.
- ➢ Select Ascii('Abc') -> Kết quả: 65

❑ Hàm CHAR(ascii_code)

- ➢ Hàm trả về ký tự có mã ASCII tương ứng.
- ➢ Select Char(65) -> Kết quả: A

3. Các hàm cơ bản trong SQL Server

❖ Các hàm trên kiểu chuỗi.

- ❑ Hàm LEFT(string,n)
  - ➢ Hàm trích ra n ký tự bên trái của string.
  - ➢ Select Left('ABCD',2) -> Kết quả: AB
- ❑ Hàm RIGHT(string,n)
  - ➢ Hàm trích ra n ký tự bên phải của string
  - ➢ Select RIGHT('ABCD',2) -> Kết quả: CD

3. Các hàm cơ bản trong SQL Server

❖ Các hàm trên kiểu chuỗi.

- ❑ Hàm LEN(string)
  - ➢ Hàm trả về độ dài của chuỗi string.
- ❑ Hàm LOWER(string)
  - ➢ Chuyển chuỗi string thành chữ thường.
- ❑ Hàm UPPER(string)
  - ➢ Chuyển chuỗi string thành chữ hoa.
- ❑ Hàm LTRIM(string)
  - ➢ Bỏ các khoảng trắng thừa bên trái.
- ❑ Hàm RTRIM(string)
  - ➢ Bỏ các khoảng trắng thừa bên phải.

3. Các hàm cơ bản trong SQL Server

❖ Các hàm trên kiểu chuỗi.

- ❑ Hàm REVERSE(string)
  - ➢ Hàm trả về chuỗi đảo ngược của string.
- ❑ Hàm REPLACE(string1,string2,string3)
  - ➢ Trả về một chuỗi có được bằng cách thay thế các chuỗi string2 trong chuỗi string1 bởi chuỗi string3.
- ❑ Hàm SUBSTRING(string, m, n)
  - ➢ Trích ra từ n ký tự từ string bắt đầu từ ký tự thứ m.

3. Các hàm cơ bản trong SQL Server

❖ Các hàm ngày giờ.

Bảng mô tả các định dạng trong các hàm thời gian

Giá trị	Định dạng
Năm	yy, yyyy
Quý	qq, q
Tháng	mm, m
Ngày trong năm	dy, y
Ngày trong tuần	dw
Ngày trong tháng	dd, d
Tuần	wk, ww
Giờ	hh
phút	mi, n
giây	ss, s

❖ Các hàm ngày giờ.

- ❑ Hàm Getdate() ->Trả về ngày hiện hành.
- ❑ Hàm Day(),Month(),Year()

  -> Trả về ngày, tháng, năm.
- ❑ Hàm Datename(Định dạng, Ngày)
  - ➢ Trả về chuỗi thời gian.
  - ➢ Print datename(dw,getdate()) ->Thursday
- ❑ Hàm Datepart(Định dạng, Ngày)
  - ➢ Trả về một giá trị trong của ngày.
  - ➢ Ví dụ: Cho biết QUÝ của ngày hiện tại.

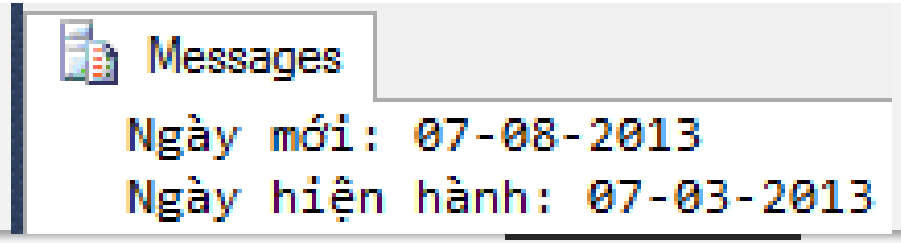
  Print datepart(qq,getdate()) -> 2

3. Các hàm cơ bản trong SQL Server

- Hàm DateAdd(Định dạng,Số,Ngày)
  - dùng cộng một số vào giá trị ngày và trả về một giá trị ngày mới.
  - Ví dụ: Cộng thêm 5 tháng.

```
Declare @ngaymoi Datetime
Set @ngaymoi=DateAdd(mm,5,Getdate())
Print N'Ngày mới: '
      + Convert(Char(10),@ngaymoi,105)
Print N'Ngày hiện hành: '
      + Convert(Char(10),Getdate(),105)
```

- Kết quả:



3. Các hàm cơ bản trong SQL Server

- Hàm Datediff(định dạng,ngày_1,ngày_2)
  - Trả về khoảng cách của hai ngày.
  - Ví dụ:

```
Print Datediff(mm,Getdate(),@ngaymoi)
```

  - Kết quả: 5
- Hàm Round(số,số chữ số thập phân)
  - Hàm làm tròn số.
  - Ví dụ:

```
Print Cast(Round(12.3456,2) as Char(10))
```

  - Kết quả:

4. Kiểu dữ liệu con trỏ(Cursor)

❖ Giới thiệu:

- ❑ Khi cần xử lý vấn đề trên một mẩu tin hoặc trên nhiều mẩu tin, cùng thời gian với hình thức tính toán khác nhau. ta sử dụng kiểu dữ liệu Cursor.
- ❑ Cursor là đối tượng dùng để chứa dữ liệu lấy từ CSDL (câu truy vấn).

4. Kiểu dữ liệu con trỏ(Cursor)

❖ **Cú pháp khai báo biến Cursor:**

Declare Tên_Biến Cursor

[phạm vi] [di chuyển][trạng thái][xử lý]

For <Câu lệnh Select>

[For Update [OF danh sách cột]]

4. Kiểu dữ liệu con trỏ(Cursor)

Trong đó:

- [Phạm vi]:
  - Local: Chỉ sử dụng trong phạm vi khai báo(mặc định).
  - Global: Sử dụng chung cho cả kết nối.

- [Di chuyển]:
  - Forward_Only: Chỉ di chuyển một hướng từ trước ra sau(mặc định).
  - Scroll: Di chuyển tùy ý.

- Trạng thái
  - Static: Dữ liệu trên Cursor không thay đổi mặt dù dữ liệu trong bảng nguồn thay đổi(mặc định)
  - Dynamic: Dữ liệu trên Cursor sẽ được thay đổi khi dữ liệu trong bảng nguồn thay đổi (Cursor chế động).
  - KeySet :Giống Dynamic nhưng chỉ thay đổi những dòng bị cập nhật.
- Xử lý:
  - Read_Only : Chỉ đọc(mặc định)
  - Scroll_Lock: Đọc/ghi.

4. Kiểu dữ liệu con trỏ(Cursor)

- Câu lệnh Select: Chỉ đến các cột bên trong bảng mà chúng ta cần đọc dữ liệu.

- [Danh sách cột]: Chỉ định danh sách tên các cột sẽ được phép thay đổi giá trị trong Cursor.

- Ví dụ 1: Tạo biến cursor chứa dữ liệu trong bảng MatHang, các dòng dữ liệu trong cursor cho phép được cập nhật.

```
Declare Cur_MatHang Cursor
Dynamic
For Select *From MatHang
```

- Ví dụ 2: Tạo cursor chứa thông tin KhachHang, dữ liệu trong cursor chỉ được phép đọc và chỉ đọc theo một chiều đi tới.

```
Declare Cur_KhachHang Cursor
Forward_only
Static
Read_only
For Select * From KhachHang
```

4. Kiểu dữ liệu con trỏ(Cursor)

❖ Đọc và xử lý dữ liệu trong Cursor.

❑ Cú pháp:

FETCH <Hướng di chuyển>
From <Tên_biến> Into <Danh sách biến>

❑ <Hướng di chuyển>:

- ➢ Next: Di chuyển về sau
- ➢ Prior: Di chuyển về trước
- ➢ First: Di chuyển về đầu
- ➢ Last: Di chuyển về cuối

4. Kiểu dữ liệu con trỏ(Cursor)

- <Hướng di chuyển>:
  - ABSOLUTE n: Di chuyển đến mẩu tin thứ n tính từ mẩu tin đầu tiên.
  - RELATIVE n: Di chuyển đến mẩu tin thứ n tính từ mẩu tin hiện hành.

- Trong quá trình di chuyển để kiểm tra việc di chuyển có thành công hay không. Ta kiểm tra biến hệ thống: @@Fetch_Status nếu <>0 là thất bại.

4. Kiểu dữ liệu con trỏ(Cursor)

❖ **Đóng và giải phóng Cursor:**

- ❑ Close Tên_Biến
- ❑ Deallocate Tên_Biến

❖ **Chú ý:** Thứ tự các thao tác trên Cursor.

1. Định nghĩa biến Cursor
2. Mở Cursor
3. Duyệt và xử lý dữ liệu trên Cursor
4. Đóng và giải phóng Cursor

4. Kiểu dữ liệu con trỏ(Cursor)

❑ **Ví dụ:** Đọc thông tin từng nhân viên

```
Declare Cur_nv Cursor
For Select MSNV,TENNV From NhanVien
Open Cur_nv
Declare @ms char(6),@ten nvarchar(50)
Fetch next From Cur_nv Into @ms,@ten
While @@fetch_status = 0
Begin
    Print 'MSNV: '+@ms+N' Tên: '+@ten
    Fetch next from Cur_nv into @ms,@ten
End
Close Cur_nv
Deallocate Cur_nv
```

5. Thủ tục (Stored Procedure)

❖ **Giới thiệu:**

- ❑ Stored Procedure (SP) là “Chương trình con” của Sql Server.
- ❑ Có tham số vào, tham số ra và có thể trả về kết quả.
- ❑ Có thể gọi thực thi trong SQL hay trong các ứng dụng khác.
- ❑ Xử lý xây dựng trong SP sẽ chạy nhanh hơn khi xây dựng ngoài ứng dụng.
- ❑ Theo mô hình lập trình client-server tất cả các xử lý điều tập trung tại server.

5. Thủ tục (Stored Procedure)

❖ **Cú pháp tạo thủ tục:**

```
Create Proc Tên_Thủ_Tục[(Các tham số)]
As
 <Các câu lệnh>
```

❖ **Trong đó:** <Các tham số> có hai loại:

❑ Tham số vào: Nhận giá trị từ người dùng truyền vào cho SP xử lý.

➢ Cú pháp:

@TênBiến Kiểu_DữLiệu[=giá trị mặc định]

5. Thủ tục (Stored Procedure)

- Tham số ra: Nhận kết quả trả về từ SP và hiển thị cho người dung.
- Cú pháp:

  @TênBiến Kiểu_DữLiệu OutPut

Gọi thực hiện SP:

```
Exec Tên_Thủ_Tục[@ten_Tham_số_vào=gia_trị[,..]]
      [@ten_Tham_số_ra=@ten_Bien output],[,...]
```

- Trong SP có thể dùng lệnh Return để trả về kết quả hoặc để thoát khỏi SP.

5. Thủ tục (Stored Procedure)

- **Ví dụ:** Cho lược đồ CSDL như sau:

NhanVien(msnv, tennv, ngaysinh, phai,
diachi, dienthoai)

KhachHang(mskh, tenkh, phai, diachi, dienthoai)

MatHang(msmh, tenmh, sl_ton, dongia, donvitinh)

HoaDon(mshd, msnv, mskh, ngayhd, tongtien)

ChiTiet_HD(mshd,msmh, soluong, thanhtien).

5. Thủ tục (Stored Procedure)

- Ví dụ 1: Tạo sp với tham số vào mskh. Hãy cho biết thông tin các đơn hàng do khách hàng trên đã đặt mua.

```
Create Proc p_TT_HoaDon(@mskh int)
As
    Select *From HoaDon
    Where MSKH = @mskh
```

- Thực thi SP:

```
Exec p_TT_HoaDon @mskh = 11
```

5. Thủ tục (Stored Procedure)

- Ví dụ 2: Tạo sp với tham số vào mskh. Đầu ra: Hãy cho biết tổng tiền của khách hàng trên đã mua.

```
Create Proc p_TongTien_KH(
    @mskh int,
    @tt int OutPut
)As
    Select @tt = Sum(TongTien)
    From HoaDon
    Where MSKH = @mskh
```

- Thực thi SP:

```
Declare @Tong int
Exec p_TongTien_KH @mskh=2,@tt=@Tong OutPut
Print N'Tổng tiền: '+Cast(@Tong as char(10))
```

5. Thủ tục (Stored Procedure)

- Ví dụ 3: Tạo sp với tham số vào msmh. Đầu ra: Hãy cho biết đơn giá của mặt hàng.

```
Create Proc p_DonGia(@msmh char(6))
As
    Declare @g money
    Select @g = DonGia
    From MatHang
    Where Msmh = @msmh
    Return @g
```

- Thực thi SP:

```
Declare @dg money
Exec @dg = p_DonGia @msmh='C0002'
Print N'Đơn giá: '+Cast(@dg as char(10))
```

5. Các dạng Stored Procedure

5. Giao tác(Transaction):

- Là tập hợp các lệnh sẽ được thực hiện nếu tất cả đều thành công, nếu có một lệnh thất bại, thì sẽ không có lệnh nào được thực hiện.
- Ví dụ: Giao tác chuyển tiền của ngân hàng. Chuyển số lượng **N** từ tài khoản A sang tài khoản B, các công việc được thực hiện:
  - TaiKhoanA=TaiKhoanA - **N**
  - TaiKhoanB=TaiKhoanB + **N**
- Hai công việc này sẽ được thực hiện nếu không có lệnh nào gây lỗi.

5. Các dạng Stored Procedure

- Cú pháp xây dựng 1 giao tác trong SQL Server:
  - Lệnh Bắt đầu 1 Giao tác :
    Begin Tran
  - Lệnh kết thúc thành công 1 giao tác:
    Commit Tran
  - Lệnh kết thúc thất bại 1 giao tác:
    Rollback Tran
- Để kiểm tra các lệnh thực hiện có thành công hay không: Truy cập đến giá trị của biến @@Error.
  - Nếu @@Error<>0 là thất bại.

5. Các dạng Stored Procedure

- Cú pháp:

```
Begin Tran
    <tập các lệnh>
If @@error<>0
Begin
    print N'Giao tác thất bại'
    Rollback Tran
End
Else
    Commit Tran
```

- Ví dụ: Xử lý hóa đơn.

```
Begin tran HD
    declare @err1 int, @err2 int, @err3 int
    insert into khachhang(mskh,tenkh,phai,diachi,dienthoai)
        values(4,N'Nguyễn Ngọc Anh',N'Nữ',N'HCM','')
    set @err1 = @@error
    insert into hoadon(mshd,msnv,mskh,ngayhd)
        values(3,'NV002',4,GETDATE())
    set @err2 = @@error
    insert into chitiet_hd(mshd,msmh,soluong) values(3,'M0001',99)
    insert into chitiet_hd(mshd,msmh,soluong) values(3,'C0002',1)
    set @err3 = @@error
if @err1 <>0 or @err2 <>0 or @err3 <>0
begin
    raiserror (N'Lỗi',16,1)
    rollback tran HD
end
else
    commit tran HD
```

ocedure

```
--Ví dụ: Cách bắt lỗi trên MS SQL Server
Create table ABC( ms int primary key)
go
Begin tran HD
    begin try
        insert into abc (ms) values (1)
        insert into abc (ms) values (2)
        insert into abc (ms) values (3)
        insert into abc (ms) values (1)
    end try
    begin catch
        if ERROR_NUMBER()<>0
        begin
            print ERROR_NUMBER()
            rollback tran HD
        end
    end catch
    if ERROR_NUMBER() = 0
        commit tran HD
```

❑ Các hàm đọc thông tin lỗi

STT	TÊN HÀM	CHỨC NĂNG
1	ERROR_NUMBER()	Trả lại mã lỗi (dưới dạng số)
2	ERROR_SEVERITY()	Trả lại mức độ nghiêm trọng của lỗi
3	ERROR_STATE()	Trả lại trạng thái của lỗi (dưới dạng số)
4	ERROR_PROCEDURE()	Trả lại tên của Stored Procedure hoặc tên của Trigger đã phát sinh lỗi
5	ERROR_LINE()	Trả lại vị trí dòng lệnh đã phát sinh ra lỗi.
6	ERROR_MESSAGE()	Trả lại thông báo lỗi dưới hình thức văn bản (text)

❑ Các hàm đọc thông tin lỗi

```
BEGIN TRANSACTION;
BEGIN TRY -- Generate a constraint violation error.
        DELETE FROM Production.Product
        WHERE ProductID = 980;
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;
        IF @@TRANCOUNT > 0
        ROLLBACK TRANSACTION;
END CATCH;
IF @@TRANCOUNT > 0
        COMMIT TRANSACTION;
```

```
BEGIN TRANSACTION;
BEGIN TRY -- Generate a constraint violation error.
        DELETE FROM Production.Product
        WHERE ProductID = 980;
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;
        IF @@TRANCOUNT > 0
        ROLLBACK TRANSACTION;
END CATCH;
IF @@TRANCOUNT > 0
        COMMIT TRANSACTION;
```

6. Hàm (Function)

1. Khái Niệm.

- Là đối tượng được bổ sung vào từ SQL Server 2000.
- Mang đầy đủ tính chất của một hàm. Có thể có tham số vào, xử lý và trả về kết quả.
- Các loại hàm:
  - Hàm xác định(deter-ministic): Luôn trả về 1 giá trị khi nhận các giá trị truyền vào như nhau.
  - Hàm không xác định(non-deterministic): Cho giá trị khác nhau tùy thời gọi như hàm Getdate().

6. Hàm (Function)

2. Xây dựng hàm.

❖ **Hàm trả về một giá trị:** giá trị trả về có kiểu dữ liệu là một trong các kiểu của SQL Server.

❑ Cú pháp:

```
Create Function Tên_Hàm[(Các tham số)]
Returns Kiểu_dữ_liệu_trả_về
As
Begin
    <Các lệnh xử lý>
    Return kết_Quả
End
```

6. Hàm (Function)

- Ví dụ: Tính tổng 2 số nguyên.

```
Create Function f_TinhTong(@a int, @b int)
Returns int
As
Begin
    Declare @t int
    Set @t= @a + @b
    return @t
End
```

- Ta có thể gọi các hàm loại này trong câu truy vấn, trong lệnh tạo bảng, trong thủ tục nội tại …

6. Hàm (Function)

- Sử dụng hàm:
  - Ví dụ:

```
Declare @tong int
set @tong = dbo.f_TinhTong(15,10)
Print N'Tổng='+Cast(@tong as char(10))
```

- Xóa hàm: Drop Function <Tên_Hàm>
  - Ví dụ:

```
Drop Function f_TinhTong
```

6. Hàm (Function)

- **Hàm trả về dữ liệu được lấy từ các bảng trong CSDL:** Giống như View nhưng có tham số vào.
  - Cú pháp:

```
Create Function Tên_Hàm[(các tham số)]
Returns Table
As
    Return(Câu lệnh Select)
```

  - Gọi hàm loại này giống như View.

6. Hàm (Function)

- Ví dụ: Viết hàm tìm mặt hàng theo tên.

```
Create Function f_Tim_MH
(
    @TenMH nvarchar(100)
) Returns Table
As
    Return( Select *
            from MatHang
            Where TENMH Like @TenMH
          )
```

- Sử dụng hàm: `Select *from f_Tim_MH(N'CP%')`

- Kết quả:

Results | Messages

	MSMH	TENMH	SL_TON	DONGIA	DONVITINH
1	C0001	CPU i5 4x2.5 GHz	49	2000000	Chiếc
2	C0002	CPU i3 4x2.0 GHz	49	1500000	Chiếc
3	C0003	CPU i7 8x1.8 GHz	49	2000000	Chiếc

6. Hàm (Function)

- **Hàm tạo bảng**: Tạo và trả về một bảng, trong hàm có thể chứa tất cả các lệnh T-SQL.
  - Cú pháp:

```
Create Function Tên_Hàm[(các tham số)]
Returns @TênBảng_trả_về  Table
(
    Tên_cột Kiểu_dữ_liệu[,…]
)As
Begin
    <Các lệnh T-SQL>
    Insert into @TênBảng_trả_về ...
    Return
End
```

- Ví dụ: Viết hàm thống kê doanh thu theo từng khách hàng trong ngày.

```
Create Function f_ThongKe (@ngay Datetime)
Returns @tk Table
(    Mskh int Primary Key,
     Tenkh nvarchar(50)  ,
     DThoai varchar(15)  ,
     TongTien_DaMua money
)As
Begin
    Insert @tk
    Select kh.Mskh, Tenkh, DienThoai,Sum(TongTien)
    From KhachHang kh, HoaDon hd
    Where  kh.Mskh = hd.Mskh
           And DATEDIFF(dd,@ngay, NgayHD) = 0
    Group By  kh.Mskh, Tenkh, DienThoai
    Return
End
```

- Sử dụng hàm:

```
Select *From f_ThongKe('03/27/2004')
```

- Kết quả:

Results | Messages

	Mskh	Tenkh	DThoai	TongTien_DaMua
1	3	Phạm Ngọc Lan	0933124456	2150000.00