TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM ĐÀO TẠO QUỐC TẾ - ITEC
MÔN HỌC: CƠ SỞ DỮ LIỆU NÂNG CAO

TÀI LIỆU HƯỚNG DẪN THỰC HÀNH

MÔ TẢ CSDL QUẢN LÝ THƯ VIỆN

DocGia (madg, hoten, socmnd, ngsinh, gioitinh, diachi, email, matkhau)

Độc giả có mã độc giả duy nhất, họ tên, số cmnd của độc giả, ngày sinh, giới tính, địa chỉ thường trú, email và mật khẩu giúp độc giả đăng nhập vào hệ thống thư viện điện tử.

DauSach (isbn, tensach, tacgia, namxb, nhaxb, soluong, mucgiaphat, theloai)

Thư viện quản lý các đầu sách theo mã isbn, một đầu sách sẽ có thông tin tên sách, tác giả, năm xuất bản, nhà xuất bản, số lượng cuốn sách hiện đang có trong thư viện của đầu sách đó, mức giá phạt mỗi ngày khi độc giả trả trễ sách thuộc đầu sách này, đầu sách sẽ thuộc một thể loại (“khoa học cơ bản”, “khoa học ứng dụng”, “xã hội”, “ngoại ngữ”).

CuonSach (isbn, masach, tinhtrang)

Mỗi cuốn sách thuộc một đầu sách có mã sách duy nhất để phân biệt các cuốn sách khác nhau của cùng một đầu sách. Cuốn sách sẽ có tình trạng là “đang được mượn” nếu sách đang được độc giả mượn hoặc “có thể cho mượn” nếu sách đang ở thư viện và chưa có độc giả mượn.

PhieuMuon (mapm, madg, ngaymuon)

Mỗi phiếu mượn có một mã số duy nhất, do một độc giả mượn vào một ngày nhất định.

CT_PhieuMuon (mapm, isbn, masach, songayquydinh)

Một lần mượn độc giả có thể mượn nhiều cuốn sách khác nhau. Mỗi cuốn sách sẽ có số ngày quy định cho biết độc giả được giữ cuốn sách đó trong bao nhiêu ngày. Trong một lần mượn không được mượn nhiều cuốn sách thuộc cùng một đầu sách.

PhieuTra (mapt, mapm, ngaytra)

Mỗi phiếu trả có một mã số duy nhất, phiếu trả sẽ trả cho một phiếu mượn nhất định vào một ngày nhất định. Độc giả có thể trả sách nhiều lần cho cùng một phiếu mượn.

CT_PhieuTra (mapt, isbn, masach, mucgiaphat, tienphat)

Một lần trả sách độc giả có thể trả nhiều cuốn sách. Ứng với từng cuốn sách trả sẽ có mức giá phạt từng ngày và tiền phạt độc giả phải đóng trong trường hợp trả sách trễ. Tiền phạt được tính theo công thức bên dưới:

$$tienphat = mucgiaphat * (ngaytra - ngaymuon - songayquydinh)$$

NỘI DUNG 1 – ÔN TẬP TRUY VẤN

I. HƯỚNG DẪN

a. Truy vấn cơ bản

SELECT thuocTinhA [tta], thuocTinhB [ttb] …
FROM bangR R, bangS S, …
WHERE <điều kiện lọc dữ liệu>
ORDER BY thuocTinhA asc | desc, ...

- **Select:** danh sách các thuộc tính cần hiển thị ở kết quả
- **From:** danh sách các bảng sử dụng để truy xuất dữ liệu
- **Where:** điều kiện lọc dữ liệu theo dòng, chỉ các dòng dữ liệu thỏa mãn điều kiện ở Where mới được đưa vào kết quả
- **Order by:** liệt kê các thuộc tính sử dụng để sắp xếp kết quả trả ra (asc – sắp tăng theo thuộc tính, desc – sắp giảm theo thuộc tính)

Ví dụ 1: Cho biết các độc giả (madg, hoten, cmnd) sử dụng địa chỉ yahoomail. Sắp kết quả tăng dần theo cmnd.

```
select dg.madg, dg.hoten, dg.socmnd
from DocGia dg where dg.email like '%@yahoo.com%'
order by dg.socmnd
```

Ví dụ 2: Cho biết các đầu sách được xuất bản trong giai đoạn 2000 đến 2014 có mức giá phạt theo ngày trên 10000.

```
select * from DauSach ds
where (ds.namxb between 2000 and 2004) and (ds.mucgiaphat > 10000)
```

Ví dụ 3: Cho biết tên, số cmnd và tuổi các độc giả có mượn sách "Toán cao cấp A1".

```
select dg.hoten, dg.socmnd, YEAR(getdate()) - YEAR(dg.ngsinh) as TuoiDG
from DocGia dg, PhieuMuon pm, CT_PhieuMuon ct, DauSach ds
where dg.madg = pm.madg and pm.mapm = ct.mapm and ct.isbn = ds.isbn and
      ds.tensach like N'Toán cao cấp A1'
```

Ví dụ 4: Cho biết tên, số cmnd các độc giả không cung cấp email từng trả sách trễ.

```
select dg.socmnd , dg.hoten
from DocGia dg, PhieuMuon pm, PhieuTra pt, CT_PhieuTra ct
where pm.madg = dg.madg and pm.mapm = pt.mapm and
      pt.mapt = ct.mapt and ct.tienphat > 0 and dg.email is null
```

b. Truy vấn lồng: trong các mệnh đề select, from hoặc where có sử dụng một câu truy vấn khác. Truy vấn bên ngoài gọi là truy vấn cha, truy vấn được lồng bên trong các mệnh đề của truy vấn cha được gọi là truy vấn con. Truy vấn cha và con ở mệnh đề where có thể được nối với nhau bằng (NOT) **IN**, (NOT) **EXISTS**, **ALL** hoặc **ANY**

SELECT thuocTinhA [tta], thuocTinhB [ttb] …
FROM bangR R, bangS S, …
WHERE …(SELECT … FROM ... WHERE …)

Truy vấn con

Ví dụ 5: Cho biết tên đầu sách có mức tiền phạt cao nhất.

```
select ds.tensach
from DauSach ds
where ds.mucgiaphat >= All (select ds1.mucgiaphat
                            from DauSach ds1)
```

Ví dụ 6: Cho biết tên và số cmnd của độc giả lớn tuổi hơn một độc giả ở TP.HCM.

```
select * from DocGia dg
where dg.ngsinh < ANY (select dg1.ngsinh from DocGia dg1
                       where dg1.diachi like N'%TP.HCM%')
```

Ví dụ 7: Cho biết mã cuốn sách chưa từng được mượn.

Cách 1: dùng IN

```
select cs.masach from CuonSach cs
where cs.masach not in (select masach from CT_PhieuMuon)
```

Cách 2: dùng EXISTS

```
select cs.masach from CuonSach cs
where not exists (select ct.masach from CT_PhieuMuon ct
                  where ct.masach = cs.masach)
```

c. Bảng tóm tắt một số điều kiện lọc dữ liệu

Điều kiện	Cú pháp	Ý nghĩa	Ví dụ
(not) like	ttA like chuoi	Toán tử so sánh chuỗi	socmnd not like '1%' hoten like N'Lê%'
<, >, =, !=, <=, >=	ttA = gia_tri	Toán tử so sánh	soluong >= 8
(not) between … and	ttA between a and b	Kiểm tra giá trị ttA có thuộc khoảng a và b	soluong between 5 and 10
is (not) null	ttA is null	Kiểm tra ttA có null không	email is null

(not) exists	exists (select * from …)	Kiểm tra tồn tại	exists (Select * From DocGia Where diachi like N'TP.HCM')
(not) IN	ttA IN (gia_tri_a, …)	Kiểm tra ttA có bằng 1 trong các giá trị phía sau IN	madg not IN (Select madg From PhieuMuon) namxb in (1990, 1996)
ALL	ttA >, =, … ALL (gia_tri_a, …)	Kiểm tra ttA có lớn hơn, bằng, … tất cả các giá trị sau ALL	madg != ALL (Select madg From PhieuMuon) namxb !=ALL (1990, 1996)
ANY	ttA <, =, .. ANY (gia_tri_a, …)	Kiểm tra ttA có nhỏ hơn, bằng, …ít nhất một giá trị sau ANY	madg =ANY (Select madg From PhieuMuon) namxb =ANY (1990, 1996)

d. Bảng tóm tắt một số hàm thường sử dụng

Hàm	**Ý nghĩa**	**Ví dụ**
isnull (ttA, gia_tri)	Gán ttA = gia_tri trong trường hợp ttA là null	isnull (email, '-')
round (ttA, n)	Làm tròn ttA với n số lẻ phía sau.	round (0.999, 2)
getdate()	Lấy ngày giờ hiện hành của hệ thống	getdate()
datepart (p, ngayA)	Lấy thành phần p của ngayA, p có thể là: - dd – ngày - mm – tháng - yyyy – năm - hh – giờ - mi – phút - ss - giây	datepart (dd, '12/26/1991') ⇨ trả ra 26 datepart (mm, '12/26/1991') ⇨ trả ra 12
datediff (p, ngayA, ngayB)	Tính khoảng cách ngayA và ngayB theo p, p có thể là: - dd – ngày	datediff (yy, '1/1/1996, '1/1/1997') ⇨ trả ra 1

Hàm	Ý nghĩa	Ví dụ
	- mm – tháng - yyyy hoặc yy – năm - hh – giờ - mi – phút - ss - giây	datediff (dd, '1/1/1996, '1/1/1997') ⇨ trả ra 365
convert (p, ttA)	Chuyển ttA sang kiểu dữ liệu p	convert (char(4), namxb)
cast (ttA as p)		cast (namxb as char(4))

II. BÀI TẬP

Viết câu truy vấn cho biết:

1. Họ tên và cmnd các độc giả trên 30 tuổi có địa chỉ ở TP.HCM.
2. Họ tên và cmnd các độc giả nữ sử dụng địa chỉ gmail có tên bắt đầu bằng C hoặc T. Sắp kết quả tăng dần theo số cmnd.
3. Tên các đầu sách có số lượng từ 3 đến 10. Sắp giảm dần theo số lượng.
4. Họ tên và cmnd các độc giả từng mượn sách vào ngày 20 tháng 12.
5. Họ tên và cmnd các độc giả từng mượn sách "Toán cao cấp A1" vào năm 2010.
6. Số isbn và tên của đầu sách được xuất bản lâu năm nhất.
7. Họ tên và cmnd của độc giả nam lớn tuổi nhất trong số các độc giả nam.
8. Tên đầu sách có mức phạt theo ngày thấp nhất khi độc giả trả sách trễ.
9. Tên đầu sách từng được các độc giả sinh một trong các năm sau mượn: 1974, 1986, 1990 hoặc 1992.
10. Họ tên và số cmnd độc giả có cung cấp thông tin email và chưa từng mượn sách được xuất bản vào các năm 2000, 2005 hoặc 2009.
11. Thông tin lần mượn sách gần nhất (mã phiếu mượn, ngày mượn) của các độc giả mang họ "Lê" hoặc "Trần".
12. Họ tên và số cmnd độc giả từng trả tiền phạt cao nhất cho một cuốn sách trả trễ.

NỘI DUNG 2 – TRUY VẤN NÂNG CAO

I. HƯỚNG DẪN

a. Truy vấn gom nhóm: được sử dụng khi có nhu cầu

SELECT thuocTinhA [tta], thuocTinhB [ttb] …
FROM bangR R, bangS S, …
WHERE <điều kiện lọc dữ liệu – **TRÊN DÒNG**>
GROUP BY thuocTinhA, thuocTinhB, ...
HAVING <điều kiện lọc dữ liệu – **TRÊN NHÓM**>
ORDER BY thuocTinhA asc | desc, ...

- **Group by**: thuộc tính gom nhóm, các dòng dữ liệu có giá trị giống nhau ở thuộc tính góm nhóm sẽ được xếp vào cùng một nhóm
- **Having:** điều kiện lọc dữ liệu **theo nhóm** → chỉ các **nhóm** dữ liệu thỏa mãn điều kiện ở Having mới được đưa vào kết quả
- **Một số hàm kết hợp thường được sử dụng sau khi gom nhóm dữ liệu:** min, max, count, sum, average.

b. Một số lưu ý

❖ Hàm kết hợp min, max, … **KHÔNG** được xuất hiện ở mệnh đề WHERE vì các hàm này chỉ được dùng cho các nhóm dữ liệu
❖ Sau khi thực hiện phép toán gom nhóm, chỉ có các thuộc tính có ở biểu thức gom nhóm GROUP BY mới được sử dụng ở SELECT hoặc HAVING, nếu không thì các thuộc tính này phải được đưa vào các hàm kết hợp min, max, count, …
❖ Khi cần sử dụng hàm kết hợp min, max, … nhưng không có nhu cầu chia nhóm dữ liệu thì biểu thức GROUP BY có thể bỏ. Tuy nhiên trường hợp này được xem như gom nhóm theo thuộc tính rỗng, do đó các thuộc tính muốn được sử dụng ở SELECT hoặc HAVING **PHẢI** được đưa vào các hàm kết hợp.

Ví dụ 1: Cho biết tên đầu sách và số lượng cuốn sách đang trong tình trạng "có thể mượn" của đầu sách này.

```
select ds.tensach, COUNT(cs.masach)as SoLuong
from DauSach ds, CuonSach cs
where ds.isbn = cs.isbn and cs.tinhtrang like N'có thể mượn'
group by ds.isbn, ds.tensach
```

Ví dụ 2: Đếm số lượng độc giả nam.
Cách 1: lọc điều kiện giới tính trước khi gom nhóm → điều kiện đặt tại WHERE

```
select COUNT(*) from DocGia where gioitinh = N'Nam'
```

Cách 2: lọc điều kiện giới tính sau khi gom nhóm → điều kiện đặt tại HAVING

```
select COUNT(*) from DocGia group by gioitinh
having gioitinh = N'Nam'
```

Ví dụ 3: Cho biết tên đầu sách có số lượng cuốn sách đang trong tình trạng "đang được mượn" ít nhất.

```
select ds.tensach
from DauSach ds, CuonSach cs
where ds.isbn = cs.isbn and cs.tinhtrang like N'đang được mượn'
group by ds.isbn, ds.tensach
having COUNT(cs.masach) <= ALL(select count(cs2.masach)
                               from CuonSach cs2
                               where cs.tinhtrang like N'đang được mượn'
                               group by cs2.isbn)
```

II. BÀI TẬP

Viết câu truy vấn cho biết:

1. Số lượng độc giả có địa chỉ tại Thủ Đức.
2. Đếm số phiếu mượn trong ngày 25 tháng 12.
3. Đếm số lượng đầu sách xuất bản sau năm 2000 hiện có trong thư viện.
4. Số isbn, mã sách và số lượng độc giả đã từng mượn cuốn sách này.
5. Số cmnd, họ tên và số lượng đầu sách mà độc giả này đã từng mượn.
6. Mã phiếu trả, ngày trả và tổng tiền phạt của phiếu trả đó.
7. Số isbn, mã sách của cuốn sách có nhiều độc giả mượn nhất.
8. Số cmnd, họ tên độc giả mượn ít đầu sách nhất.
9. Ngày có ít phiếu trả nhất.
10. Phiếu trả có tổng tiền phạt lớn nhất.
11. Ngày có nhiều phiếu mượn nhất.
12. Số cmnd, họ tên độc giả của phiếu mượn có nhiều sách được mượn nhất.
13. Số cmnd, họ tên độc giả của phiếu mượn có nhiều phiếu trả nhất.

NỘI DUNG 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)*.

NỘI DUNG 4 – THỦ TỤC THƯỜNG TRÚ (STORED PROCEDURE)

I. HƯỚNG DẪN

Ví dụ 1: Viết stored procedure nhận vào một ngày, xuất ra danh sách các phiếu trả trong ngày đó (mã phiếu trả, số cmnd độc giả của phiếu trả đó).

```
create proc sp_timPhieuTra @ngay date
as
begin
    select pt.mapt, dg.socmnd from PhieuTra pt, PhieuMuon pm, DocGia dg
    where pt.mapm = pm.mapm and pm.madg = dg.madg and pt.ngaytra = @ngay

end
go

exec sp_timPhieuTra '11/01/2015'
```

Ví dụ 2: Viết stored procedure nhận vào một số cmnd, xuất ra danh sách các phiếu mượn của người này (mã phiếu mượn, ngày mượn và tổng số sách mượn của phiếu mượn đó).

```
create proc sp_timPhieuMuon @cmnd nchar(10)
as
begin
    select pm.mapm, pm.ngaymuon, count(ct.isbn) as sosachmuon
    from PhieuMuon pm, DocGia dg, CT_PhieuMuon ct
    where pm.mapm = ct.mapm and pm.madg = dg.madg and dg.socmnd = @cmnd
    group by pm.mapm, pm.ngaymuon
end
go
exec sp_timPhieuMuon '2211015325'
```

Ví dụ 3: Viết stored procedure nhận vào một số cmnd, xóa độc giả mang số cmnd này theo các bước sau:

B1: Kiểm tra số cmnd có tồn tại không → nếu không trả ra mã lỗi là 1

B2: Kiểm tra độc giả có từng mượn sách không → nếu có trả ra mã lỗi là 2

B3: Nếu thỏa mãn điều kiện thực hiện xóa độc giả và trả về 0 báo hiệu xóa thành công và không có lỗi phát sinh.

```
create proc sp_xoaDocGia @cmnd nchar(10)
as
begin
    --neu khong ton tai doc gia co socmnd can xoa --> loi 1
    if not exists (select * from DocGia where socmnd = @cmnd)
        return 1

    --neu doc gia tung muon sach --> loi 2
    if exists (select * from PhieuMuon pm, DocGia dg
               where pm.madg = dg.madg and dg.socmnd = @cmnd)
        return 2

    --neu thoa 2 dieu kien tren thi xoa doc gia va tra ve 0
    delete from DocGia where socmnd = @cmnd
    return 0

end
go
```

II. BÀI TẬP

Viết stored procedure thực hiện các yêu cầu sau:

1. Nhận vào mã phiếu mượn, trả ra số cmnd, họ tên, địa chỉ, ngày sinh của độc giả mượn phiếu đó.
2. Nhận vào một năm, xuất ra thông tin các độc giả sinh năm đó.
3. Xuất ra thông tin độc giả trẻ tuổi nhất.
4. Nhận vào một mã phiếu mượn, trả ra số cmnd, họ tên độc giả mượn và tổng số cuốn sách được mượn trong lần đó.
5. Nhận vào số isbn một đầu sách, xuất ra danh sách độc giả (số cmnd, họ tên, ngsinh, địa chỉ) của các độc giả từng mượn đầu sách đó.
6. Nhận vào số cmnd một độc giả, xuất ra thông tin các cuốn sách (mã isbn, mã sách) mà độc giả đã từng mượn.
7. Nhận vào một mã phiếu mượn trả ra số lượng phiếu trả cho phiếu mượn đó.
8. Nhận vào một mã isbn, xóa đầu sách mang mã isbn đó theo các bước sau:
    - Kiểm tra đầu sách có tồn tại không → nếu không trả về mã lỗi là 1
    - Kiểm tra đầu sách đó đã có cuốn sách nào chưa
        - Nếu chưa có cuốn sách nào thuộc đầu sách, tiến hành xóa đầu sách và trả về 0 báo hiệu xóa thành công.
        - Nếu có, kiểm tra các cuốn sách thuộc đầu sách đã từng có độc giả mượn hay trả chưa:
            - Nếu chưa, xóa các cuốn sách thuộc đầu sách sau đó xóa đầu sách và trả về 0 báo hiệu xóa thành công.
            - Nếu có, không được xóa và trả về mã lỗi là 2.
9. Nhận vào mã phiếu trả, cập nhật ngày trả theo các bước sau:
    - Kiểm tra mã phiếu trả có tồn tại không → nếu không trả về mã lỗi là 1
    - Cập nhật ngày trả của phiếu trả

- Cập nhật tiền phạt của các chi tiết phiếu trả tương ứng theo công thức:
  *tiền phạt = mucgiaphat * (ngày trả mới – ngày mượn – số ngày quy định)*

10. Nhận vào thông tin một phiếu mượn (mã phiếu mượn, mã độc giả), thêm phiếu mượn vào CSDL theo các bước sau:
    - Kiểm tra mã phiếu mượn đã tồn tại chưa → nếu đã tồn tại trả về mã lỗi là 1
    - Kiểm tra mã độc giả phải khác null và phải tồn tại trong bảng độc giả → nếu không trả về mã lỗi 2
    - Thêm phiếu mượn vào CSDL, biết rằng ngày mượn luôn là ngày hiện tại của hệ thống. Trả về 0 báo hiệu thêm thành công.

NỘI DUNG 5 – GIAO TÁC (TRANSACTION)

I. HƯỚNG DẪN

a. Định nghĩa và tính chất

Định nghĩa:

- Tập hợp các thao tác trên CSDL trong đó tất cả các bước của giao tác phải được thực hiện thành công, nếu 1 trong các bước thất bại phải hủy toàn bộ giao tác.
- CSDL được chuyển từ trạng thái nhất quán này sang trạng thái nhất quán khác sau khi thực hiện giao tác.

Tính chất ACID:

- **A**utomicity – tính nguyên tố: các thao tác trong giao tác không thể chia cắt.
- **C**onsistency – tính nhất quán: dữ liệu không được ở trạng thái bất hợp lý (không nhất quán) sau khi thực hiện giao tác
- **I**solation – tính cô lập: việc thực hiện giao tác là độc lập, không phụ thuộc vào giao tác khác
- **D**urability – tính bền vững: kết quả thực hiện giao tác được lưu trữ bền vững trong CSDL (commit)

Ví dụ mẫu: giao tác thực hiện chức năng chuyển **khoản tiền x** từ **tài khoản A** sang tài **khoản B**

Bước 1: Kiểm tra tài khoản A còn đủ khoản tiền x để chuyển hay không, nếu đủ qua bước 2

Bước 2: Trừ khoản tiền x từ tài khoản A

Bước 3: Cộng thêm khoản tiền x vào tài khoản B

Các bước không thể tách rời, nếu 1 trong các bước của giao tác không thành công phải hủy bỏ giao tác, nếu không dữ liệu có thể bị mất tính nhất quán. VD: thực hiện bước 1 & 2 thành công nhưng bước 3 thất bại → trừ tiền tài khoản A nhưng không cộng thêm tiền vào tài khoản B

b. Cú pháp khai báo giao tác

Giao tác được bắt đầu bằng BEGIN TRANSACTION và kết thúc bằng COMMIT TRANSACTION nếu thành công hoặc ROLLBACK TRANSACTION nếu thất bại.

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

```
        IF ERROR
        BEGIN
            RAISERROR (N'Lỗi …', 16, 1)
            ROLLBACK TRANSACTION -- hủy giao tác khi phát sinh lỗi
        END
        COMMIT TRANSACTION -- cập nhật thay đổi khi thực hiện giao tác thành công
    END
    GO
```

Ví dụ 1: Viết giao tác thêm một đầu sách mới.

```
create proc sp_themDauSach
    @isbn nchar(12), @tensach nvarchar(100), @tacgia nvarchar(50), @namxb int,
    @nhaxb nchar(20), @soluong int, @mucphat float, @theloai nvarchar(70)
as
begin transaction
    --kiem tra ma isbn ton tai tra ve 1
    if exists (select * from DauSach ds where ds.isbn = @isbn)
    begin
        rollback transaction
        return 1
    end
    --kiem tra so luong khac 0 thi tra ve loi 2
    if @soluong!=0
    begin
        rollback transaction
        return 2
    end
    --kiem tra the loai phai la
    --"khoa học cơ bản", "khoa học ứng dụng", "xã hội", "ngoại ngữ"
    --neu khong tra ve loi 3
    if @theloai not in (N'khoa học cơ bản',N'khoa học ứng dụng',N'xã hội',N'ngoại ngữ')
    begin
        rollback transaction
        return 3
    end
    insert into DauSach values(@isbn,@tensach,@tacgia,@namxb,@nhaxb,@soluong,@mucphat,@theloai)
    commit transaction
    return 0
end
```

II. BÀI TẬP

Viết giao tác (được lồng vào stored procedure) thực hiện yêu cầu sau:

1. Thêm độc giả.
    - Input: thông tin độc giả
    - Output: 0 – thêm thành công hoặc mã lỗi (số nguyên > 0) nếu thêm thất bại.
    - Các bước thực hiện:
        - B1: Kiểm tra mã độc giả tồn tại → nếu có trả về lỗi 1
        - B2: Kiểm tra email trùng → nếu trùng trả về lỗi 2
        - B3: Kiểm tra số cmnd trùng → nếu trùng trả về lỗi 3
        - B4: Kiểm tra độc giả phải từ 18 tuổi trở lên → nếu không đủ trả về lỗi 4
        - B5: Kiểm tra giới tính phải là nam, nữ hoặc null → nếu không trả về lỗi 6
        - B6: Thêm độc giả vào, trả về 0.

2. Thêm một cuốn sách.
    - Input: thông tin cuốn sách
    - Output: 0 – thêm thành công hoặc mã lỗi (số nguyên > 0) nếu thêm thất bại.
    - Các bước thực hiện:
        - B1: Kiểm tra isbn phải tồn tại ở đầu sách → nếu không trả về lỗi 1
        - B2: Kiểm tra mã sách có trùng với cuốn sách khác thuộc cùng đầu sách → nếu trùng trả về lỗi 2
        - B3: Kiểm tra tình trạng phải là "có thể mượn" → nếu không trả về lỗi 3
        - B4: Thêm cuốn sách vào, trả về 0.

3. Thêm một phiếu mượn.
    - Input: thông tin phiếu mượn (không truyền vào ngày mượn)
    - Output: 0 – thêm thành công hoặc mã lỗi (số nguyên > 0) nếu thêm thất bại.
    - Các bước thực hiện:
        - B1: Kiểm tra madg phải tồn tại ở độc giả → nếu không trả về lỗi 1
        - B2: Kiểm tra mã phiếu có trùng với phiếu khác → nếu trùng trả về lỗi 2
        - B3: Thêm phiếu mượn vào với ngày mượn là ngày hiện tại của hệ thống, trả về 0.

4. Thêm một chi tiết cho phiếu mượn.
    - Input: thông tin chi tiết phiếu mượn
    - Output: 0 – thêm thành công hoặc mã lỗi (số nguyên > 0) nếu thêm thất bại.
    - Các bước thực hiện:
        - B1: Kiểm tra mapm phải tồn tại ở phiếu mượn → nếu không trả về lỗi 1
        - B2: Kiểm tra cuốn sách được mượn (isbn, mã sách) phải tồn tại và tình trạng phải là có thể mượn → nếu không thỏa trả về lỗi 2
        - B3: Kiểm tra số ngày quy định phải có và là số nguyên dương → nếu không trả về lỗi 3
        - B4: Thêm chi tiết phiếu mượn vào, trả về 0.

5. Thêm một phiếu trả.
    - Input: thông tin phiếu trả (không truyền vào ngày trả)
    - Output: 0 – thêm thành công hoặc mã lỗi (số nguyên > 0) nếu thêm thất bại.
    - Các bước thực hiện:
        - B1: Kiểm tra mapm phải tồn tại ở phiếu mượn → nếu không trả về lỗi 1
        - B2: Kiểm tra mã phiếu có trùng với phiếu khác → nếu trùng trả về lỗi 2
        - B3: Thêm phiếu trả vào với ngày trả là ngày hiện tại của hệ thống, trả về 0.

6. Thêm một chi tiết cho phiếu trả.
    - Input: thông tin phiếu trả (không truyền vào mức giá phạt và tiền phạt)

- Output: 0 – thêm thành công hoặc mã lỗi (số nguyên > 0) nếu thêm thất bại.
- Các bước thực hiện:
  - B1: Kiểm tra mapt phải tồn tại ở phiếu trả → nếu không trả về lỗi 1
  - B2: Kiểm tra cuốn sách được trả (isbn, mã sách) phải **là cuốn sách được mượn và chưa được trả** → nếu không thỏa trả về lỗi 2
  - B3: Thêm chi tiết phiếu trả vào, trả về 0. Lưu ý: mức giá phạt được lấy từ mức giá phạt hiện tại của đầu sách tương ứng và tiền phạt được tính theo công thức như bên dưới:

    *tienphat = mucgiaphat * (ngaytra – ngaymuon – songayquydinh)*

7. Cập nhật ngày mượn của một phiếu mượn.
   - Input: mã phiếu mượn và ngày mượn mới
   - Output: 0 – cập nhật thành công hoặc mã lỗi (số nguyên > 0) nếu thất bại.
   - Các bước thực hiện:
     - B1: Kiểm tra mapm phải tồn tại → nếu không trả về lỗi 1
     - B2: Cập nhật ngày mượn
     - B3: Cập nhật lại tiền phạt cho các chi tiết phiếu trả ứng với phiếu mượn vừa được cập nhật ngày
     - B4: Trả về 0 báo hiệu thành công

8. Cập nhật mức giá phạt cho một chi tiết trong phiếu trả.
   - Input: mã phiếu trả, số isbn, mã sách và mức giá phạt mới
   - Output: 0 – cập nhật thành công hoặc mã lỗi (số nguyên > 0) nếu thất bại.
   - Các bước thực hiện:
     - B1: Kiểm tra mapt phải tồn tại → nếu không trả về lỗi 1
     - B2: Kiểm tra cuốn sách (số isbn, mã sách) phải tồn tại trong chi tiết phiếu trả → nếu không trả về lỗi 2
     - B3: Kiểm tra mức giá phạt phải là số dương → nếu không trả về lỗi 3
     - B4: Cập nhật mức giá phạt
     - B5: Cập nhật lại tiền phạt với mức giá phạt mới
     - B6: Trả về 0 báo hiệu thành công

9. Xóa độc giả.
   - Input: mã độc giả
   - Output: 0 – xóa thành công hoặc mã lỗi (số nguyên > 0) nếu thất bại.
   - Các bước thực hiện:
     - B1: Kiểm tra madg phải tồn tại → nếu không trả về lỗi 1
     - B2: Kiểm tra độc giả đã từng mượn hay trả sách chưa → nếu rồi trả về lỗi 2 báo hiệu không được xóa
     - B3: Xóa độc giả và trả về 0 báo hiệu xóa thành công

10. Xóa phiếu mượn.
    - Input: mã phiếu mượn
    - Output: 0 – xóa thành công hoặc mã lỗi (số nguyên > 0) nếu thất bại.
    - Các bước thực hiện:
        - B1: Kiểm tra mapm phải tồn tại → nếu không trả về lỗi 1
        - B2: Kiểm tra phiếu mượn đã có phiếu trả tương ứng chưa → nếu có trả về lỗi 2 báo hiệu không được xóa
        - B3: Xóa chi tiết phiếu mượn tương ứng
        - B4: Xóa phiếu mượn
        - B5: Trả về 0 báo hiệu thành công

11. Xóa phiếu trả.
    - Input: mã phiếu trả
    - Output: 0 – xóa thành công hoặc mã lỗi (số nguyên > 0) nếu thất bại.
    - Các bước thực hiện:
        - B1: Kiểm tra mapt phải tồn tại → nếu không trả về lỗi 1
        - B2: Xóa chi tiết phiếu trả tương ứng
        - B3: Xóa phiếu trả
        - B4: Trả về 0 báo hiệu thành công

NỘI DUNG 6 – FUNCTION

NỘI DUNG 7 – TRIGGER

I. HƯỚNG DẪN

1. **Đặc điểm**: trigger là 1 loại thủ tục đặc biệt, không tham số, thuộc duy nhất 1 bảng, được tự động thực thi khi người dùng cố gắng thay đổi chỉnh sửa dữ liệu qua các lệnh INSERT, UPDATE, DELETE trên table hay view. Hoạt động dựa trên hai bảng tạm INSERTED, DELETED.
2. **Mục tiêu**: thường được dùng để kiểm tra các ràng buộc toàn vẹn (data integrity) và quy tắc nghiệp vụ (business rules)
3. **Cú pháp**:

```
Create trigger tên_trigger
On {tên_bảng|tên_view}                    <- Quan hệ bị ảnh hưởng
{For| After| Instead of } { [delete] [,] [insert] [,]  [update] }   <- Thao tác bị ảnh hưởng
As
Begin
        { các lệnh T-sql xử lý khi RBTV bị vi phạm}
End
Go
```

Lưu ý: Nếu thao tác insert/ delete/ update thực hiện trên nhiều dòng, trigger cũng chỉ được gọi một lần → Bảng inserted/ deleted có thể chứa nhiều dòng

Ví dụ: Cho CSDL có 2 quan hệ

- DonHang (MaDH,…,NgayDatHang)
- PhieuGH (MaPG, MaDH,…,NgayGiaoHang)

RBTV : Ngày giao hàng phải sau ngày đặt hàng và không trễ quá 1 thàng kể từ ngày đặt hàng.

Bước 1: Vẽ bảng tầm ảnh hưởng

	insert	delete	update
DonHang	−	−	+ *(NgayDatHang)*
PhieuGH	+	−	+ *(MaDH,NgayGiaoHang)*

Bước 2: Tạo trigger

```
Create trigger trg_DH_PGH On DonHang
For update
As
Begin
        If exists(select * from Insreted I, PhieuGH P Where P.MaDH=I.MaDH And
        (P.NgayGiaoHang < I.NgayDatHang Or Datediff (MM, I.NgayDatHang,
        P.NgayGiaoHang) > 1))
```

```
        Begin
            Raiserror('Ngay dat hang khong hop le',0,1)
            Rollback transaction
        End
End
```

4. **Sử dụng lệnh alter để thay đổi nội dung trigger**

```
Alter trigger trg_DH_PGH On DonHang
For update, Insert
As
Begin
    If exists (select * from Insreted I, PhieuGH P Where P.MaDH=I.MaDH And
    (P.NgayGiaoHang < I.NgayDatHang Or Datediff (MM, I.NgayDatHang, P.NgayGiaoHang)
    > 1))
    Begin
        Raiserror('Thao tac da bi huy',0,1)
        Rollback transaction
    End
End
```

5. **Xóa trigger**

   Drop trigger *tên_trigger_cần _xóa [,...n]*

   Ví dụ:

   Drop trigger trg_DH_PGH

II. BÀI TẬP

1. Ngày mượn sách phải là ngày hiện tại
2. Ngày trả sách phải là ngày hiện tại.
3. Tình trạng sách chỉ bao gồm : "đang được mượn" hoặc "có thể cho mượn"
4. Đọc giả muốn mượn sách phải từ 18 tuổi trở lên.
5. Số lượng đầu sách hiện có >=0.
6. Thêm thuộc tính "SoLanTra" vào bảng PhieuMuon. SoLantra = số phiếu trả của phiếu mượn tương ứng.
7. Thêm thuộc tính "NgayTraDuKien" vào bảng CT_PhieuMuon. Ngày trả dự kiến = ngày mượn + số ngày quy định được mượn.
8. Tình trạng cuốn sách được tự động cập nhật "đang được mượn" khi thủ thư thực hiện cho độc giả mượn sách này.