

Hệ quản trị CSDL

Thủ tục nội - Stored Procedure



© 2023.08 – Đào tạo từ xa – ĐH. Khoa học tự nhiên TpHCM

Lương Vĩ Minh – lvminh@fit.hcmus.edu.vn

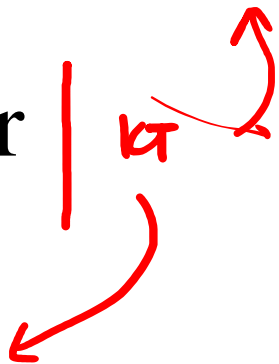
Nội dung

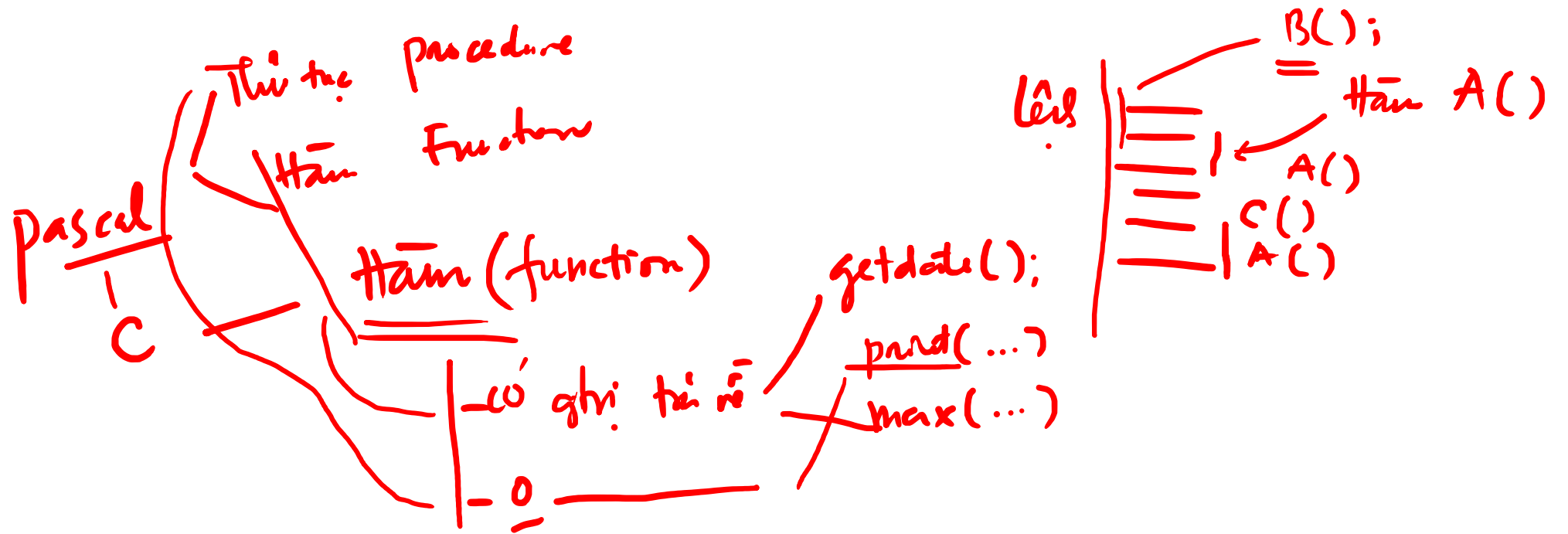
>50%

■ Thủ tục nội – Stored Procedure

■ Con trỏ dữ liệu – Cursor

10% ■ Hàm - Function





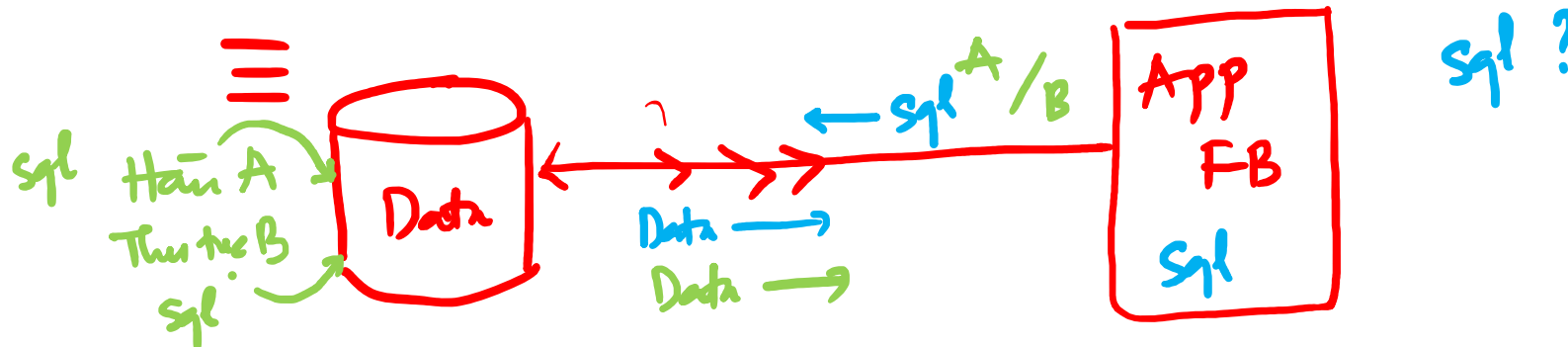
Thủ tục nội - Stored Procedure

Thủ tục nội - 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:

1. Chúng ta có thể lưu trữ các chương trình cục bộ và tạo các ứng dụng để gọi các lệnh đến SQL Server và xử lý các kết quả
2. 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.



Thủ tục nội - Giới thiệu - Đặc tính

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

- Chấp nhận 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.

Thủ tục nội - Giới thiệu - Phân loại

Ta có thể dùng lệnh **EXECUTE** 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:

1. *Stored procedure cho phép điều chỉnh chương trình cho phù hợp.*
2. *Stored procedure cho phép thực thi nhanh hơn. ✓*
3. *Stored procedure có thể làm giảm bớt vấn đề kẹt đường truyền mạng. ✓*
4. *Stored procedure có thể sử dụng trong vấn đề bảo mật của máy.*

Thủ tục nội - Giới thiệu

- **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.

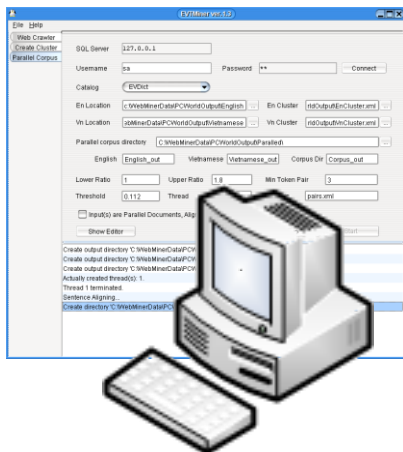
Thủ tục nội - Giới thiệu

- **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 gửi 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.

Truy vấn dữ liệu thông thường

1 

2 



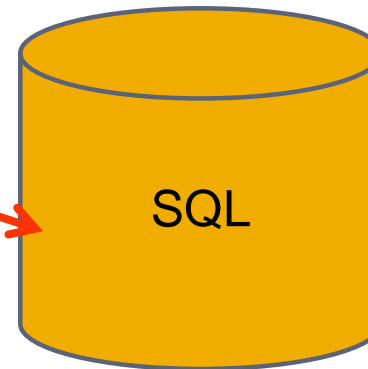
Select * from Sinhvien

login

un
pw

Select
from
where

u = 'un' and
p = 'pw'



SQL1
SQL2
SQL3
SQL5

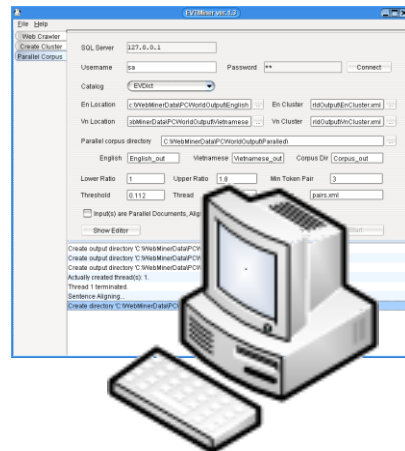
✓ 1. Kiểm tra cú pháp

✓ 2. Kiểm tra nghĩa

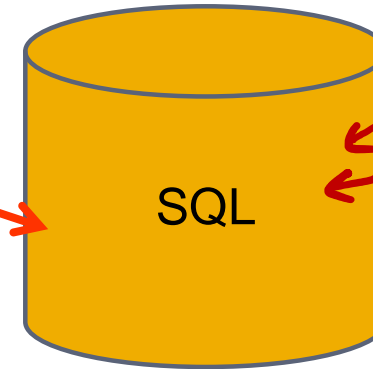
Results		Messages			
	isbn	ma_tu_vch	ngonngu	bia	trangthai
1	1	1	ANH	BIA CUNG	Y
2	3	1	DUC	BIA CUNG	Y
3	4	1	HAN	BIA CUNG	Y
4	5	1	HOA	BIA CUNG	Y
5	6	1	NHAT	BIA CUNG	Y
6	7	1	PHAP	BIA CUNG	Y
7	9	1	VIET	BIA CUNG	Y
8	10	1	Y	BIA CUNG	Y
9	11	2	ANH	BIA CUNG	Y
10	13	2	DUC	BIA CUNG	Y
11	14	2	HAN	BIA CUNG	Y

hoạch thực hiện
y vấn
truy vấn

Truy vấn dữ liệu với stored-proc



Tên - Stored-
Procedure



A()
B()

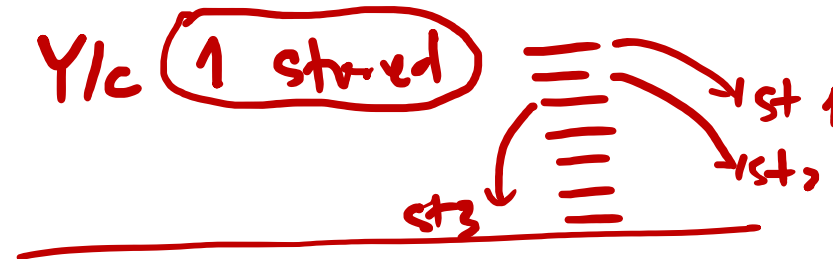
1. Kiểm tra cú pháp

2. Kiểm tra nghĩa

Results					
Messages					
	isbn	ma_tuasach	ngonngu	bia	trangthai
1	1	1	ANH	BIA CUNG	Y
2	3	1	DUC	BIA CUNG	Y
3	4	1	HAN	BIA CUNG	Y
4	5	1	HOA	BIA CUNG	Y
5	6	1	NHAT	BIA CUNG	Y
6	7	1	PHAP	BIA CUNG	Y
7	9	1	VIET	BIA CUNG	Y
8	10	1	Y	BIA CUNG	Y
9	11	2	ANH	BIA CUNG	Y
10	13	2	DUC	BIA CUNG	Y
11	14	2	HAN	BIA CUNG	Y

hoạch thực hiện
y vấn
truy vấn

Thủ tục nội - Khai báo



- 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

Create P-A

A_2

E

$\left(\begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \end{array} \right)$

```
CREATE PROCEDURE [procedure_name]
    {@parameter data type input/output} /*các biến tham số vào ra*/
AS
Begin
    [khai báo các biến cho xử lý]
    {Các câu lệnh transact-sql}
End
```

Ghi chú:

- Trong SQL Server, có thể ghi tắt một số từ khóa mà tên có chiều dài hơn 4 ký tự. Ví dụ: có thể thay thế Create Procedure bằng Create Proc.
- Tên hàm, tên biến trong SQL Server không phân biệt hoa thường.

Khai báo biến và gán giá trị cho biến, Ghi chú

```
/*Khai báo biến*/  
DECLARE @parameter_name data_type;  
/*Gán giá trị cho biến*/  
SET @parameter_name = value  
SELECT @parameter_name = value  
/*In thông báo ra màn hình*/  
print N'Chuỗi thông báo unicode'  
--Ghi chú 1, một dòng  
/*  
    Ghi chú 2  
    Nhiều dòng  
*/
```

cast(km, int)

```
Declare @tongsv int  
Select @tongsv = count(masv)  
From Sinhvien
```

Biên dịch và gọi thực thi một stored-procedure

- Biên dịch:
 - Chọn (tô chọn) toàn bộ mã lệnh Tạo stored-procedure → Nhấn **F5**
- Gọi thực thi một store-Procedure đã được biên dịch bằng lệnh **exec** theo cú pháp sau:

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

Lệnh cập nhật Procedure

```
ALTER PROCEDURE  procedure_name  
    [ { @parameter data_type } ]  
AS  
Begin  
    [khai báo các biến cho xử lý]  
    {Các câu lệnh transact-sql}  
End
```

Lệnh xóa Procedure

```
DROP PROCEDURE procedure_name
```


Một số kỹ thuật

@cnt = 12

C# (@cnt == 12)

*✓ 'and'
or*

■ Vòng lặp

■ While (điều kiện)

Begin
.....
End

creat proc KTS (@Hbint)

*AS if (exists (select 1 from tv
where MSN = @Masb))*

■ Điều kiện

■ If (điều kiện)

Begin End

Else

Begin End

■ Hàm SQL và các biến đặc biệt

■ Exists (câu truy vấn Select)

Thủ tục nội - Ví dụ 1

- Tạo stored-procedure tính tổng của 2 số nguyên

```
--Tạo stored-procedure sp_tong  
CREATE PROCEDURE sp_Tong  
    @So1 int, @So2 int, @Tong int out  
AS  
Begin  
    SET @Tong = @So1 + @So2;  
End
```

--Biên dịch stored-procedure → F5

```
--Kiểm tra  
Declare @Sum int;  
Exec sp_Tong 1, 2, out @Sum;  
Select @Sum
```

Thủ tục nội - Ví dụ 2

- Tạo stored procedure liệt kê những thông tin của đầu sách, thông tin tựa sách và số lượng sách hiện chưa được mượn của một đầu sách cụ thể (ISBN). (Sử dụng CSDL Quản lý thư viện)

Sql

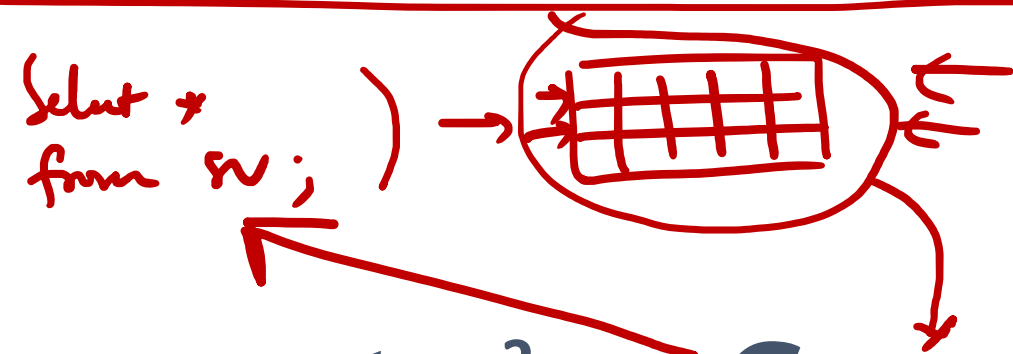
```
CREATE PROCEDURE sp_ThongtinDausach
    @isbn int
AS
Begin
    SELECT tuasach, tacgia, ngonngu, bia, trangthai, count(*)
    FROM dausach ds, tuasach ts, cuonsach cs ← Join
    WHERE
        ds.ma_tuasach = ts.ma_tuasach AND
        ds.isbn = cs.isbn AND
        ds.isbn = @isbn AND
        tinhtrang = yes
    GROUP BY tuasach, tacgia, ngonngu, bia, trangthai
End
```

~~let~~ Declare Cx int ;
 | select Cx=cont(*)
 | from sv ;

1 ghi: đtr

Ⓢ x int
 8B
 \$x - 16

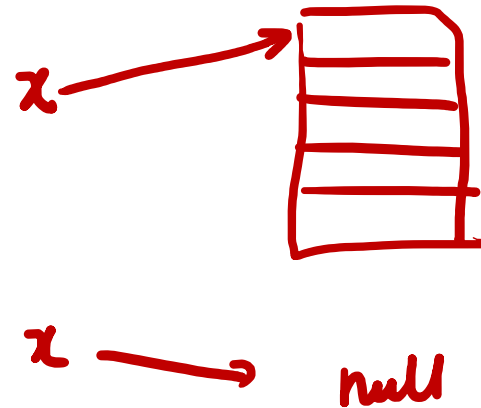
cursor x → 100
 ABC
 [Adv] →



Kiểu dữ liệu con trỏ - Cursor

Giới thiệu Cursor

- Con trỏ - Cursor trong CSDL là một kiểu dữ liệu đặc biệt được sử dụng để lưu trữ kết quả của MỘT câu lệnh SELECT trong quá trình lập trình CSDL.
- Cú pháp lệnh:
 1. Khai báo
 2. Mở cursor
 3. Kiểm tra Cursor
 4. Truy cập dữ liệu trong Cursor
 5. Đóng Cursor



Cursor - Khai báo & Mở

■ Lệnh khai báo biến Cursor

a) `DECLARE @cursor_name [INSENSITIVE] [SCROLL] CURSOR FOR Select_statement;`

b) `DECLARE @cursor_name CURSOR;
Set @cursor_name = CURSOR FOR Select_statement;`

■ Lệnh mở Cursor

→ `OPEN @cursor_name;`

Cursor - Kiểm tra & Lấy dữ liệu

- Kiểm tra trạng thái kết quả của Cursor:

@@FETCH_STATUS = 0 : lấy dữ liệu thành công.
@@FETCH_STATUS < 0 : không lấy được dữ liệu.

*select A, B, C
from Table1,*

(luôn kiểm tra trạng thái Cursor sau mỗi lần đọc dữ liệu Fetch)

- Lấy dữ liệu từ trong Cursor:

`FETCH NEXT FROM @cursor_name INTO @variable1, @variable2, ... ;`

Cursor - Đóng

- Đóng cursor sau khi đã duyệt và sử dụng xong

```
CLOSE @cursor_name;  
DEALLOCATE @cursor_name;
```


Cursor - Ví dụ

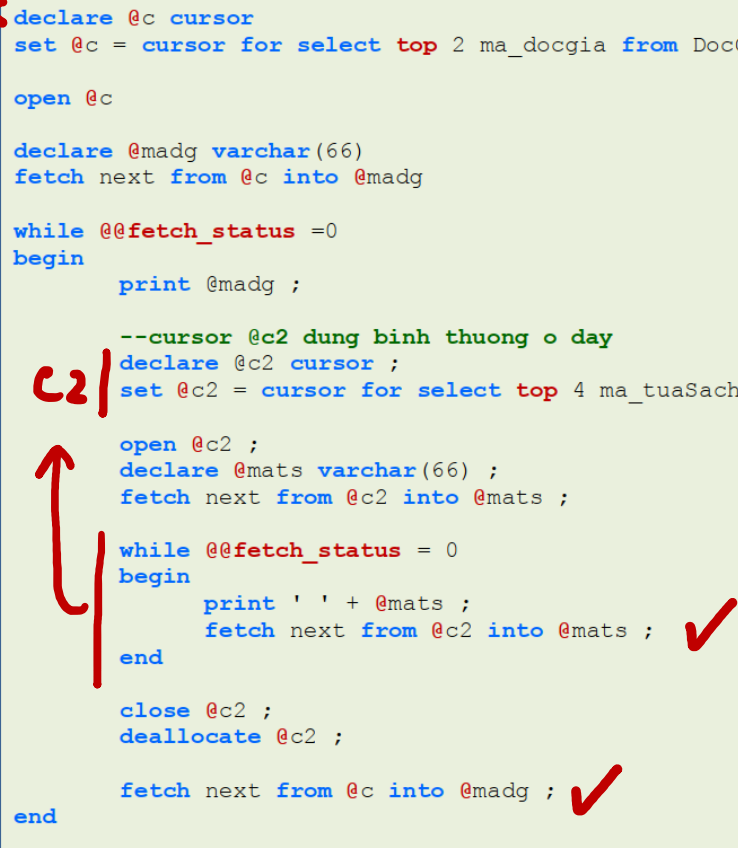
```
--Khai báo biến
Declare @btt int, @btt2 int ; ✓
Declare @c cursor
Set @c = cursor for Select tt,tt2 From bang1 Where [điều kiện] ;

--Mở cursor open @c ;
Fetch next from @c into @btt, @btt2 ; → dữ 1

--Duyệt cursor
|| while @@fetch_status = 0
Begin
    --Sử dụng 2 biến @btt, @btt2. Sau đó, gọi tiếp ✓
    fetch next from @c into @btt, @btt2, ... ; ✓
End
→ dữ kế tiếp

--Đóng cursor
|| Close @c ;
Deallocate @c ;
```

Cursor - Ví dụ



```
declare @c cursor
set @c = cursor for select top 2 ma_docgia from DocGia

open @c

declare @madg varchar(66)
fetch next from @c into @madg

while @@fetch_status = 0
begin
    print @madg ;

    --cursor @c2 dung binh thuong o day
    declare @c2 cursor ;
    set @c2 = cursor for select top 4 ma_tuaSach from TuaSach ;

    open @c2 ;
    declare @mats varchar(66) ;
    fetch next from @c2 into @mats ;

    while @@fetch_status = 0
    begin
        print ' ' + @mats ;
        fetch next from @c2 into @mats ;
    end

    close @c2 ;
    deallocate @c2 ;

    fetch next from @c into @madg ;

end

close @c
deallocate @c
```

Hàm trong SQL

Hàm trong SQL

- 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

Hàm - Cú pháp

udf

getdate();

■ Khai báo hàm:

```
CREATE FUNCTION function_name  
    ( [ @parameter_name parameter_data_type ] )  
    RETURNS [return Data-type] /*từ khoá Returns có 's' */  
AS Begin  
    .....  
    return [scalar value/select command]  
End
```

■ Gọi thực thi hàm:

Dbp.function_name (.....);

■ Chỉnh sửa hàm:

Alter function function_name (.....)

■ Xoá hàm:

Drop function function_name;

Hàm - Ví dụ

```
--Xóa hàm nếu đã có
if object_id('fTuoi','FN') is not null
    drop function fTuoi
go

--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.
```

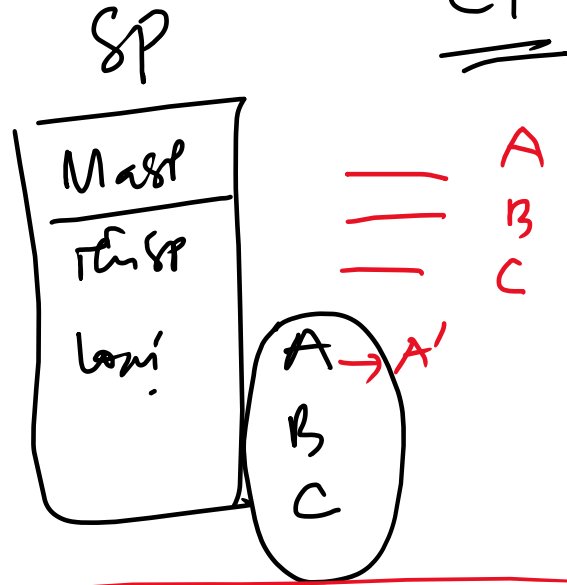
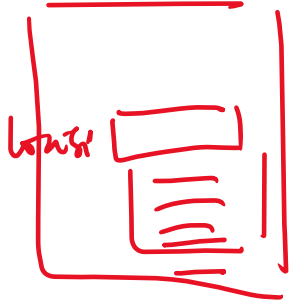
Hàm - Ví dụ

```
--Xóa hàm nếu đã có
if object_id('fDSach','IF') is not null
    drop function fDSach
go

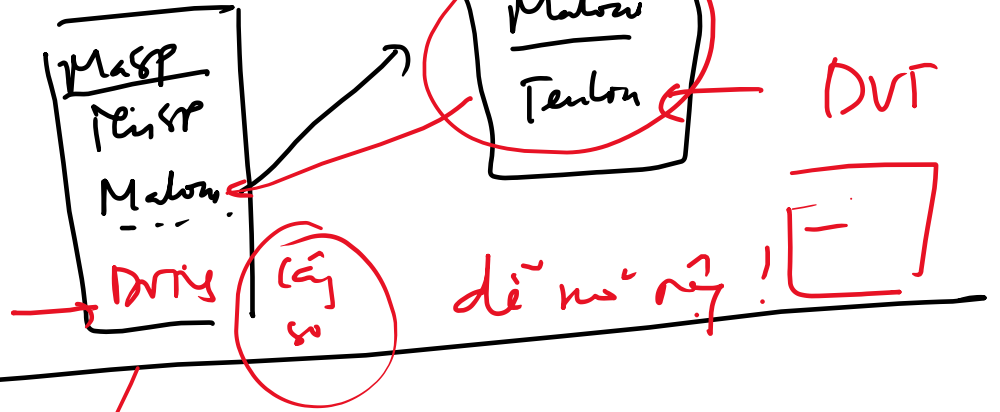
--Tạo hàm, giả sử trong CSDL ta đã có bảng T(namsinh int)
Create function fDSach (@ns int) --phải đặt tham số vào dấu ngoặc nhọn
Returns table
As
    Return (select * From T Where namsinh=@ns)
go
```

```
--Kiểm tra thử hàm
Select *
From fDSach(1982) --không cần dbo.
```

App ←



C2
= SP



Câu hỏi

10
20

BÁ

Ư/Ư Desg ← → DBA

3 by

1.00