- tblVATTU: **MaVTu char(4)**, TenVtu nvarchar(30), DVTinh nvarchar(10), Dongia int.

VT01	GẠCH
VT02	
VT03	
VT04	
VT05	

DH - tblDONDH: **SoDH Int**, NgayDH datetime, Manhacc char(4)

1	1/1/2025	CC01
2		CC02
3		CC01

CTDH - tblCTDONDH: **SoDH Int, MaVTu char(4),** SLDat int

1	VT01	100
1	VT02	250
1	VT04	50
2	VT02	250
2	VT01	100
3	VT05	250

NC - tblNHACC: **Manhacc char(4),** Tennhacc nvarchar(30), Diachi nvarchar(30), Dienthoc char(15)

CC01	Lê A
CC02	Phạm B
CC03	Nguyễn C

PN - tblPNHAP: **SoPN char(4),** SoDH Int, Ngaynhap datatime,

PN01	1	6/1/2025
PN02	1	8/1/2025
PN03	1	18/1/2025

CTPN - tblCTPNHAP: **SoPN char(4), MaVTu char(4),** Slnhap int.

PN01	VT01	20
….		
PN02	VT01	30
….		
PN03	VT01	80(<=100-∑(20+30+..))
PN01	VT05	

PX - tblPXUAT: **SoPX char(4),** Ngayxuat datetime.

CTPX - tblCTPXUAT: **SoPX char(4), MaVTu char(4),** SLXuat int.

KHOÁ CHÍNH:

KHOÁ NGOẠI:

Câu lệnh Select:

```
Select * / Danh_sách_các_cột
    From Tên_bảng [Bí_danh]
    [Inner|Left|Right|Full  Join Tên_bảng_qh [Bí_danh] On điều_kiện_quan_hệ,…]
    [Where Điều_kiện ]
    [Group By Danh_sách_cột_nhóm_dl ]
    [Having Điều_kiện_lọc_nhóm ]
    [Order By Tên_cột_sx Desc,…]
    [Computer count/min/max/sum/avg (Tên cột) By tên_cột_nhóm]
```

GIẢI THÍCH:

ᔓ DÒNG 3:

```
[Inner|Left|Right|Full  Join Tên_bảng_qh [Bí_danh] On điều_kiện_quan_hệ,…]
```

- Nếu ta có 2 bảng với dữ liệu như sau:

Bảng 1: tblHOCSINH_Lop10 có dữ liệu như sau:

…..	HoTen	Lop	SBD	…..
	Nguyễn A	10A	1	
	Lê B	10B	2	
	Phạm C	10A	3	
	Văn D	10C	4	
	Thị F	10B	5	

Bảng 2: tblDiem có dữ liệu như sau:

…..	SBD	Điem	….
	3	6	
	4	3	
	5	7	
	6	5	

	7	3	
	8	9	

- Nếu kết nối 2 bảng trên theo `Inner:`

Cú pháp:
```
select * from tblHocSinh HS
    INNER join tblDiem DD on HS.SBD = DD.SBD
```
Ta được kết quả sau:

…	HoTen	Lop	SBD	SBD	Điem	…
	Phạm C	10A	3	3	6	
	Văn D	10C	4	4	3	
	Thị F	10B	5	5	7	

- Nếu kết nối 2 bảng trên theo `Left:`

Cú pháp:
```
select * from tblHocSinh HS
    LEFT join tblDiem DD on HS.SBD = DD.SBD
```
- Ta được kết quả sau:

…	HoTen	Lop	SBD	SBD	Điem	…
	Nguyễn A	10A	1	NULL	NULL	
	Lê B	10B	2	NULL	NULL	
	Phạm C	10A	3	3	6	
	Văn D	10C	4	4	3	
	Thị F	10B	5	5	7	

- Nếu kết nối 2 bảng trên theo `Right:`

Cú pháp:
```
select * from tblHocSinh HS
    RIGHT join tblDiem DD on HS.SBD = DD.SBD
```
- Ta được kết quả sau:

…	HoTen	Lop	SBD	SBD	Điem	…
	Phạm C	10A	3	3	6	
	Văn D	10C	4	4	3	
	Thị F	10B	5	5	7	
	NULL	NULL	NULL	6	5	
	NULL	NULL	NULL	7	3	
	NULL	NULL	NULL	8	9	

- Nếu kết nối 2 bảng trên theo `Full:`

Cú pháp:
```
select * from tblHocSinh HS
   FULL join tblDiem DD on HS.SBD = DD.SBD
```

- Ta được kết quả sau:

…	HoTen	Lop	SBD	SBD	Điem	…
	Nguyễn A	10A	1	NULL	NULL	
	Lê B	10B	2	NULL	NULL	
	Phạm C	10A	3	3	6	
	Văn D	10C	4	4	3	
	Thị F	10B	5	5	7	
	NULL	NULL	NULL	6	5	
	NULL	NULL	NULL	7	3	
	NULL	NULL	NULL	8	9	

ᔓ Kết nối nhiều bảng, thì mỗi bảng thêm vào 1 dòng …JOIN…ON…

Bí danh của tệp

VT

DH

CTDH

Ví dụ: Câu 1: Tạo CSDL QLBH gồm các bảng sau :

- tblVATTU: **MaVTu char(4)**, TenVtu nvarchar(30), DVTinh nvarchar(10), Dongia int.
- tblDONDH: **SoDH Int**, NgayDH datetime, Tennhacc nvarchar(40)
- tblCTDONDH: **SoDH Int, MaVTu char(4),** SLDat int

Bước 1: đọc đề xác định các trường cần qua đó xác định các bảng cần

Bước 2: gõ câu SELECT kết nối các bảng.

1. Viết câu lệnh cho xem Tennhacc, MaVTu, SLDat của các số đơn hàng

```
select * from tbldh DH
    inner join tblctdh CTDH on CTDH.Sodh=DH.Sodh
```

2. Viết câu lệnh cho xem Tennhacc, TenVtu, SLDat của các số đơn hàng

```
select * from tbldh DH
    inner join tblctdh CTDH on CTDH.Sodh=DH.Sodh
```

```
        inner join tblvattu VT on VT.Mavt = CTDH.Mavt
```

- DÒNG 4: [Where Điều_kiện]

  Nếu đề có lọc bớt bản ghi thì ta dùng where, lưu ý: có AND và OR

3. Viết câu lệnh cho xem Tennhacc, MaVTu, SLDat của số đơn hàng là 1

```
select * from tbldh DH
    inner join tblctdh CTDH on CTDH.Sodh=DH.Sodh
    WHERE DH.Sodh=1
```

4. Viết câu lệnh cho xem Tennhacc, TenVtu, SLDat của Tên nhà cung cấp là A:

```
select * from tbldh DH
    inner join tblctdh CTDH on CTDH.Sodh=DH.Sodh
    inner join tblvattu VT on VT.Mavt = CTDH.Mavt
    WHERE Tenncc = N'Sơn Trần'
```

Buổi 3: TRUY VẤN LỒNG

- Cần mở và đóng ngoặc cho câu lệnh truy vấn con

1. Kết quả trả về của truy vấn con là một giá trị: khi đề có chữ LỚN NHẤT hoặc NHỎ NHẤT

Ví dụ: Hiện các đơn đặt hàng trong ngày gần đây nhất

ngày đặt hàng gần đây nhất

Cách làm:

Bước 1: bỏ chữ (LỚN NHẤT hoặc NHỎ NHẤT) đó đi trong đề ra và làm câu SELECT

Bước 2: triển khai câu lệnh SELECT của truy vấn con: có chữ (LỚN NHẤT hoặc NHỎ NHẤT) đó

Bước 3:

- ở cha gõ: WHERE <TÊN TRƯỜNG CHUNG> =

- ở con: bưng câu ở bước 2 vào

DẠNG 2: khi đề có thêm chữ TỔNG LỚN NHẤT

Bước 1: bỏ chữ (LỚN NHẤT hoặc NHỎ NHẤT) đó đi trong đề ra và làm câu SELECT

Bước 2: tạo view cho câu lệnh select ở trên: create view vw1 as….

Bước 3: triển khai câu lệnh SELECT của truy vấn con có chữ (LỚN NHẤT hoặc NHỎ NHẤT) đó

Bước 4:

- ở cha gõ: WHERE <TÊN TRƯỜNG CHUNG> =

- ở con: bưng câu ở bước 3 vào

```
create view vv
as
      select vt.MaVTu,TenVtu,sum(Slnhap) as tsln
      from tblVATTU vt
      inner join tblCTPNHAP ct on ct.MaVTu=vt.MaVTu
      group by vt.MaVTu,TenVtu
      ---
select * from vv
      where tsln = (select max(tsln) from vv)
      --
drop view vv
```

2. Kết quả trả về của truy vấn con là danh sách bảng: khi đề có chữ CHƯA hoặc ĐÃ TỪNG

Ví dụ: Hiện danh sách các nhà cung cấp mà công ty

chưa bao giờ đặt hàng.

Bước 1: tách đề thành 2 phần TRƯỚC và SAU chữ (CHƯA hoặc ĐÃ TỪNG) đó

Bước 2: gõ 2 câu SELECT đó

Bước 3: nối cha với con:

ở cha gõ: WHERE <TÊN TRƯỜNG CHUNG> NOT IN

ở con gõ: select <TÊN TRƯỜNG CHUNG> đó

Buổi 4:

4.1. Biến và các phép toán trên biến

- Trong SQL có 2 loại biến: Biến cục bộ và Biến hệ thống

- Khai báo biến cục bộ
  DECLARE @s_ten_bien Kieu_du_lieu[,…]
- Gán giá trị cho biến
  - Dùng lệnh SELECT cùng với phép gán =
  - Dùng lệnh SET để gán các giá trị cụ thể hoặc biểu thức tính toán hoặc giá trị tính toán từ các biến khác
- Xem giá trị hiện hành của biến
  PRINT @s_ten_bien

Ví dụ: Hiện tổng sldat của tên vật tư A

```
DECLARE @s_sldat int, @s_n int
select  @s_sldat = sum(SLDat)
from tblVATTU vt
          inner join tblCTDONDH ctdh on ctdh.MaVTu=vt.MaVTu
          where TenVtu =N'Gạch'
set @s_n= 100

PRINT N'tổng số lượng đặt:=' + str(@s_sldat,10)
PRINT N'N=' + str(@s_n,10)
```

- Biến hệ thống :
  - Nó cung cấp danh sách các biến hệ thống, bắt đầu bằng @@, giá trị của các biến hệ thống do MS SQL Server cung cấp
  - Người lập trình không can thiệp được vào biến hệ thống của MS SQL Server
  - Người sử dụng thường dùng các giá trị của nó để thực hiện các hoạt động theo ý riêng của mình.
  - Một số biến hệ thống thường dùng
    - @@VERSION
    - @@FETCH_STATUS
      + Đọc dữ liệu trong bảng theo từng dòng cursor
      + Khi đọc mẫu tin thành công thì biến có giá trị = 0
    - @@ROWCOUNT
    - @@SERVERNAME
    - @@ERROR

4.2. Các hàm chuyển đổi kiểu dữ liệu

- CAST (bieuthuc AS kieudulieu [(do_dai)])
- STR (number, length, decimal)
- Convert (Kiểu_dữ_liệu, Biểu_thức[, Định dạng])
  - Kiểu dữ liệu: tên của kiểu mà biểu thức sẽ chuyển đổi sang.
  - Biểu thức: tên cột hoặc biểu thức muốn chuyển kiểu.
  - Định dạng: là một con số chỉ định việc định dạng cho việc chuyển đổi dữ liệu từ dạng ngày sang chuỗi

Định dạng năm (yy)	*Định dạng năm (yyyy)*	*Hiển thị dữ liệu*
1	101	mm/dd/yy
2	102	yy.mm.dd
3	103	dd/mm/yy
4	104	dd.mm.yy
5	105	dd-mm-yy
11	111	yy/mm/dd
	21 hoặc 121	yyyy-mm-dd

5. Cấu trúc điều khiển

5.1 Cấu trúc rẽ nhánh IF … ELSE

IF (bieu_thuc_logic)

Câu lệnh 1 | Khối lệnh 1

[ELSE

Câu lệnh 2 | Khối lệnh 2]

- Lưu ý: Khối lệnh có từ 2 lệnh trở lên, trong trường hợp này bắt buộc phải đặt các lệnh trong cặp BEGIN … END
- Ví dụ:
  - Hiển thị xem có vật tư nào đã bán trong 1 phiếu xuất có số lượng nhiều hơn 5 không?
  - Nếu có thì in ra danh sách đó, ngược lại thông báo không có,
  - chúng ta sử dụng cú pháp IF … ELSE như sau:

```
declare @s_n int
set @s_n =10
if
 ( select count(*)
  from tblCTPXUAT ctpx
  inner join tblVATTU vt on vt.MaVTu=ctpx.MaVTu
  where SLXuat >@s_n )>0

 select *
  from tblCTPXUAT ctpx
  inner join tblVATTU vt on vt.MaVTu=ctpx.MaVTu
  where SLXuat >@s_n

 else
  print 'Không có'
```

-

- Dùng từ khóa EXISTS:
- EXISTS (câu lệnh SELECT…): NẾU câu lệnh SELECT… có bản ghi thì EXISTS cho giá trị YES; Ngược lại cho giá trị NO

```
declare @s_n int
set @s_n =10000
if
 exists( select *
  from tblCTPXUAT ctpx
  inner join tblVATTU vt on vt.MaVTu=ctpx.MaVTu
  where SLXuat >@s_n
  )

select *
  from tblCTPXUAT ctpx
  inner join tblVATTU vt on vt.MaVTu=ctpx.MaVTu
  where SLXuat >@s_n

else
  print 'Không có'
```

-

- Bài tập
    1. Hiển thị xem có vật tư nào đã xuất hàng trong 1 số phiếu xuất có số lượng xuất nhiều hơn N không? Nếu có thì in ra danh sách đó, ngược lại thông báo không có.
    2. Hiển thị xem có vật tư nào có tổng số lượng nhập hàng nhiều hơn N không? Nếu có thì in ra danh sách đó, ngược lại thông báo không có.
    3. Hiển thị xem có nhà cung cấp nào nhập một vật tư có tổng số lượng nhập hàng nhiều hơn N không? Nếu có thì in ra danh sách đó, ngược lại thông báo không có.

Buổi 5:

5.2. Biểu thức Case

- Cú pháp 1:

```
CASE Biểu_thức
  When Giá_trị_1 Then Biểu_thức_1
[ When Giá_trị_2 Then Biểu_thức_2
…]
[ ELSE Biểu_thức_N ]
END
```

Cú pháp 2:

```
CASE
    When Điều_kiện_1 Then Biểu_thức_1
[  When Điều_kiện_2 Then Biểu_thức_2
…]
[  ELSE Biểu_thức_N ]
END
```

Hàm iif (điều_kiện, biểu_thức_1, biểu_thức_2)

Ví dụ:

```
select MaVTu,
      case left(mavtu,2)
            when 'DD' then N'Đầu DVD'
            when 'LO' then N'Loa thùng'
            when 'TV' then N'Tivi'
            else N'Vật tư'
            end as 'LoaiVT'
      ,TenVtu,Dongia,
      case
            when Dongia <50 then N'Giá thấp'
            when Dongia <100 then N'Giá vừa'
            else N'Giá cao'
            end as 'LoaiGia',

iif(Dongia <50,N'Giá thấp',iif(Dongia <100,N'Giá vừa',N'Giá cao'
       from tblVATTU
```

5.3. Cấu trúc lặp WHILE

- Cú pháp

WHILE (biểu_thức_logic)

BEGIN

Các lệnh lặp

END

- Ví dụ

```
DECLARE @sn INT
SET @sn = 100
WHILE (@sn<110)
BEGIN
   PRINT 'So nguyen: ' + CONVERT(char(3),@sn)
   SET @sn = @sn + 1
END
```

Trong While có BREAK: thoát khỏi while; và CONTINUE: tiếp tục…

6. Sử dụng biến kiểu dữ liệu CURSOR

- Khai báo & Định nghĩa
  DECLARE Ten_Cursor CURSOR
  [Kiểu đọc, cập nhật dữ liệu]
  FOR Câu_truy_vấn
- Mở
  OPEN Ten_Cursor
- Sử dụng
  while 0=0
  begin
  ….
  end
- Đóng & Hủy bỏ Cursor
  CLOSE Ten_Cursor

DEALLOCATE Ten_Cursor

Sử dụng

- Đọc và xử lý dữ liệu trong cursor
- Cú pháp

FETCH [NEXT | PRIOR | FIRST | LAST |
ABSOLUTE n | RELATIVE n]

FROM Ten_Cursor
[INTO Danh_sách_biến]

- Dùng biến hệ thống @@FETCH_STATUS để kiểm tra tình trạng đọc dữ liệu thành công hay thất bại, giá trị của nó = 0 là thành công.

Ví dụ: Dùng biến con trỏ CURSOR Hiện SoPX,TenVtu,SLXuat có SLXuat >=A

Bước 1: gõ câu SELECT: Hiện SoPX,TenVtu,SLXuat có SLXuat >=A
Bước 2: Khai báo & Định nghĩa biến kiểu dữ liệu CURSOR

DECLARE Ten_Cursor CURSOR
Keyset
FOR Câu_truy_vấn

Bước 3: Mở Biến_Cursor

OPEN Ten_Cursor

Bước 4: sử dụng: dùng vòng lặp while

```
--Định nghĩa
declare @s_sopx char(4),@s_tenvtu nvarchar(30),@s_slxuat int
DECLARE     Ten_Cursor  CURSOR
    Keyset
    FOR
    select SoPX,TenVtu,SLXuat
    from tblVATTU vt
    inner join tblCTPXUAT ct on ct.MaVTu=vt.MaVTu
    where SLXuat >100
--  Mở
OPEN    Ten_Cursor
--- Sử dụng
while  0=0
    begin
        fetch next from Ten_Cursor INTO @s_sopx,@s_tenvtu,@s_slxuat
        if  @@FETCH_STATUS<>0   break
        print @s_sopx+ @s_tenvtu+ str(@s_slxuat,10)
        end
--  Đóng & Hủy bỏ Cursor
CLOSE   Ten_Cursor
DEALLOCATE  Ten_Cursor
```

?1: là thứ tự các trường trong câu select
?2: biến @s_slxuat từ kiểu int chuyển thành char(10)

Bước 5: Đóng & Hủy bỏ Cursor

CLOSE Ten_Cursor
DEALLOCATE Ten_Cursor

```
--Ví dụ: Xem các mặt hàng có Tên bắt đầu là V.
declare cur_vt cursor
keyset
for
    select * from tblvattu
    where tenvtu like 'S%'
    order by mavtu
open cur_vt
while 0=0
begin
    fetch next from cur_vt INTO ...
    if @@fetch_status<>0 break
    Print...
    end
close cur_vt
deallocate cur_vt
```

Bài tập: Sử dụng biến dữ liệu kiểu CURSOR làm các bài tập sau:

2.1. Hiện thông tin Nhà cung cấp, số đặt hàng, tên vật tư, số lượng đặt hàng.
2.2. Hiện thông tin Nhà cung cấp, số đặt hàng, tên vật tư, số lượng đặt hàng có số lượng đặt hàng >=A.
2.3. Hiện các phiếu nhập, tên vật tư, số lượng nhập theo ngày nhập hàng tăng dần.
2.4. Hiện số lượng đặt hàng của các vật tư có đơn đặt hàng là N.
2.5. Hiện số lượng đặt hàng của các vật tư có số lượng đặt hàng >=N
2.6. Hiện số lượng xuất hàng của các vật tư có phiếu xuất là N.
2.7. Hiện số lượng xuất hàng của các vật tư có số lượng xuất hàng >=N
2.8. Hiện tổng số lượng đặt hàng của các vật tư.
2.9. Hiện các vật tư có tổng lượng đặt hàng >=N.
2.10. Hiện tổng số lượng bán hàng của các vật tư.
2.11. Thống kê số lượng đặt hàng của các vật tư với Nhà cung cấp là A .
2.12. Thống kê số lượng tổng nhập hàng của các vật tư theo Nhà cung cấp.

Ví dụ trên viết đầy đủ

```
--Ví dụ: Xem các mặt hàng có Tên bắt đầu là S.
declare @s_mavtu char(4),@s_tenvtu nchar(30),@s_dvtinh char(10),@s_dongia int
declare cur_vt cursor
keyset
for
    select * from tblvattu
    where tenvtu like 'S%'
    order by mavtu
open cur_vt
while 0=0
begin
    fetch next from cur_vt INTO @s_mavtu,@s_tenvtu,@s_dvtinh,@s_dongia
    if @@fetch_status<>0 break
    Print @s_mavtu+@s_tenvtu+@s_dvtinh+str(@s_dongia,10)
    end
close cur_vt
deallocate cur_vt
```

Thủ tục:

Cú pháp:

```
CREATE PROCEDURE   Ten_Thu_Tuc
     @tên_tham_số  kiểu_dữ_liệu  loại
     AS
     [DECLARE  Bien_cuc_bo]
          Các_lệnh
---Lời gọi

EXEC[UTE]       Ten_thu_tuc giá_trị | @biến  [output]
```

Ví dụ:

```
create proc tong @s_m int, @s_n int output
as
      set @s_m = @s_m+5
      set @s_n = @s_n+5
      print 'Trong thủ tục '+str(@s_m,10)+'   '+str(@s_n,10)

declare @s_a int, @s_b int
set @s_a = 10
set @s_b = 10
      print 'Truoc thủ tục '+str(@s_a,10)+'   '+str(@s_b,10)
exec tong @s_a,@s_b output

      print 'Sau  thủ tục '+str(@s_a,10)+'   '+str(@s_b,10)
```

Các bước tạo thủ tục:

Bước 1: Xác định INPUT và OUTPUT của thủ tục:

Bước 2: gõ câu SELECT

Bước 3: gán vào thủ tục

- Tham số hình thức: copy INPUT và OUTPUT ở trên
- Thân thủ tục: copy đoạn Bước 2 và chỉnh sửa

Ví dụ 1: Xây dựng thủ tục tính tổng trị giá của một sodh

Bước 1: Xác định INPUT và OUTPUT của thủ tục:

INPUT: một sodh @s_ **SoDH int**

OUTPUT: tổng trị giá @s_tongtien int

Bước 2: gõ câu SELECT

```
Select sum(vt.Dongia*ct.SLDat)
from tblVATTU vt
inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
where SoDH=3
```

Bước 3: gán vào thủ tục

```
CREATE PROCEDURE   Ten_Thu_Tuc   @s_SoDH  int,@s_tongtien  int output
     AS
```

```
           Select @s_tongtien = sum(vt.Dongia*ct.SLDat)
           from tblVATTU vt
           inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
           where SoDH=@s_SoDH

---lời gọi
declare    @ss_tongtien int
EXEC         Ten_thu_tuc 3,  @ss_tongtien  output
print      @ss_tongtien
```

BÀI TẬP:

1.2. Tạo thủ tục tính tổng sô lượng nhập hàng của một vật tư
1.3. Tạo thủ tục tính tổng sô lượng đặt hàng của một nhà cung cấp
1.4. Tạo thủ tục tính tổng tiền của 1 phiếu nhập hàng.
1.5. Tạo thủ tục tính tổng tiền của 1 phiếu xuất hàng.
1.6. Xây dựng thủ tục trả về danh sách các tên vật tư có số lượng đặt hàng nhiều nhất trong năm tháng nào đó.
1.7. Hiện vật tư nào có tổng số lượng đặt hàng lớn nhất.

- Sửa đổi thủ tục
    - ALTER PROCEDURE Ten_thu_tuc
- Xóa thủ tục
    - DROP PROCEDURE Ten_thu_tuc

Ví dụ 2: Hiện Tên vật tư có tổng số lượng nhập hàng lớn nhất.

Bước 1:

INPUT: không

OUTPUT: Tên vật tư @s_TenVtu nvarchar(30)

Bước 2: GÕ CÂU SELECT

```
select vt.MaVTu,vt.TenVtu,sum(Slnhap)as 'tsln'  INTO #BTAM
from tblVATTU vt
inner join tblCTPNHAP  ct on ct.MaVTu=vt.MaVTu
group by vt.MaVTu,vt.TenVtu
--
select TenVtu from #BTAM
where tsln = ( select max(tsln) from #BTAM)
--
drop TABLE #BTAM
```

Bước 3: gán vào thủ tục

```
CREATE PROCEDURE   Ten_Thu_Tuc2  @s_tenvatu  nvarchar(30) output
AS

    select vt.MaVTu,vt.TenVtu,sum(Slnhap)as 'tsln'  INTO #BTAM
    from tblVATTU vt
    inner join tblCTPNHAP  ct on ct.MaVTu=vt.MaVTu
    group by vt.MaVTu,vt.TenVtu
    --
    select @s_tenvatu = TenVtu from #BTAM
    where tsln = ( select max(tsln) from #BTAM)
    --
    drop TABLE #BTAM
--loi goi
declare @s_tvt nvarchar(30)
EXEC       Ten_thu_tuc2 @s_tvt output
print @s_tvt
```

- Bài tập: Viết thủ tục sinh ra mã vật tư tự động theo quy tắc:
  - Mã vật tư có dạng: VT01
  - 'VT' : quy định (luôn có)
  - 01 : là số
  - VD:
    - Hiện tại Vật tư có mã cao nhất là VT04
    - Thì sinh mã mới là VT05

Bước 1:
Input: Không có
Output: @s_Mavtu char(10)
Bước 2: gõ đoạn select

```
declare @s_s int, @s_mavtu char(10)
select @s_s = max(cast(substring(MaVTu,3,2) as int))+1 from tblVATTU
set @s_mavtu = '000'+ ltrim(str(@s_s,10))
set @s_mavtu = 'VT'+substring(@s_mavtu,len(@s_mavtu)-1,2)
print @s_mavtu
```

Bước 3:

```
CREATE PROCEDURE   tt_sinh_ma
    @s_MaVTu  char(10) output
    AS

    declare @s_s int
    select @s_s = max(cast(substring(MaVTu,3,2) as int))+1 from tblVATTU
    set @s_mavtu = '000'+ ltrim(str(@s_s,10))
    set @s_mavtu = 'VT'+substring(@s_mavtu,len(@s_mavtu)-1,2)

---Lời gọi
declare @ss_MaVTu  char(10)
EXEC       tt_sinh_ma @ss_MaVTu   output
print @ss_mavtu
```

BÀI TẬP:

1.1 Viết thủ tục sinh ra mã nhà cung cấp tự động cho bảng tblNhaCC theo quy tắc trên.

1. Sử dụng RETURN trong thủ tục

- RETURN dùng để thoát ra khỏi thủ tục trong trường hợp dữ liệu không hợp lệ.
- Nó cho phép trả về 1 số nguyên (giống hàm)

- RETURN không có giá trị chỉ định ngầm định 0
- Trong trường hợp thủ tục dùng Return, khi gọi thủ tục ta sử dụng 1 biến cục bộ để đón nhận giá trị trả về của thủ tục đó.

Exec @Biến = Tên_thủ_tục[các_tham_số]

Ví dụ: tính tổng tiền của 1 SoDH,nếu SoDH không tồn tại cho dòng thông báo.

```
alter PROCEDURE   Ten_Thu_Tuc   @s_SoDH  int,@s_tongtien  int output
     AS
if not  exists(select * from tblCTDONDH where sodh =@s_SoDH )
     RETURN 1
else

          Select @s_tongtien = sum(vt.Dongia*ct.SLDat)
          from tblVATTU vt
          inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
          where SoDH=@s_SoDH

---lời gọi
declare @ss_tongtien int, @s_b int
EXEC       @s_b=  Ten_thu_tuc 21,  @ss_tongtien  output
if @s_b=1
     print 'Không có sodh này'
     else
print 'Toongrtien ='+ str(@ss_tongtien,10)
```

BÀI TẬP:

1.2. Tạo thủ tục tính tổng tiền của 1 phiếu nhập hàng. nếu phiếu nhập hàng không có trong bảng tblpnhap, cho dòng thông báo ;

1.3. Tạo thủ tục tính tổng tiền của 1 phiếu xuất hàng. nếu phiếu xuất hàng không có trong bảng tblpxuat, cho dòng thông báo ;

1.4. Tạo thủ tục tính tổng sô lượng nhập hàng của một vật tư, nếu vật tư không có trong bảng tblVattu, cho dòng thông báo 1; nếu tên vật tư chưa đặt hàng thì cho dòng thông báo 2.

1.5. Tạo thủ tục tính tổng sô lượng đặt hàng của một nhà cung cấp; nếu nhà cung cấp không có trong bảng tblNhacc, cho dòng thông báo 1; nếu nhà cung cấp chưa đặt hàng thì cho dòng thông báo 2.

1.6.

```
alter PROCEDURE   Ten_Thu_Tuc       @s_TenVTu  nvarchar(30),@s_tsln  int output
 AS
if not  exists(select * from tblVATTU Vt where VT.TenVtu=@s_TenVTu )
 RETURN 1
else
        if not exists
         (
            select * from tblVATTU VT
            inner join tblCTDONDH CTDH on CTDh.MaVTu=VT.MaVTu
            where VT.TenVtu=@s_TenVTu
         )
          return 2
         else
         select @s_tsln=sum(CTPN.Slnhap) from tblVATTU VT
        inner join tblCTPNHAP CTPN on CTPN.MaVTu=VT.MaVTu
        where VT.TenVtu=@s_TenVTu

---lời gọi
declare @ss_tsln int, @s_b int
EXEC          @s_b=  Ten_thu_tuc N'Đầu DVD Hitachi',  @ss_tsln  output
if @s_b=1
 print 'Không có sodh này'
 else
 if @s_b=2 print 'Ten vat tu nay chua dat hang'
 else
print 'Tong so luong nhap ='+ str(@ss_tsln,10)
```

2. Sử dụng bảng tạm trong thủ tục

- Cú pháp:

  Select Ds_cột INTO #Tên_btạm
  From tên_bảng_dữliệu
- Nếu sử dụng bảng tạm ta nhớ phải xóa nó bằng DROP TABLE trước khi kết thúc thủ tục

BÀI TẬP:

1.6. Xây dựng thủ tục trả về danh sách các tên vật tư có số lượng đặt hàng nhiều nhất trong năm tháng nào đó.

1.7. Hiện vật tư nào có tổng số lượng đặt hàng lớn nhất.

set **nocount on**

Buổi 8: Tham số kiểu CURSOR bên trong thủ tục

OUTPUT của thủ tục:

- Là 1 giá trị: (các bài ta làm từ trước tới nay), ta có 1 biến nhớ để lưu nó về (VD: @s_tongtien,@s_tsld,…)
- Là 1 ds bảng:
  + nếu thủ tục chỉ yêu cầu cho xem: không làm gì cả (Ví dụ dưới)
  + nếu thủ tục chỉ yêu cầu lưu đầu ra (lưu danh sách): dùng CURSOR

1. Viết thủ tục cho tên vật tư, sldat của 1 số đặt hàng

Bước 1:

INPUT: 1 số đặt hàng @s_SoDH int

OUTPUT: tên vật tư, sldat: có dạng ds bảng

Bước 2: gõ đoạn select

```
select vt.MaVTu,vt.TenVtu,SLDat
    from tblVATTU vt
    inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
    where SoDH=2
```

Bước : gán vào thủ tục:

```
create proc tt1 @s_SoDH int
as
        select vt.MaVTu,vt.TenVtu,SLDat
        from tblVATTU vt
        inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
        where SoDH=@s_SoDH
---lời gọi
exec tt1 3
```

NHẮC LẠI SỬ DỤNG BIẾN CURSOR:

2.3. Sử dụng biến kiểu dữ liệu CURSOR

- Khai báo & Định nghĩa

  DECLARE Ten_Cursor

  Set Ten_Cursor = CURSOR

[Kiểu đọc, cập nhật dữ liệu]
FOR Câu_truy_vấn

- Mở

  OPEN Ten_Cursor
- Sử dụng

  while 0=0
  begin
  ....
  end
- Đóng & Hủy bỏ Cursor

  CLOSE Ten_Cursor
  DEALLOCATE Ten_Cursor

THAM SỐ KIỂU CURSOR TRONG THỦ TỤC

- Dùng trong trường hợp tham số trả về là các dòng giá trị
- Các hành động liên quan đến các biến dữ liệu kiểu CURSOR được chia thành 2 phần: bên trong thủ tục và bên ngoài thủ tục.
- Bên trong khai báo thủ tục gồm: KHAIBAO, định nghĩa biến kiểu cursor và mở Cursor.
- Bên ngoài thủ tục gồm: đọc từng dòng dữ liệu trong Cursor và đóng nó.

Ví dụ: Viết thủ tục CURSOR cho tên vật tư, só lượng đặt của 1 số đơn hàng
Bước 1:
INPUT: 1 số đặt hàng @s_SoDH int
OUTPUT: tên vật tư, sldat: có dạng ds bảng: @s_cur CURSOR VARYING

```
alter proc tt1 @s_SoDH int,@s_cur CURSOR VARYING OUTPUT
as
     set @s_cur = CURSOR
          KEYSET
          FOR
          select vt.MaVTu,vt.TenVtu,SLDat
               from tblVATTU vt
               inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
               where SoDH=@s_SoDH
     open @s_cur
```

```
---lời gọi
declare @ss_cur CURSOR
declare @s_MaVTu char(4),@s_TenVtu nvarchar(30),@s_SLDat int
exec tt1 3,@ss_cur output
while 0=0
     begin
     fetch next from @ss_cur into @s_MaVTu,@s_TenVtu,@s_SLDat
     if @@FETCH_STATUS <>0 break
     print @s_MaVTu+@s_TenVtu+str(@s_SLDat,10)
     end
close @ss_cur
deallocate @ss_cur
```

Bài Tập:
1. Viết thủ tục CURSOR cho xem MaVTU, TenVTu, SLXuat của số phiếu xuất N, với N là đầu vào của thủ tục này.
2. Viết thủ tục CURSOR cho xem MaVTU, TenVTu, tổng SLĐặt hàng của nhà cung cấp A, với A là đầu vào của thủ tục này
3. Viết thủ tục CURSOR tìm số phiếu xuất bán trong năm tháng nào đó (với đầu vào là năm tháng này) để xóa tất cả những phiếu xuất này trong CSDL (tức là phải xóa trong bảng tblCTPXuat, và bảng tblPXuat)
4. Viết thủ tục CURSOR tìm số phiếu nhập có nhập mặt hàng tên A (với đầu vào là tên mặt hàng này) để xóa tất cả những phiếu nhập này trong CSDL (tức là phải xóa trong bảng tblCTPNhap, và bảng tblPNhap)

```
create proc tt1 @s_sopn char(4)
as
delete from tblCTPNHAP where sopn= @s_sopn
delete from tblPNHAP where SoPN=@s_sopn
---lời gọi
exec tt1 'PN02'
select * from tblPNHAP
```

```
-----xóa nhà cc
        delete from tblCTPNHAP where SoPN in (select
SoPN from tblPNHAP where sodh in (select SoDH from
tblDONDH where Manhacc=@manhacc) )

        delete from tblPNHAP where sodh in (select
SoDH from tblDONDH where Manhacc=@manhacc)

        delete from tblCTDONDH where sodh in (select
    SoDH from tblDONDH where Manhacc=@manhacc)

        delete from tblDONDH where Manhacc=@manhacc
        delete from tblNHACC where Manhacc=@manhacc
```

BUỔI 9

1. Thủ tục cập nhật bảng dữ liệu: cập nhật trên 1 bảng

Viết Thủ tục cập nhật cho bảng dữ liệu sau tblCTDONDH: **SoDH Int, MaVTu char(4),** SLDat int. Với yêu cầu phải thực hiện kiểm tra tính hợp lệ của dữ liệu được bổ sung, với nguyên tắc là **không được trùng khóa chính** và đảm bảo toàn vẹn dữ liệu tham chiếu đến các bảng có liên quan.

- **khóa chính: SoDH Int, MaVTu char(4) không trùng lặp**
- khóa ngoại: **SoDH Int tham chiếu đến bảng tblDonDH, MaVTu char(4) tham chiếu đến bảng tblVattu**
- Miền giá trị: 10 <= SLDat <=100

Bước 1:

Input: @s_**SoDH Int, @s_MaVTu char(4),** @s_SLDat int

Output: cập nhật được 1 bộ giá trị cho bảng

```
alter proc tt1 @s_SoDH Int, @s_MaVTu char(4), @s_SLDat int
as
--  khóa chính
if exists (select * from tblCTDONDH
                  where SoDH=@s_SoDH and MaVTu=@s_MaVTu)
         begin
         print'Trùng khóa chính'
         return
         end
-- -khóa ngoại SoDH
if not exists (select * from tblDONDH
                       where SoDH =@s_SoDH)
         begin
         print'lỗi khóa ngoại SoDH'
         return
         end
-- -khóa ngoại MaVtu
if not exists (select * from tblVATTU
                       where MaVTu =@s_MaVTu)
         begin
         print'lỗi khóa ngoại MaVtu'
         return
         end
--- Miền giá trị:
if (@s_SLDat<10) or (@s_SLDat>100)
         begin
         print'lỗi Miền giá trị'
         return
         end

insert into tblCTDONDH values(@s_SoDH , @s_MaVTu , @s_SLDat)
```

```
---lời gọi '2 VT05'sn
exec tt1 1,'VT04',200
exec tt1 25,'VT04',200
exec tt1 2,'VT14',200
exec tt1 2,'VT05',2
exec tt1 2,'VT05',200
```

```
exec tt1 2,'VT05',20
```

2. Thủ tục hiện thị bảng dữ liệu

Ví dụ: Viết thủ tục cho tên vật tư, sldat của 1 số đặt hàng, nếu gọi thủ tục mà không truyền tham số thì xem như hiện thị toàn bộ các Sodh có trong bảng tblDondh

```
alter proc tt2 @s_SoDH int = NULL
as
if @s_SoDH is null
        select sodh,vt.MaVTu,vt.TenVtu,SLDat
        from tblVATTU vt
        inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
else
        select vt.MaVTu,vt.TenVtu,SLDat
        from tblVATTU vt
        inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
        where SoDH=@s_SoDH
```

```
---lời gọi
exec tt2 1
exec tt2 2

exec tt2
```

BÀI TẬP: ViẾT Thủ tục cập nhật cho bảng dữ liệu sau. Với yêu cầu phải thực hiện kiểm tra tính hợp lệ của dữ liệu được bổ sung, với nguyên tắc là **không được trùng khóa chính** và đảm bảo toàn vẹn dữ liệu tham chiếu đến các bảng có liên quan.

- tblDONDH: **SoDH Int**, NgayDH datetime, Manhacc char(4)

- tblPNHAP: **SoPN char(4),** SoDH Int, Ngaynhap datetime,

- tblCTPNHAP: **SoPN char(4), MaVTu char(4),** Slnhap int.

- tblPXUAT: **SoPX char(4),** Ngayxuat datetime.

- tblCTPXUAT: **SoPX char(4), MaVTu char(4),** SLXuat int.

Bài tập thủ tục hiện thị:

1. Xây dựng thủ tục hiện thị dữ liệu cho báo cáo xuất hàng: tên vật tư, số lượng xuất, đơn giá, tổng tiền. Tham số vào soPX, nếu gọi thủ tục mà không truyền tham số thì xem như hiện thị toàn bộ các SoPX có trong bảng tblpxuat.

2. Xây dựng thủ tục hiện thị dữ liệu cho báo cáo gồm Họ tên nhàn cung cấp, tên vật tư, số lượng đặt, tổng tiền. Tham số vào tên nhà cung cấp, nếu gọi thủ tục mà không truyền tham số thì xem như hiện thị toàn bộ các nhà cc.

Buổi 10: HÀM

OUTPUT của hàm: tương tự thủ tục có 2 loại

- Là 1 giá trị:
- Là 1 ds bảng:

1. Hàm trả về giá trị vô hướng

- Cú pháp

```
Create function Tên_hàm (tham_số_vào)
        returns kiểu_dl_trả_về
As
Begin
Declare  @Biến_trả_về  đúng_kiểu_dl_trả_về
     Các_câu_lệnh
Return @Biến_trả_về
End
```

- Lời gọi hàm:

  Print dbo.Tên_hàm (các_tham_số_thực_sự)

Các bước tạo hàm:

Bước 1: Xác định INPUT và OUTPUT của hàm

Bước 2: gõ câu SELECT

Bước 3: gán vào hàm

- Tham số hình thức: copy INPUT ở trên và OUTPUT trong `returns`
- Thân HÀM: copy đoạn Bước 2 và chỉnh sửa

Ví dụ 1: Viết hàm sinh mã VT tự động.(Hiện tại mã vật tư lớn nhất trong tblvattu là vt07, đưa ra mã vt08)

Bước 1: Xác định INPUT và OUTPUT

Input:

Output: @s_MaVtu char(10)

Bước 2: gõ câu SELECT

```
declare @s_n int, @s_MaVtu char(10)
select @s_n=max(cast(substring(MaVTu,3,2) as int))+1 from tblVATTU
set @s_MaVtu ='000'+ ltrim(str(@s_n,10))
set @s_MaVtu= 'VT'+substring(@s_mavtu,len(@s_mavtu)-1,2)
print @s_Mavtu
```

Bước 3: gán vào hàm

```
Create function ham1 ()
        returns char(10)
As
Begin
    declare @s_n int, @s_MaVtu char(10)
      select @s_n=max(cast(substring(MaVTu,3,2) as int))+1 from tblVATTU
```

```
        set @s_MaVtu ='000'+ ltrim(str(@s_n,10))
        set @s_MaVtu= 'VT'+substring(@s_mavtu,len(@s_mavtu)-1,2)

Return @s_MaVtu
End
---•  Lời gọi hàm:
Print dbo.ham1 ()
```

- Bài tập:

1. Viết hàm tham số vào là mã hàng hoá trả về đơn giá của hàng hoá này.

2. Viết hàm tham số vào là tên hàng hoá trả về đơn giá của hàng hoá này.

3. Viết hàm tham số vào là nhà cung cấp trả về tổng tiền phải trả cho nhà cc này.

4. Viết hàm tham số vào là tên nhà cung cấp và tên hàng hoá trả về tổng tiền phải trả cho nhà cc này với tên hàng hoá trên.

5. Viết hàm tham số vào là số đặt hàng trả về tổng tiền của số đặt hàng này.

Viết hàm tham số vào là số phiếu xuất trả về tổng tiền của nó.

7. Viết hàm tham số vào là tên hàng hoá trả về tổng số lượng nhập hàng của hàng hoá này.

8. Viết hàm tham số vào là tên hàng hoá trả về tổng số lượng xuất hàng của hàng hoá này.

9. Viết hàm tham số vào là tên hàng hoá trả về tổng tiền nhập hàng của hàng hoá này.

10. Viết hàm tham số vào là tên hàng hoá trả về tổng tiền bán hàng của hàng hoá này

Ví dụ: 4. Viết hàm tham số vào là tên nhà cung cấp và tên hàng hoá trả về tổng tiền phải trả cho nhà cc này với tên hàng hoá trên.

Bước 1:

Input: tên nhà cung cấp và tên hàng hoá @s_tennhacc nvarchar(30),@s_tenvtu nvarchar(30)

Output: tổng tiền @s_tongtien int

Bước 2:

Buổi 11: HÀM trả về dạng bảng

2.5.2. Hàm trả về dạng bảng Inline Table

- Cú pháp:

Create function tên_hàm(tên_các_tham_số)

Returns Table

As

Return (Câu lệnh Select)

- Lời gọi:

Select * from dbo.tên_hàm(tên_tham_số)

Ví dụ: Viết hàm cho mã vật tư, tên vật tư, sldat của 1 số đặt hàng

Bước 1:

INPUT: 1 số đặt hàng @s_SoDH int

OUTPUT: mã vật tư, tên vật tư, sldat: có dạng ds bảng

Bước 2: gõ đoạn select

select vt.MaVTu,vt.TenVtu,SLDat

from tblVATTU vt

inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu

where SoDH=2

Bước : gán vào hàm:

```
Create function ham20(@s_SoDH int)
Returns Table
As
Return (          select vt.MaVTu,vt.TenVtu,SLDat
            from tblVATTU vt
            inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
            where SoDH=@s_SoDH
            )


--•    Lời gọi:
      Select * from dbo.ham20(3)
```

Bài tập hàm dạng bảng Inline Table:

1. Viết hàm hiển thị các nhà cung cấp đã cc hàng hoá A
2. Input:
3. Output:

2. Tạo hàm tham số vào là nhàcc, ra là tên các hàng hoá, đơn giá, Tổng số lượng đặt hàng, Tiền đã cung cấp

3. Tạo hàm hiển thị các tên hàng hoá, số lượng đặt mua, đơn giá, Tiền của số đặt hàng s.

4. Tạo hàm hiển thị các tên hàng hoá, số lượng bán hàng, đơn giá, Tiền của số phiếu xuất s.

2.5.4. Hàm trả về dạng bảng Multi Statement

- Nó cho phép thực hiện các câu lệnh select phức tạp,các lệnh update, insert into,...Hàm dạng này luôn trả về một biến table
- Cú pháp:

```
Create Function Tên_hàm (DS_tham_số)
Returns @ Tên_Table TABLE
( Tên_cột_1 kiểu _dl,
  Tên_cột _2 kiểu dl,...)
As
Begin
  Các _câu_ lệnh
  Return
End
```

Ví dụ: Tạo hàm đầu vào là sodh, ra là table chứa tên hàng hoá, số lượng đặt, đơn giá, tổng tiền của sodh này

Input: sodh @s_sodh int

Ouput: table chứa tên hàng hoá, số lượng đặt, đơn giá, tổng tiền

```
Create Function h22 (@s_sodh int)
Returns @table1 TABLE
( tenvtu nvarchar(50),
  sldat int,
  dongia money,
  tongtien money)
As
Begin
   insert into @table1 (tenvtu,sldat,dongia)
   select vt.TenVtu,SLDat,Dongia
     from tblVATTU vt
     inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
     where sodh=@s_sodh
     --
     update @table1
     set tongtien= sldat* dongia
   Return
End
```

```
--tên hàng hoá, số lượng đặt, đơn giá, tổng tiền
----lời gọi
select * from dbo.h22(1)
select * from dbo.h22(2)
```

VD trên: thêm cột Giảm giá= nếu tổng tiền <= A, để nguyên không giảm, nếu từ A đến B giảm 5%, lớn hơn B giảm =10%, Cột tiền phải trả = tổng tiền-giảm.

```
alter Function h22 (@s_sodh int)
Returns @table1 TABLE
( tenvtu nvarchar(50),
  sldat int,
  dongia money,
  tongtien money,
  giamgia money,
  tpt money)
As
Begin
   insert into @table1 (tenvtu,sldat,dongia,tongtien)
   select vt.TenVtu,SLDat,Dongia,SLDat*Dongia
     from tblVATTU vt
     inner join tblCTDONDH ct on ct.MaVTu=vt.MaVTu
     where sodh=@s_sodh
     --
     update @table1
     set giamgia =
               CASE
                    When tongtien>=10000 Then 0.1*tongtien
                    When tongtien>=5000 Then 0.05*tongtien
                    else 0
                    END

     update @table1
     set tpt = tongtien - giamgia

   Return
End


--tên hàng hoá, số lượng đặt, đơn giá, tổng tiền
```

```
----lòi gọi
select * from dbo.h22(1)
select * from dbo.h22(2)
```

Bài tập: Tạo hàm tạo table gồm: tên nhà cc, tổng tiền,cột giảm giá: giảm 10% nếu tổng tiền > A, nếu từ A đến B giảm 5%, ít hơn B giảm =0, Cột tiền phải trả = tổng tiền-giảm.

BUỔI 12: Trigger:

- Cơ chế hoạt động của trigger: Khi thêm mới mẫu tin vào table nó sẽ kích hoạt 1 table lưu trữ dữ liệu mẫu tin mới này với tên INSERTED. Khi xóa mẫu tin nó sẽ kích hoạt table với tên DELETED. Khi sửa là phối hợp 2 lệnh delete và insert, do thế bảng DELETED chứa dòng thông tin cũ, bảng INSERTED chứa dòng dữ liệu mới.
- Trigger có các loại sau: INSTEAD OF và AFTER (FOR).
    - INSTEAD OF: bỏ qua hành động kích hoạt trigger (thao tác insert, delete , update khi gọi sẽ không được thực hiện trên table), thay vào đó nó sẽ thực hiện các câu lệnh bên trong trigger. Nó bỏ qua hành động tác động đến CSDL nhưng việc lưu trữ dữ liệu trên các bảng Inserted và Deleted vẫn được thực hiện. Instead of có thể được định nghĩa trên table hoặc view.
    - AFTER(hoặc FOR): có vai trò bổ sung thêm hành động kích hoạt trigger. Các câu lệnh bên trong trigger chỉ được thi hành sau khi các hành động kích hoạt đã được thực hiện rồi. Trigger loại After chỉ được định nghĩa duy nhất trên Table.
- Trên 1 Table, nếu ta định nghĩa cả 2 loại trigger và các constraint, thứ tự thi hành sẽ là trigger INSTEAD OF, các constraint được xử lý và sau cùng là trigger AFTER.
- Nếu các constraint bị vi phạm (tính toàn vẹn dữ liệu), các hành động của trigger Instead of sẽ được quay lui, trigger After sẽ không được thi hành.

Ví dụ 1: viết trigger khi thêm hoặc sửa bản ghi trong bảng tblnhacc phải thõa mãn độ dài trường dienthoai LÀ 10 KÝ TỰ

```
alter trigger them_nhacc
        on tblnhacc
        after insert,update
        as
        if      (select len(Dienthoai) from Inserted)<>10
                begin
                raiserror ('???TBL: trung khoa chinh',16,1)
                rollback tran
                RETURN

                end


Lời gọi:
        insert into tblnhacc
        Values('CC99','XNK ggh','gfhf','096432')

update tblNHACC
set Dienthoai='1234567890'
where Manhacc='cc01'
--
select * from tblNHACC
```

- Ví dụ 2: Sửa thông tin bảng tbldondh thõa mãn:
  - Không cho phép sửa dữ liệu cột sodh
  - Sửa manhacc phải tồn tại trong tblnhacc
  - Cột ngaydh phải trước ngày nhập hàng trong tblpnhap.

- alter table tblCTDONDH nocheck constraint all

```
--Sửa thông tin bảng tbldondh
alter trigger sua_tbldondh
    on tbldondh
    after update
    as
--  Không cho phép sửa dữ liệu cột sodh

    if update(sodh)
            begin
            raiserror ('??? lỗi 1 ửa dữ liệu cột sodh',16,1)
            rollback tran
            RETURN

            end
---2Sửa manhacc phải tồn tại trong tblnhacc
    if not exists(
                        select * from inserted  ins,tblNHACC nc
                        where nc.Manhacc=ins.Manhacc
                        )
            begin
            raiserror ('??? manhacc phải tồn tại trong tblnhacc',16,1)
            rollback tran
            RETURN

            end
    ---Cột ngaydh phải trước ngày nhập hàng trong tblpnhap.
    declare @s_sodh int,@s_ngaynhap datetime
    select @s_sodh=SoDH from inserted
    select @s_ngaynhap = min(Ngaynhap)  from tblPNHAP where SoDH = @s_sodh
    if      (select NgayDH from inserted ) > @s_ngaynhap
            begin
            raiserror ('??? lỗi 3 ngay đặt phải trước ngà nhap',16,1)
            rollback tran
            RETURN

            end
Lời gọi:

update tblDONDH
set NgayDH= '2012-07-07'
where SoDH=1
--
```

BÀI TẬP:

Câu 1: Viết trigger để khi xóa một bản ghi trong bảng **tblPNhap** thì phải xóa luôn SoPN đó có trong bảng tblCTPNhap.

Câu 2: Viết trigger để khi thêm một bản ghi mới vào bảng tblNhacc thì thỏa các yêu cầu sau:

- MaNhacc phải bắt đầu bằng 2 ký tự 'CC'.
- DiaChi chỉ nhận 2 giá trị 'Nghệ An' và 'Thanh Hóa'.

Câu 3: Viết trigger để khi sửa một bản ghi trong bảng **tblCTDONDH** thì thỏa các yêu cầu sau:

- Không cho phép sửa cột SoDH.

- 20 ≤ SLDat ≤ 50

Câu 4: Viết trigger để khi xóa một bản ghi trong bảng **tblPNHap** thõa mãn: nếu SoPN đó đã có trong bảng **tblCTPNhap** thì không được xóa.

Câu 5:- Hãy thêm cột tổng tiền kiểu money cho bảng tblCTPXuat.

Sau đó viết trigger để sửa một bản ghi trong bảng **tblCTPXuat** thì thỏa các yêu cầu sau:

- Không được sửa cột PXuat.
- Nếu sửa cột SLXuat thì phải sửa thêm cột TongTien = SLXuat * DonGia. (DonGia trong bảng tblVattu)

Câu 1:

```
alter trigger abc1 on tblPNhap  INSTEAD OF  delete
as
begin
       delete from tblCTPNHAP where SoPN=(select sopn from deleted)
       delete from tblPNHAP where sopn=(select sopn from deleted)
end
delete from tblPNHAP where SoPN='pn02'

select * from tblPNHAP
select * from tblCTPNHAP
```

Câu 2:

```
-- Viết trigger để khi thêm một bản ghi mới vào bảng tblNhacc
--thì thỏa các yêu cầu sau:

alter trigger axtt
on tblNhacc
for insert
as
-- MaNhacc phải bắt đầu bằng 2 ký tự 'CC'.
if (select left(manhacc,2) from inserted)<>'CC'

              begin
              raiserror ('??? lỗi MaNhacc phải bắt đầu bằng 2 ký tự 'CC'',16,1)
              rollback tran
              RETURN
              end
-- DiaChi chỉ nhận 2 giá trị 'Nghệ An' và 'Thanh Hóa'.
if not exists (select * from inserted
          where (Diachi  = N'Thanh Hóa') or (Diachi  = N'Nghệ An'))
              begin
              raiserror ('??? lỗi DiaChi chỉ nhận 2 giá trị 'Nghệ An' và 'Thanh
Hóa'',16,1)
              rollback tran
              RETURN
              end
select * from tblNHACC
-- lời gọi
```

```
insert tblNHACC values('cc16','hjgj',N'Nghệ An','096')
```

Câu 3:

```
Create trigger ten
On  tblCTDONDH
AFTER  update
As
If update(SoDH)
            begin
            raiserror ('???TBL: lỗi 1',16,1)
            rollback tran
            RETURN

            end

If ((select sldat from inserted) <20) or
                  ((select sldat from inserted) >50)

            begin
            raiserror ('???Tlỗi 2',16,1)
            rollback tran
            RETURN

            end

---lời gọi
update  tblCTDONDH
set   sldat =10
where  SoDH= 1 and mavtu ='vt01'
---
select * from tblCTDONDH
```

Câu 4:

```
--Viết trigger để khi xóa một bản ghi trong bảng tblPNHap thõa
mãn:
--nếu SoPN đó
--đã có trong bảng tblCTPNhap thì không được xóa.
alter trigger axtt1
on tblPNHap
INSTEAD OF  delete
as
--
if exists( select * from deleted ii, tblctpnhap ct
                        where ii.SoPN=ct.SoPN)

            begin
            raiserror ('??? không xóa',16,1)
            rollback tran
            RETURN
            end
      else
      delete from tblPNHAP where sopn=(select sopn from deleted)
            --
delete from tblPNHAP where sopn='pn36'
delete from tblPNHAP where sopn='pn20'
```

Câu 5:

```
--viết trigger để sửa một bản ghi trong bảng tblCTPXuat thì thỏa các yêu cầu sau:
--- Không được sửa cột PXuat.
--- Nếu sửa cột SLXuat thì phải sửa thêm cột TongTien = SLXuat * DonGia.  (DonGia
trong bảng tblVattu)
alter trigger gfg
on tblCTPXuat
for update
as
--      Không cho phép sửa dữ liệu cột PXuat

        if update(sopx)
                begin
                raiserror ('??? lỗi 1 ửa dữ liệu cột PXuat',16,1)
                rollback tran
                RETURN
                end
        if update(slxuat)
                begin
                declare @s_n money

                select @s_n = SLXuat*Dongia from inserted ik
                        inner join tblVATTU vt on vt.MaVTu=ik.MaVTu
                update tblCTPXuat
                                set Tongtien=@s_n
                                where sopx=(select sopx from inserted)
                                        and MaVTu=(select MaVTu from inserted)
                end
---lời gọi
update tblCTPXuat
set SLXuat=20
where sopx='px02' and MaVTu='vt01'

--
select * from tblCTPXUAT
```