KHAI BÁO CÁC RÀNG BUỘC TOÀN VẸN

- ✓ Primary key
- ✓ Foreign key
- ✓ Check
- ✓ Rule
- ✓ Trigger
- ✓ Function, Stored Procedure
- ✓ ….

RÀNG BUỘC TOÀN VẸN

THỰC HÀNH

Phân loại ràng buộc toàn vẹn

- 1. RBTV có bối cảnh trên 1 quan hệ
  - 1.1 Ràng buộc miền giá trị (check, rule)
  - 1.2 Ràng buộc liên thuộc tính (check, rule)
  - 1.3 Ràng buộc liên bộ (primary key, trigger)
- 2. RBTV có bối cảnh trên nhiều quan hệ
  - 2.1 Ràng buộc tham chiếu (foreign key)
  - 2.2 Ràng buộc liên thuộc tính (trigger)
  - 2.3 Ràng buộc liên bộ (trigger,…)
  - 2.4 Ràng buộc thuộc tính tổng hợp (trigger, func, pro)
  - 2.5 Ràng buộc đồ thị có chu trình (trigger, …)
- 3. Phụ thuộc hàm (functional dependency) (trigger,…)

RULE

```
--Tạo RULE thuộc tính GIOITINH chỉ có giá trị là ‘Nam’ hoặc ‘Nữ’.
CREATE RULE GT AS @GT IN ('Nam','Nu’)
--Gắn rule GT cho 2 thuộc tính GIOITINH của HOCVIEN và GIAOVIEN
sp_bindrule GT,'HOCVIEN.GIOITINH'
sp_bindrule GT,'GIAOVIEN.GIOITINH'
-- bỏ rule
sp_unbindrule 'HOCVIEN.GIOITINH'
```

Trigger

Tạo Trigger

```
CREATE TRIGGER <trigger_name> ON <table name>
[WITH ENCRYPTION]
AFTER | FOR DELETE, INSERT, UPDATE
AS <Các phát biểu T-sql>
```

Trong đó:

- trigger_name: Tên trigger phải phân biệt.
- ON <tablename>: tên table mà trigger sẽ thực hiện. Không sử dụng cú pháp trigger này cho View.
- WITH ENCRYPTION: Mã hóa trigger, không cho xem và sửa đổi câu lệnh tạo trigger..
- FOR DELETE, INSERT, UPDATE: Dùng chỉ định những lệnh cập nhật nào trên table sẽ kích hoạt trigger. Khi thực hiện trigger, SQL sẽ tạo các bảng tạm: INSERTED và DELETED.

TRIGGER

```
--Cho luoc do quan he GV (magv, hoten, heso, luong)
create table GV
(
     magv varchar(4) primary key, hoten varchar(20),
     heso numeric(4,3), luong money
)
--R1: cap nhat luong tang
/*
R1|Them | Xoa | Sua
--|-----|-----|---------
GV|  -  | -   |+(Luong)
*/
```


R1: Cập nhật lương phải tăng

```
CREATE trigger R1 on GV for update as
if update(luong)
      begin
      declare @i int
      select @i=(select count(*) from inserted,deleted
                        where inserted.magv=deleted.magv and
                                    inserted.luong<=deleted.luong)
      if @i>0
            begin
            print 'Vi pham RBTV, cap nhat luong phai tang'
            rollback tran
            end
      end
go
```

R2: Các GV có cùng hệ số thì cùng mức lương

```
--Kiem tra
delete from GV
select * from GV
insert into gv(magv,heso,luong) values('gv01',3.0,3000)
insert into gv(magv,heso,luong) values('gv02',3.0,3000)
insert into gv(magv,heso,luong) values('gv03',3.3,3300)
insert into gv(magv,heso,luong) values('gv04',3.3,3500)
update gv set heso=3.6, luong=4000 where magv='gv03'
```


R1: Cập nhật lương phải tăng

```
--Thao tac tren du lieu de kiem tra rang buoc R1

insert into gv(magv,luong) values('gv02',3000)
update gv set luong=3300 where magv='gv02'
update gv set luong=2500 where magv='gv02'

select * from gv
```


R2: Các GV có cùng hệ số thì cùng mức lương

```
GV (magv, hoten, heso, luong)
--R2: Cung heso thi cung luong
/*
R2|Them | Xoa | Sua
--|-----|-----|---------
GV|  +  | -   |+(Heso,Luong)
*/
```


R2: Các GV có cùng hệ số thì cùng mức lương

```
create trigger R2 on GV for insert,update as
    declare @i int
    select @i=(select count(*) from GV,inserted
                where GV.heso=inserted.heso
                    and GV.luong<>inserted.luong)
    if @i>0
    begin
        raiserror('cung heso thi cung luong',16,1)
        rollback tran
    end
go
```

RÀNG BUỘC CÓ BỐI CẢNH NHIỀU QUAN HỆ

```
/*
HOADON(SOHD,NGHD,TRIGIA)
SANPHAM(MASP,GIA,NGNHAP)
CTHD(SOHD,MASP,SL)
*/
CREATE TABLE HOADON (SOHD INT PRIMARY KEY,NGHD SMALLDATETIME,
TRIGIA MONEY)
CREATE TABLE SANPHAM(MASP VARCHAR(4) PRIMARY KEY, GIA MONEY,
NGNHAP SMALLDATETIME)
CREATE TABLE CTHD(SOHD INT, MASP VARCHAR(4), SL INT, CONSTRAINT
PK_CTHD PRIMARY KEY (SOHD,MASP))
```


R3: Ngày bán hàng phải lớn hơn hoặc bằng ngày nhập sản phẩm đó

```
HOADON(SOHD,NGHD,TRIGIA)
SANPHAM(MASP,GIA,NGNHAP)
CTHD(SOHD,MASP,SL)
-------------------------
  R3    |Them | Xoa | Sua
--------|-----|-----|---------
HOADON  |  -  | -   |+(NGHD)
--------|-----|-----|---------
SANPHAM |  -  | -   |+(NGNHAP)
--------|-----|-----|---------
CTHD    |  +  | -   |+(SOHD,MASP)
```


R3: Ngày bán hàng phải lớn hơn hoặc bằng ngày nhập sản phẩm đó

```
Create trigger R3_SP on SANPHAM for update as
if update (ngnhap)
    IF EXISTS (Select * from CTHD c, HOADON h,
                inserted s where c.masp=s.masp and
                c.sohd=h.sohd and h.nghd<s.ngnhap)
    begin
        raiserror('Ngnhap phai nho hon hoac bang nghd',16,1)
        rollback tran
    end
```


R3: Ngày bán hàng phải lớn hơn hoặc bằng ngày nhập sản phẩm đó

```
Create trigger R3_HD on HOADON for update as
if update (nghd)
    If EXISTS (Select * from CTHD c, SANPHAM s,
            inserted h where c.masp=s.masp and
            c.sohd=h.sohd and h.nghd<s.ngnhap)
    begin
        raiserror('Ngay nhap phai nho hon hoac bang nghd',16,1)
        rollback tran
end
```

R4: Siso của một lớp là số lượng học viên thuộc lớp

```
Create trigger R4_up_lop on lop for update
as
if update(siso)
begin
     declare @malop varchar(4)
     select @malop=(select malop from inserted)
     update lop set siso=(select count(*) from hv
                          where malop=@malop)
               where malop=@malop
     end
go
```


R4: Siso của một lớp là số lượng học viên thuộc lớp

```
create trigger R4_ins_LOP on LOP for insert
as
update lop set siso=0
where malop=(select malop from inserted)
go
--Kiem tra
insert into LOP values('L1',10)
select * from LOP
```


R4: Siso của một lớp là số lượng học viên thuộc lớp

```
/*-------------------------------------------------------
LOP(Malop, Siso)
HV(Mahv,Malop)
-------------------------------------------------------*/
create table LOP (Malop varchar(4) primary key,Siso int)
create table HV(Mahv varchar(4) primary key,malop varchar(4))
go
--R4: siso của một lớp la so luong hoc vien thuoc lop do
 R4|Them | Xoa | Sua
---|-----|-----|---------
LOP|  +  | -   |+(Siso)
---|-----|-----|---------
 HV|  +  | +   |+(Malop)
```


R3: Ngày bán hàng phải lớn hơn hoặc bằng ngày nhập sản phẩm đó

```
Create trigger R3_CTHD on CTHD for insert,update
as
IF EXISTS (Select * from SANPHAM s,HOADON h,
            inserted c where c.masp=s.masp and
            c.sohd=h.sohd and h.nghd<s.ngnhap)
    begin
        raiserror('Ngnhap phai nho hon hoac bang nghd',16,1)
        rollback tran
    end
```

R4: Siso của một lớp là số lượng học viên thuộc lớp

```
create trigger R4_ins_hv on HV for insert
as
declare @malop varchar(4)
select @malop=(select malop from inserted)
update lop set siso=siso+1 where malop=@malop
go

create trigger R4_del_hv on HV for delete
as
declare @malop varchar(4)
select @malop=(select malop from deleted)
update lop set siso=siso-1 where malop=@malop
go
```

Đây chỉ xét trường hợp **đơn giản** thực hiện thao tác chỉ ảnh hưởng đến 1 bộ

R4: Siso của một lớp là số lượng học viên thuộc lớp

```
create trigger R4_up_dv on hv for update
as
declare @malopnew varchar(4)
declare @malopold varchar(4)
select @malopnew=(select malop from inserted)
select @malopold=(select malop from deleted)
update lop set siso=siso-1 where malop=@malopold
update lop set siso=siso+1 where malop=@malopnew
go
```


R4: Siso của một lớp là số lượng học viên thuộc lớp

```
--Kiem tra rang buoc tren quan he LOP
delete from lop
insert into lop values('L1',10)
insert into lop values('L2',10)
select * from lop

update lop set siso=3 where malop='L1'
select * from lop
```


R4: Siso của một lớp là số lượng học viên thuộc lớp

```
--Kiem tra rang buoc tren quan he HV
insert into HV values('HV01','L1')
insert into HV values('HV02','L1')
insert into HV values('HV03','L1')
insert into HV values('HV04','L1')
insert into HV values('HV05','L2')
insert into HV values('HV06','L2')
insert into HV values('HV07','L2')
insert into HV values('HV08','L2')
insert into HV values('HV09','L2')
select * from HV
select * from LOP
```

Cursor

R4: Siso của một lớp là số lượng học viên thuộc lớp

```
--Kiem tra rang buoc tren quan he HV
delete from HV where mahv='HV05'
select * from HV
select * from LOP

update HV set malop='L2' where mahv='HV01'
select * from HV
select * from LOP
```


GIỚI THIỆU CURSOR

Cursor là kiểu dữ liệu cho phép truy xuất đến từng dòng (record) trong tập kết quả trả về của câu SELECT.

- **Một số thao tác chung trên Cursor**
  - ✓Khai báo cursor: DECLARE <cursor_name> CURSOR FOR <lệnh Select>
  - ✓Mở cursor : OPEN <cursor_name>
    (Sau lệnh mở cursor, con trỏ mẫu tin hiện hành nằm ở vùng BOF.)
  - ✓Xử lý record trên cursor:
    Di chuyển record hiện hành: FETCH NEXT FROM cursor_name
    Sử dụng Update hoặc Delete để cập nhật hay xóa record hiện hành
  - ✓Đóng cursor: CLOSE <tên cursor>
  - ✓Hủy bỏ cursor: DEALLOCATE <TÊN CURSOR>

R4: Siso của một lớp là số lượng học viên thuộc lớp

TRƯỜNG HỢP TỔNG QUÁT CÁC THAO TÁC THÊM XÓA SỬA ẢNH HƯỞNG ĐẾN NHIỀU BỘ => sử dụng CURSOR

Khai báo Cursor

Phạm vi hoạt động

- Global: (mặc định) cursor có phạm vi Global trên kết nối mà nó đã được tạo. Khi không sử dụng nữa phải đóng và giải phóng Cursor.
- Local: có phạm vi hoạt động bên trong gói lệnh đã tạo nó và tự giải phóng khi kết thúc gói lệnh.

Phương Thức Di Chuyển

- FORWARD_ONLY: là phương thức mặc định, chỉ cho phép di chuyển sang record kế tiếp.
- SCROLL: Cho phép di chuyển lên xuống trong tập record.

Khai báo Cursor

```
DECLARE  <CursorName> CURSOR
[ LOCAL | GLOBAL ] -- Phạm vi hoạt động
[ FORWARD_ONLY | SCROLL ] -- Phương thức di chuyển
[ STATIC | KEYSET | DYNAMIC ] -- Loại Cursor
[ READ_ONLY | SCROLL_LOCKS | OPTIMISTIC ]--xử lý đồng thời
[ TYPE_WARNING]
FOR <lệnh Select>
[ FOR UPDATE [ OF ColumnName [,…n]] ]
```


Khai báo Cursor

Có 3 loại Cursor:

- STATIC: có thuộc tính READ ONLY, do đó không thể cập nhật các bảng gốc thông qua Cursor này. Khi tạo Cursor Static, dữ liệu (dl) từ các bảng gốc sẽ được Copy sang một bảng tạm trong CSDL tempdb. Do đó, Nếu các table nguồn của Cursor bị thay đổi dl thì các dl không xuất hiện trên Cursor.
- DYNAMIC: Cho phép cập nhật dl trên các table nguồn (dùng mệnh đề WHERE CURRENT OF <tênCursor> trong các lệnh UPDATE or DELETE), và tự động hiển thị tất cả những thay đổi từ table nguồn. Tuy nhiên, dl và thứ tự của các record trong tập record có thể bị thay đổi.
- KEYSET: Giống như cursor Dynamic. Nhưng nó chỉ được tạo khi bảng nguồn có khai báo khóa, nếu không thì SQL tự động chuyển sang loại STATIC. Khi tạo Cursor KEYSET, Tập các khóa của bảng nguồn được lưu trên một table của CSDL tempdb. Do đó, việc xóa record hoặc thay đổi giá trị khóa trên các bảng nguồn không thông qua Cursor sẽ không phản hồi trên tập record.

Cursor kiểu STATIC, KEYSET, và DYNAMIC mặc định dùng phương thức SCROLL.

- TYPE_WARNING: Gởi thông báo chú ý về client nếu Cursor thực hiện chuyển đổi ngầm định từ kiểu yêu cầu sang một kiểu khác.

Khai báo Cursor

Xử lý đồng thời

SQL Server cung cấp 4 chọn lựa cho phép ngăn cản việc sửa đổi record cho tới khi thực hiện xong hoặc bằng cách khóa các table nguồn của cursor để bảo vệ các thay đổi của bạn.

1. READ_ONLY: Dùng khi chỉ truy xuất dữ liệu mà không sửa đổi dữ liệu.
2. SCROLL_LOCKS: Khoá các dòng đã được đọc vào Cursor đối với các User khác.
3. OPTIMISTIC WITH VALUES: Chỉ khóa các giá trị mà bạn vừa thay đổi. Nếu người dùng khác thay đổi các giá trị đó sẽ nhận được thông báo lỗi.
4. OPTIMISTIC WITH ROW VERSIONING: Khi muốn cả dòng được cập nhật, không chỉ một vài Fields trong nó.

Khai báo Cursor

Truy xuất dữ liệu trên Cursor

```
FETCH [ NEXT | PRIOR | FIRST | LAST
| ABSOLUTE { n | @nvar } | RELATIVE { n | @nvar } ]
FROM [ GLOBAL ] cursor_name
[ INTO @variable_name [ ,...n ] ]
```

- NEXT: Chuyển sang mẫu tin kế tiếp.
- PRIOR: Chuyển về mẫu tin trước đó.
- FIRST: Chuyển về mẫu tin đầu tiên.
- LAST: Chuyển đến mẫu tin cuối cùng.
- ABSOLUTE {n | @nvar}: Nếu n hoặc @nvar>0, tìm đến dòng thứ n tính từ dòng đầu tiên đếm xuống trong tập record. Nếu n hoặc @nvar<0, tìm đến dòng thứ n tính từ dòng cuối cùng đếm lên. Nếu n hoặc @nvar=0, chuyển đến vùng BOF và không có giá trị trả về. Hằng số n phải là số nguyên và biến @nvar phải thuộc kiểu smallint, tinyint, hoặc int. Không sử dụng phương thức ABSOLUTE cho kiểu DYNAMIC.

R4: Siso của một lớp là số lượng học viên thuộc lớp

```
/*------------------------------------------------------------
LOP(Malop, Siso)
HV(Mahv,Malop)
------------------------------------------------------------*/
create table LOP (Malop varchar(4) primary key,Siso int)
create table HV(Mahv varchar(4) primary key,malop varchar(4))
go
--R4: siso của một lớp la so luong hoc vien thuoc lop do
 R4|Them | Xoa | Sua
---|-----|-----|---------
LOP|  +  | -   |+(Siso)
---|-----|-----|---------
 HV|  +  | +   |+(Malop)
```


Viết lại trigger del_hv bằng cách sử dụng Cursor

```
Create/alter trigger del_hv on hv for delete as
declare @malop varchar(20)
declare hv_cur cursor for select malop from deleted
open hv_cur
fetch next from hv_cur into @malop
WHILE @@FETCH_STATUS = 0
BEGIN
     update lop set siso=siso-1 where malop=@malop
     fetch next from hv_cur into @malop
END
CLOSE hv_cur
DEALLOCATE hv_cur
```


Khai báo Cursor

Truy xuất dữ liệu trên Cursor

- RELATIVE {n | @nvar}: Nếu n hoặc @nvar > 0, chuyển xuống n dòng tính từ dòng kề dưới dòng hiện hành. Nếu n or @nvar < 0, Chuyển lên n dòng trước dòng hiện hành. Nếu n or @nvar = 0, trả về dòng hiện hành.
  - cursor_name: Tên cursor đang mở. Nếu tồn tại cursor cục bộ và cursor toàn cục có cùng tên thì tên cursor được sử dụng sẽ là cursor cục bộ nếu không có từ khóa GLOBAL.
- INTO @varname[,...n]: Danh sách biến cục bộ nhận dữ liệu tương ứng từ các cột trên mẫu tin hiện hành, theo thứ tự từ trái sang phải. Số biến phải bằng số cột đã liệt kê trong câu lệnh Select khi tạo Cursor. Kiểu dữ liệu của mỗi biến phải tương thích với kiểu dữ liệu của cột hoặc được hỗ trợ chuyển kiểu ngầm định theo kiểu của cột.
- Kiểm tra kết quả của lệnh FETCH: Sử dụng hàm @@FETCH_STATUS sau lệnh FETCH. Hàm trả về một trong 3 giá trị:
  - 0 Nếu lệnh FETCH chuyển đến 1 mẫu tin trong danh sách.
  - -1 Nếu lệnh FETCH chuyển đến vùng BOF hoặc EOF
  - -2 Nếu chuyển đến 1 dòng đã bị xóa trên Server (Keyset).

```
update hv set malop='L2' where mahv like'HV0[1-3]'
select * from lop
select * from hv
```


R5: Trị giá của một hóa đơn là tổng thành tiền của các chi tiết thuộc hóa đơn đó

```
/*
HOADON(SOHD,NGHD,TRIGIA)
SANPHAM(MASP,GIA,NGNHAP)
CTHD(SOHD,MASP,SL)
*/
CREATE TABLE HOADON (SOHD INT PRIMARY KEY,NGHD SMALLDATETIME,
TRIGIA MONEY)
CREATE TABLE SANPHAM(MASP VARCHAR(4) PRIMARY KEY, GIA MONEY,
NGNHAP SMALLDATETIME)
CREATE TABLE CTHD(SOHD INT, MASP VARCHAR(4), SL INT, CONSTRAINT
PK_CTHD PRIMARY KEY (SOHD,MASP))
```



```
create table lop (malop varchar(4) primary key,siso int)
create table hv(mahv varchar(4) primary key,malop varchar(4) foreign key
references lop(malop))
delete from hv
delete from lop
insert into lop values('L1',4)
insert into lop values('L2',5)

insert into hv values('HV01','L1')
insert into hv values('HV02','L1')
insert into hv values('HV03','L1')
insert into hv values('HV04','L1')
insert into hv values('HV05','L2')
insert into hv values('HV06','L2')
insert into hv values('HV07','L2')
insert into hv values('HV08','L2')
insert into hv values('HV09','L2')

delete from hv where mahv='HV01' or mahv='HV02'
select * from lop
select * from hv
delete from hv where malop='L2'
```


Viết lại trigger upd_hv bằng cách sử dụng Cursor

```
create trigger upd_hv on hv for update as
if update(malop)
begin
     declare @del_malop varchar(20),@ins_malop varchar(20)
     declare hv_cur cursor for
          select deleted.malop,inserted.malop from deleted,
               inserted where deleted.mahv=inserted.mahv
     open hv_cur
     fetch next from hv_cur into @del_malop,@ins_malop
     WHILE @@FETCH_STATUS = 0
     BEGIN
          update lop set siso=siso-1 where malop=@del_malop
          update lop set siso=siso+1 where malop=@ins_malop
          fetch next from hv_cur into @del_malop,@ins_malop
     END
     CLOSE hv_cur
     DEALLOCATE hv_cur
end
```

R5: Trị giá của một hóa đơn là tổng thành tiền các chi tiết thuộc hóa đơn đó

R5	Them	Xoa	Sua
SANPHAM	-	-	+(GIA)
CTHD	+	+	+(MASP,SOHD,SL)
HOADON	+	-	+(TRIGIA)

```
--tiếp theo
WHILE (@@FETCH_STATUS=0)
BEGIN
--cong don tri gia cua tung san pham vao bien TRIGIA
SET @TRIGIA=@TRIGIA + @SL* (SELECT GIA FROM SANPHAM
                                WHERE MASP=@MASP)
FETCH NEXT FROM CUR_CTHD INTO @MASP, @SL
END

CLOSE CUR_CTHD
DEALLOCATE CUR_CTHD
--tien hanh cap nhat lai tri gia HOADON
UPDATE HOADON SET TRIGIA=@TRIGIA WHERE SOHD=@SOHD
END
```

```
CREATE TRIGGER R5_ins_cthd ON CTHD FOR INSERT
AS
BEGIN
DECLARE @SOHD INT, @MASP CHAR(4), @SL INT, @TRIG
--lay thong tin cua CTHD vua moi them vao
SELECT @SOHD=SOHD, @MASP=MASP, @SL=SL FROM INSER
--tinh tri gia cua san pham moi them vao HOADON
SET @TRIGIA=@SL * SELECT GIA FROM SANPHAM WHERE
--khai bao mot CURSOR duyet qua tat ca cac CTHD
HOADON
DECLARE CUR_CTHD CURSOR
FOR
SELECT MASP, SL FROM CTHD WHERE SOHD=@SOHD
OPEN CUR_CTHD
FETCH NEXT FROM CUR_CTHD INTO @MASP, @SL
--xem phần tiếp theo trang sau
```