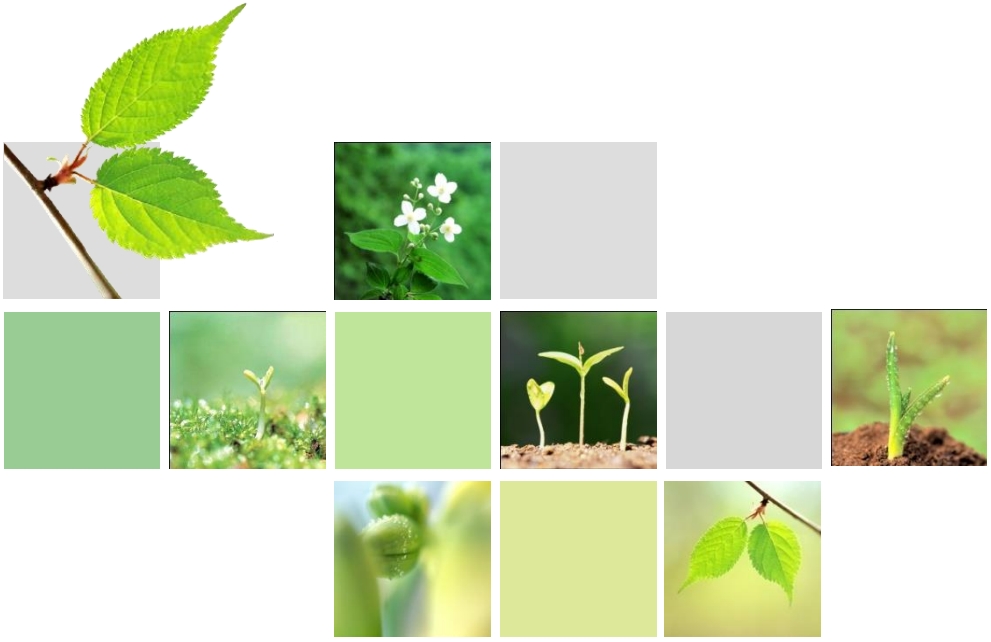
Chương 7

Giao Tác



Định nghĩa

Giao tác

- Đơn Hàng
- Chi tiết_2 ĐH
- Chi tiết_1 ĐH
- Cập nhật SL

⇨ **Giao tác là chuỗi các hành động tương tác trên CSDL**

Khai báo giao tác

Giao tác

- Đơn Hàng
- Chi tiết_2 ĐH
- Chi tiết_1 ĐH
- Cập nhật SL

Begin Tran →Bắt đầu giao tác

INSERT → Đơn hàng

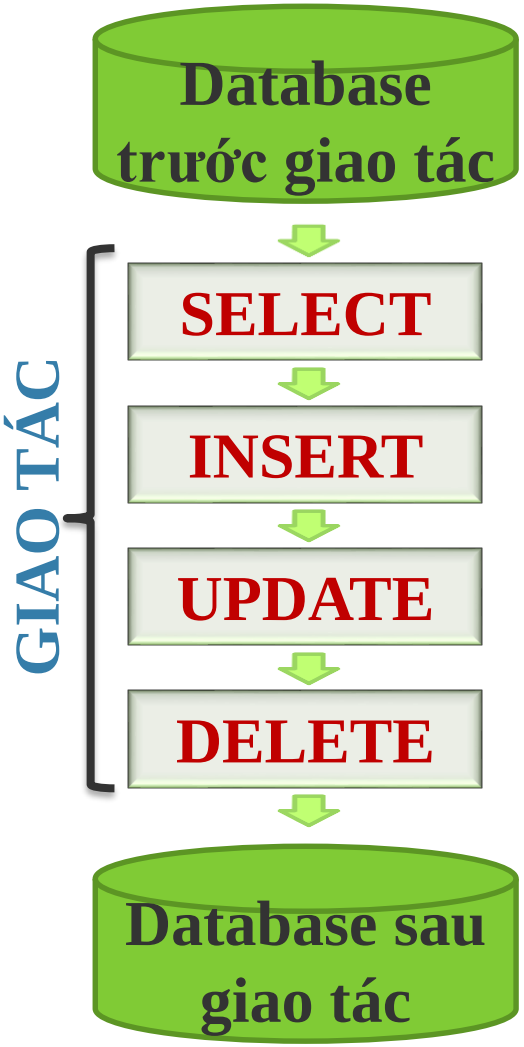
INSERT → Chi tiết đơn hàng 1

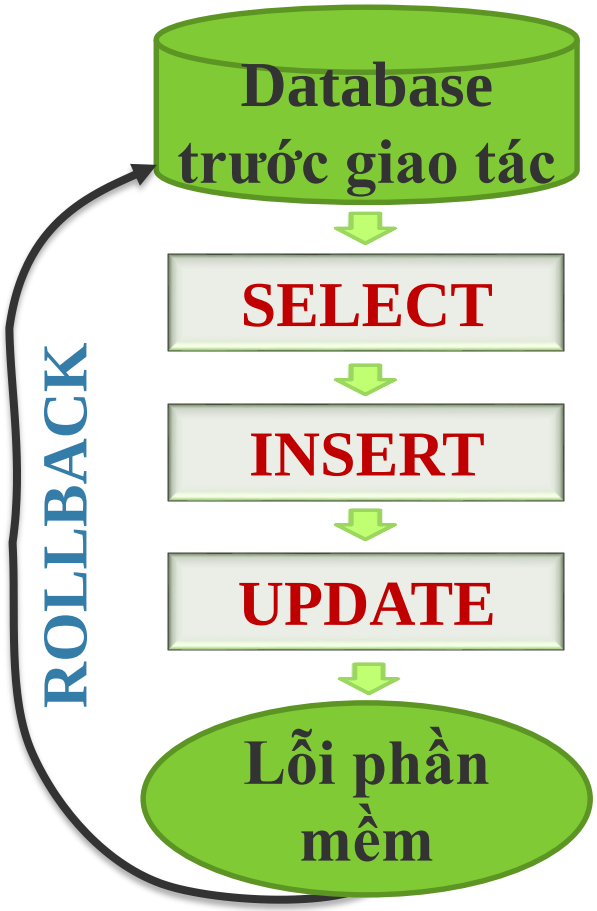
INSERT → Chi tiết đơn hàng 2

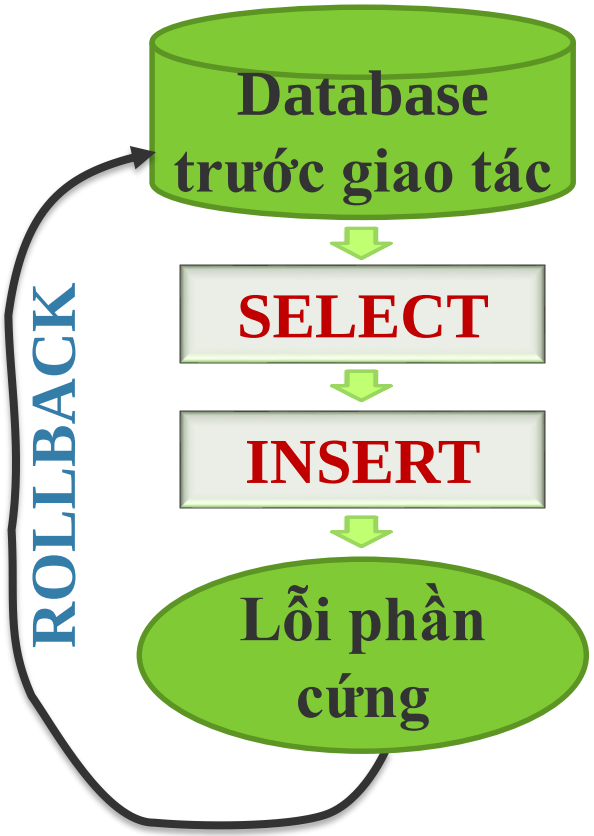
Update → Cập nhật số sản phẩm

End Tran →Kết thúc giao tác

Tình huống giao tác







Ví dụ

Database trước giao tác

Begin Tran →Bắt đầu giao tác

INSERT → Đơn hàng

INSERT → Chi tiết đơn hàng 1

INSERT → Chi tiết đơn hàng 2

Update → Cập nhật số sản phẩm

Rollback

Rollback

End Tran →Kết thúc giao tác

Database trước giao tác

Commit

Cú pháp

```
Create  proc  TenProc
As
Begin
   Begin  tran
      /*giao tác bị lỗi*/
   rollback tran
      /*giao tác thành công*/
   commit tran
End
```

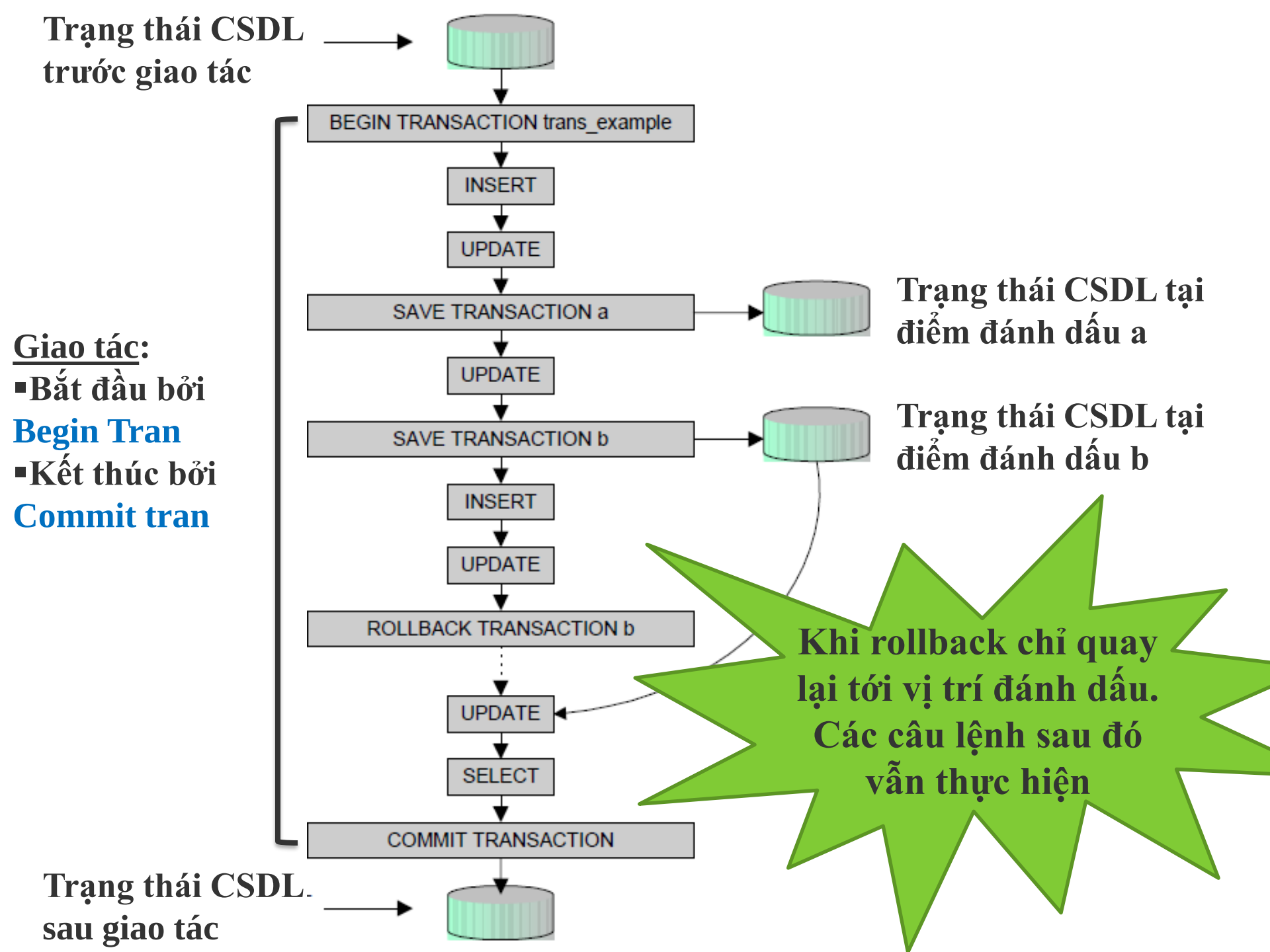
Bắt đầu giao tác

Kết thúc giao tác → hủy giao tác, đưa CSDL về trạng thái ban đầu

Kết thúc giao tác thành công→ giao tác được lưu bền vững xuống CSDL

Ví dụ

```
Create  proc  DatHang
As
Begin
    Begin  tran
        Insert into DonHang
        Insert into CT_DonHang
        Update SanPham
    commit tran
End
```


Trạng thái CSDL trước giao tác
BEGIN TRANSACTION trans_example
INSERT
UPDATE
SAVE TRANSACTION a
Trạng thái CSDL tại điểm đánh dấu a
UPDATE
SAVE TRANSACTION b
Trạng thái CSDL tại điểm đánh dấu b
INSERT
UPDATE
ROLLBACK TRANSACTION b
UPDATE
SELECT
COMMIT TRANSACTION
Trạng thái CSDL sau giao tác
Giao tác:
Bắt đầu bởi Begin Tran
Kết thúc bởi Commit tran
Khi rollback chỉ quay lại tới vị trí đánh dấu. Các câu lệnh sau đó vẫn thực hiện

Bối cảnh

Khách hàng

Mua sản phẩm

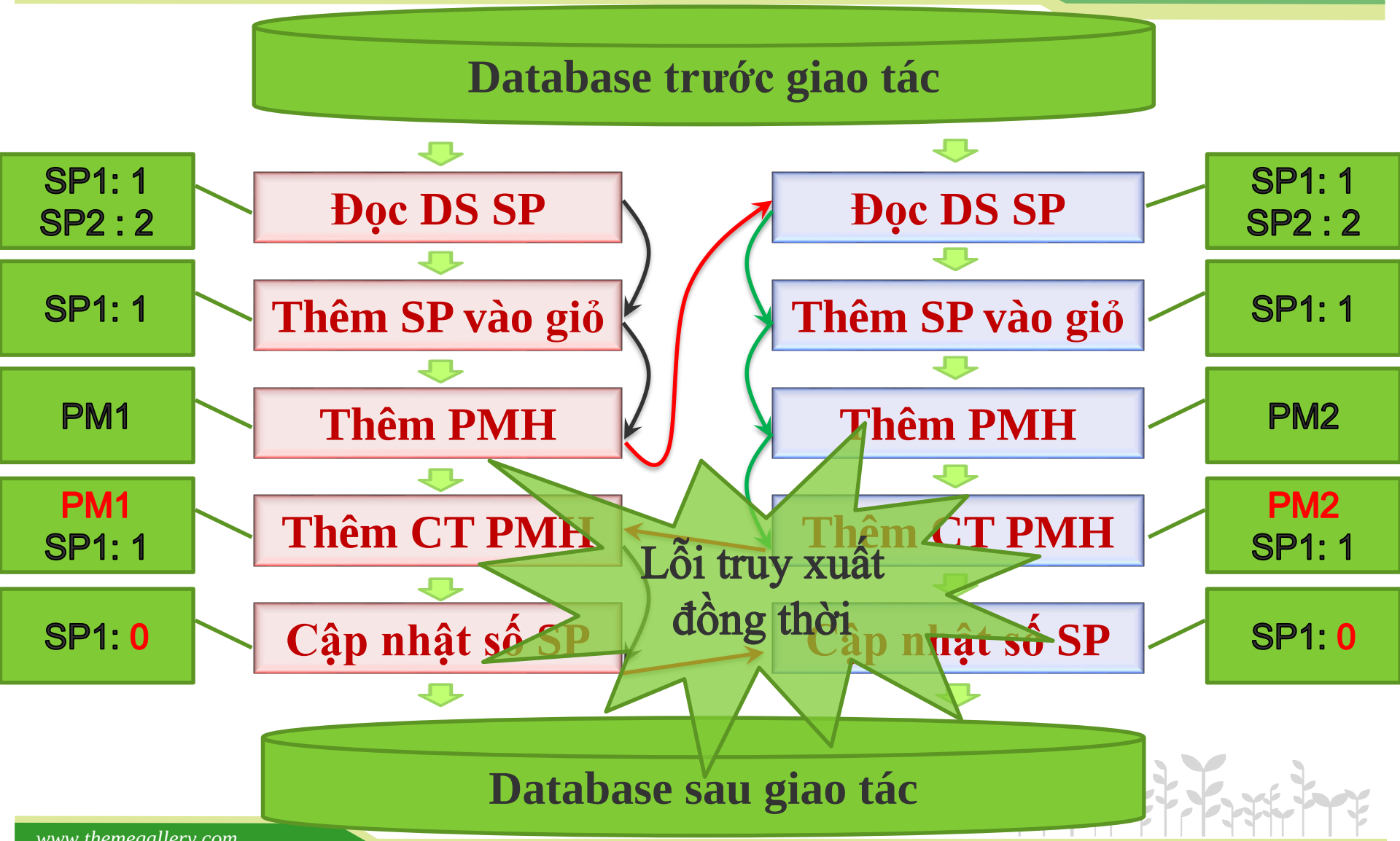
Website bàn hàng

Mua sản phẩm

Khách hàng

Mua cùng sản phẩm

Bối cảnh


Database trước giao tác
SP1: 1
SP2 : 2
Đọc DS SP
Đọc DS SP
SP1: 1
SP2 : 2
SP1: 1
Thêm SP vào giỏ
Thêm SP vào giỏ
SP1: 1
PM1
Thêm PMH
Thêm PMH
PM2
PM1
SP1: 1
Thêm CT PMH
Thêm CT PMH
PM2
SP1: 1
SP1: 0
Cập nhật số SP
Cập nhật số SP
SP1: 0
Lỗi truy xuất
đồng thời
Database sau giao tác

Bối cảnh

- ***Mô hình ứng dụng đa người dùng***

  Một CSDL lưu tại Server và nhiều clients đồng thời truy cập và thao tác trên cùng CSDL

- Giao tác T_i của ***client X*** đang thao tác trên CSDL, trong khi đó một giao tác T_j của ***client Y*** cũng đang thực hiện truy xuất và thay đổi CSDL.

  ⇨*Các T_i và T_j có thể xung đột, tranh chấp lẫn nhau.*

Giao tác lồng nhau

STT	T1	T2
1	Begin tran	
2		Begin tran
3	read (A)	
4		write (A)
5		Commit tran
….		
n	read (A)	
n+1	Commit tran	

Giao tác lồng nhau

❖ Tham số ***@@trancount*** cho biết số transaction đang thực thi.

❖ Khi khai báo ***transaction tường minh***, phải ***rollback*** hoặc ***commit tường minh*** để:

- Giải phóng tài nguyên transaction đang chiếm giữ.
- Tránh cản trở việc thực hiện của các transaction khác.

Vấn đề truy xuất đồng thời

Dirty read	Đọc dữ liệu rác
Unrepeatable read	Không thể đọc lại dữ liệu
Phantom	Bóng ma
Lost update	Mất dữ liệu đã cập nhật

Dirty read

STT	T1	T2
1	Begin tran	
2		Begin tran
3	write (A) //Insert | Update	
4		read (A)
5	If (Lỗi)	
6	Rollback tran	
7	Commit tran	
8		Commit tran

⇨ T_2 đọc dữ liệu (rác) đã bị T_1 hủy.

Unrepeatable read

STT	T1	T2
1	Begin tran	
2		Begin tran
3	Read(A)	
4		Write(A) //Update | Delete
5	Read(A)	
6		Commit tran
7	Commit tran	

⇨ T_1 đọc dữ liệu giá trị A khác nhau ở 2 lần đọc.

Phantom

STT	T1	T2
1	Begin tran	
2		Begin tran
3	Read(A)	
4		write(B) //Insert | Update
5	Read(A, B)	
6		Commit tran
7	Commit tran	

⇨ T_1 đọc tập dữ liệu 2 lần khác nhau

Lost update

STT	T1	T2
1	Begin tran	
2		Begin tran
3	Read(A)	
4		Read(A)
5		Write(A')
6	Write(A")	
7	Commit tran	
8		Commit tran

⇨ Dữ liệu được ghi bởi T_2 đã bị ghi đè bởi T_1

Lỗi giao tác

- Không có quyền truy cập trên đối tượng (table, stored procedure,…)
- Deadlock.

Lỗi giao tác

- SQL Server trả giá trị lỗi về trong biến toàn cục ***@@error***.
  - **@@error= 0**: không xảy ra lỗi
  - **@@error <> 0:** xảy ra lỗi với mã lỗi là @@error
- Giao tác không thể tự động rollback khi gặp những lỗi phát sinh trong quá trình thực hiện 1 câu lệnh thành phần trong giao tác. Vì vậy cần kiểm tra giá trị của biến **@@error** sau mỗi câu lệnh thành phần trong giao tác và cần xử lý những lỗi (nếu có): yêu cầu giao tác rollback một cách tường minh bằng lệnh rollback transaction.

Lỗi giao tác

```
Create proc sp_ThemDG
@ Ten…
as
--buoc 1 : xác định mã độc giả
  declare @madg
  set  @madg = 1
  begin transaction
  while exists (select * from
    DocGia where ma_docgia =
    @madg)
        set @madg = @madg +1
   if ( @@error <>0 )
   begin
        rollback tran
        return
      end
-- buoc 2 : insert vao bang docgia
  insert into DocGia values(…)
  if ( @@error <>0 )
     begin
        rollback tran
        return
      end
…
commit transaction
```

Xử lý tranh chấp

- **Cơ chế xử lí tranh chấp đồng thời của SQL Server**
  - *Kỹ thuật khóa* : các giao tác muốn đọc/ghi trên các đơn vị dữ liệu phải phát ra yêu cầu xin khóa trên đơn vị dữ liệu đó.
  - *Mức cô lập* : là các thiết lập trong giao tác quy định việc xin khóa/giữ khóa của những thao tác đọc/ghi lên đơn vị dữ liệu.
  - *Khóa trực tiếp trong từng câu lệnh*

Kỹ thuật khóa

- **Không đặt khóa (*Nolock*)**
- **Khóa chia sẻ (*shared lock*) :**
  - Còn gọi là khóa đọc (***read lock***) . Gọi tắt : Khóa **S**
  - Khi đọc một đơn vị dự liệu SQL tự thiết lập shared lock trên đơn vị dữ liệu đó.
  - Shared lock có thể được thiết lập trên 1 trang, 1 bảng, hay một dòng dữ liệu.

Kỹ thuật khóa

- **Khóa cập nhật (*update lock*)**
  - Còn gọi *là **Intend to write lock***. Gọi tắt: Khóa **U**.
  - Dùng khi có dự định ghi dữ lại dữ liệu đã đọc.
- **Khóa độc quyền (*exclusive lock*) :**
  - Còn gọi là khóa ghi (***write lock***). Gọi tắt : Khóa **X**.
  - Khi thực hiện thao tác ghi (Insert, Update, Delete) HQT tự động thiết lập khóa **X** trên đơn vị dữ liệu đó.
  - Khóa X được giữ đến hết giao tác.

Bảng tương thích khóa

	Shared lock	Update lock	Exclusive lock
Shared lock	☑	☑	☒
Update lock	☑	☒	☒
Exclusive lock	☒	☒	☒

☑ Cho phép (tương thích)

☒ Không cho phép (không tương thích)

Kỹ thuật khóa

❖ **Giáo tác T_1**

- Ghi trên đơn vị dữ liệu A → *Xin khóa X trên A.*
- Muốn ghi trên đơn vị dữ liệu B → *Muốn xin khóa X trên B???*

❖ **Giáo tác T_2**

- Ghi trên đơn vị dữ liệu B → *Xin khóa X trên B.*
- Muốn ghi trên đơn vị dữ liệu A → *Muốn xin khóa X trên A???*

Phải chờ nhau

Mức cô lập

- **Read Uncommitted**
  - Đọc : Không phát S khi đọc do đó không phải chờ khi đọc dữ liệu.
  - Giải quyết : Không giải quyết được bất cứ vấn đề xử lý đồng thời nào.
- **Read Committed**
  - Đây là mức mặc định của SQL.
  - Đọc : Phát S khi đọc, giải phóng S ngay sau khi đọc.
  - Giải quyết : Chỉ giải quyết được **Dirty Read**.

Mức cô lập

- **Repeatable Read**
  - Đọc : Phát S khi đọc và giữ S đến khi transaction kết thúc.
  - Giải quyết : Giải quyết được **Dirty Read** và **Unrepeatable Read**. Không ngăn chặn lệnh insert dữ liệu thỏa điều kiện thiết lập S do đó không giải quyết được **phantom**.
- **Serializable**
  - Giống ***Repeatable Read,*** có ngăn chặn lệnh insert dữ liệu thoả điều kiện thiết lập S. Giải quyết được **Dirty Read**,**Unrepeatable Read** và **Phantom**

Mức cô lập

- **Read Uncommitted**
- **Read Committed :** dirty read
- **Repeatable Read :** dirty read và unrepeatable read
- **Serializable :** dirty Read, Unrepeatable Read và Phantom.

⇨ **Chưa giải quyết được Lost update**

⇨ **Mức cô lập có tác dụng trên toàn bộ giao tác đến khi gặp 1 mức cô lập khác.**

Khóa trên từng câu lệnh

- Cấp độ khóa là các loại khóa khác nhau được gắn vào từng table trong câu lệnh from của từng thao tác select.
- Ngoài câu lệnh select, cấp độ khóa còn có thể gắn vào các câu lệnh cập nhật, tuy nhiên ở đây ta chỉ quan tâm câu select.

Các cấp độ khóa

STT	Khóa	Ý nghĩa
1	ReadUncommitted / Nolock	Không thiết lập shared lock khi đọc (tương tự như mức cô lập Read Uncommitted)
2	ReadCommitted	-Đây là chế độ mặc định (tương tự readcommitted) -Chỉ đọc những dữ liệu đã commit -Thiết lập shared lock trên đơn vị dữ liệu đọc và mở clock ngay khi đọc xong.
3	RepeatableRead	Thiết lập shared lock khi đọc và giữ shared lock đến hết giao tác

Các cấp độ khóa

STT	Khóa	Ý nghĩa
4	Serializable / Holdlock	-Thiết lập shared lock khi đọc, giữ lock đến hết giao tác. -Không cho insert dòng thỏa điều kiện thiết lập khóa
5	Updlock	-Dùng updatelock thay cho shared lock. -Chỉ sử dụng trong câu select. -Uplock được giữ đến hết giao tác.
6	XLock	Chỉ định dùng khóa độc quyền
7	Readpast	-Chỉ khóa dòng dữ liệu đang thao tác. -Áp dụng cho câu lệnh select. -Chỉ dùng được với READ COMMITTED hoặc REPEATABLE READ

Các cấp độ khóa

STT	Khóa	Ý nghĩa
8	RowLock	Chỉ đặt khóa trên dòng cần thao tác
9	TabLock	-Khóa toàn bộ dòng trên bảng đang thao tác. -Các thao tác (Insert / Update / Delete) không thể thực hiện trên bảng này.
10	TabLockX	Xlock + TabLock

⇨ 1, 2, 3, 5, 6, 7 chỉ có ý nghĩa khi dùng trong câu select.

⇨ 1, 2, 3, 5, 6, 7 có thể kết hợp với 4 (khóa theo kiểu key-range) và 8, 9 (chỉ ra đơn vị cần khóa)

Thiết lập cấp độ khóa

Select …

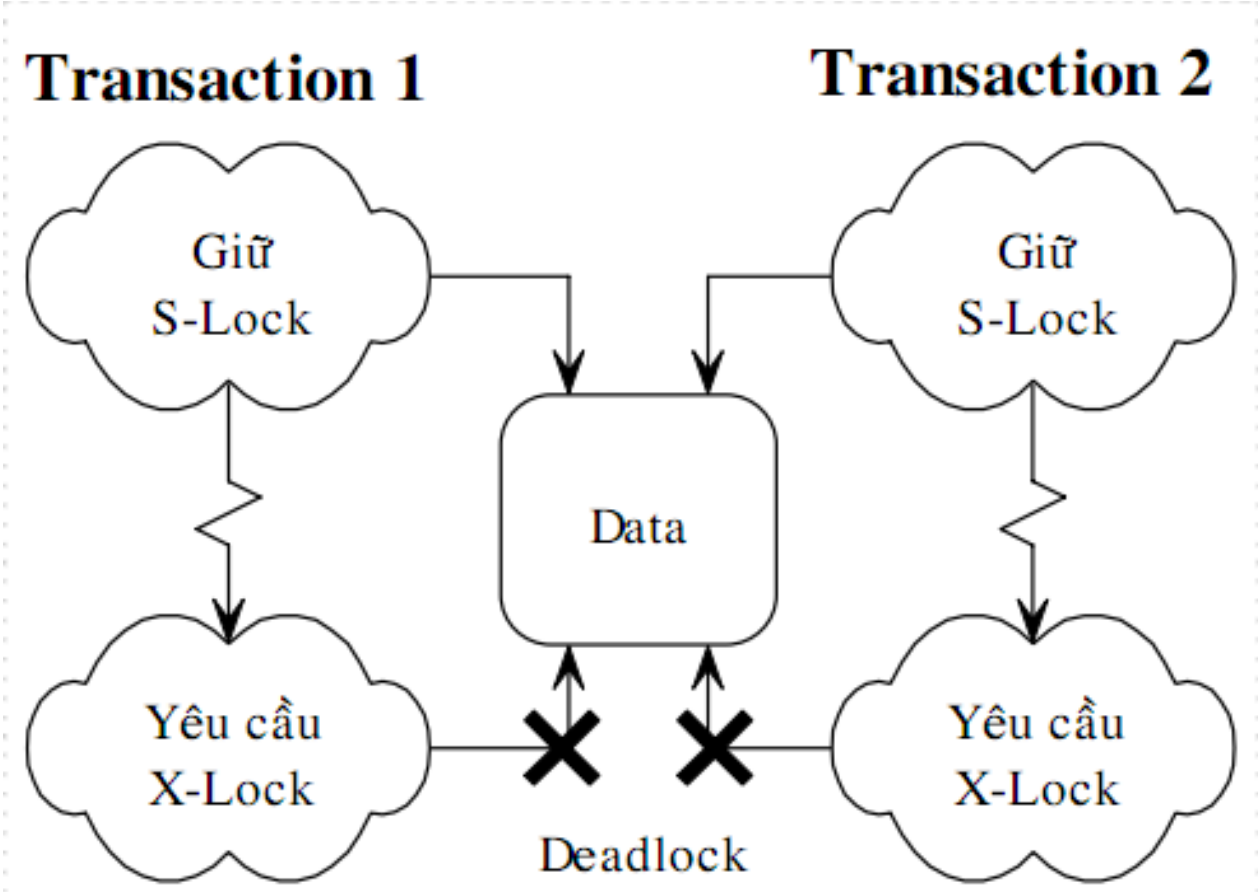
From {Tab1 Alias1 with (*Lock_mode* [,…n]} [,…n])

Where …

Ví dụ :

```
Select SV.HoVaTen, K.TenKhoa
From SinhVien SV with (ReadCommitted),
        Khoa K with (Updlock)
Where SV.Khoa = K.Ma
```

Deadlock

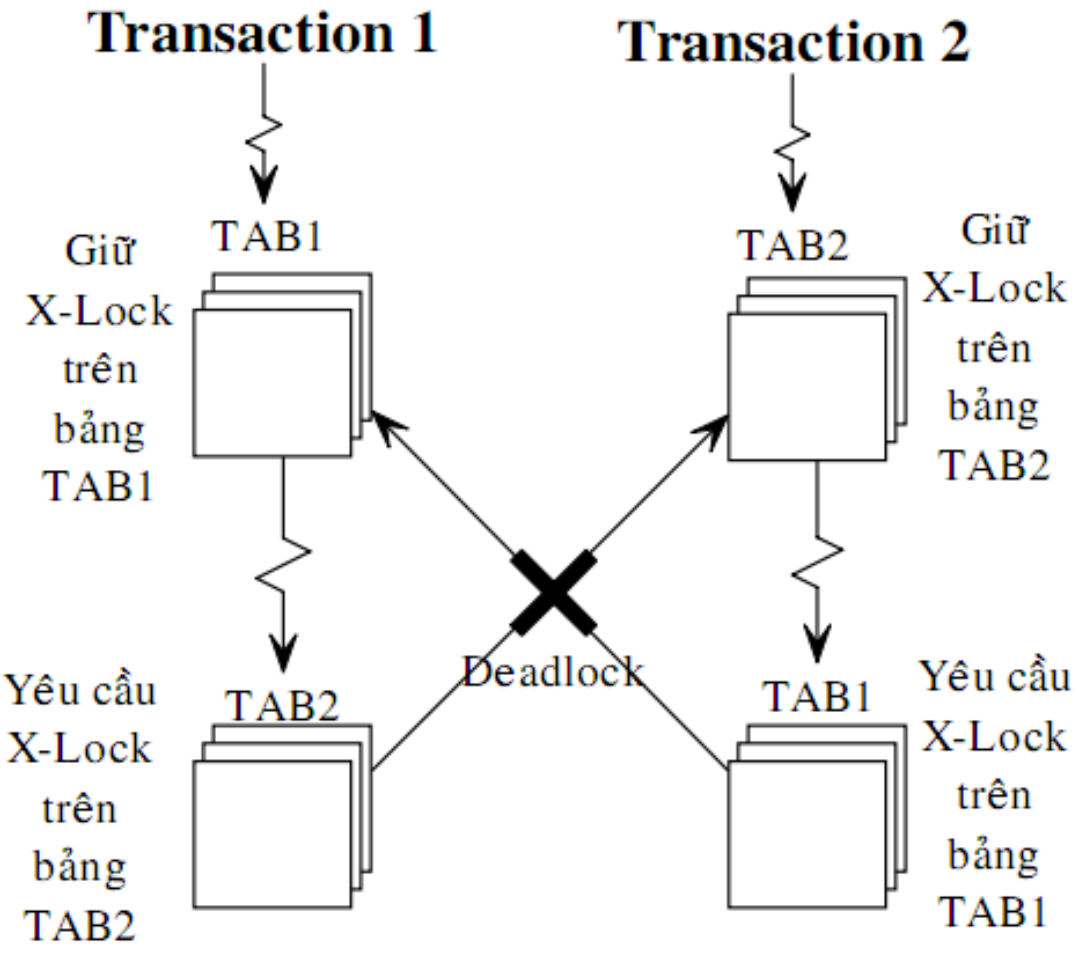


Conversion deadlock

Conversion deadlock

Transaction 1	Transaction 1
Select (A)	
	Select (A)
Insert/ Update/Delete (A)	
	Insert/ Update/Delete (A)
Commit	Commit

Deadlock



Cycle deadlock

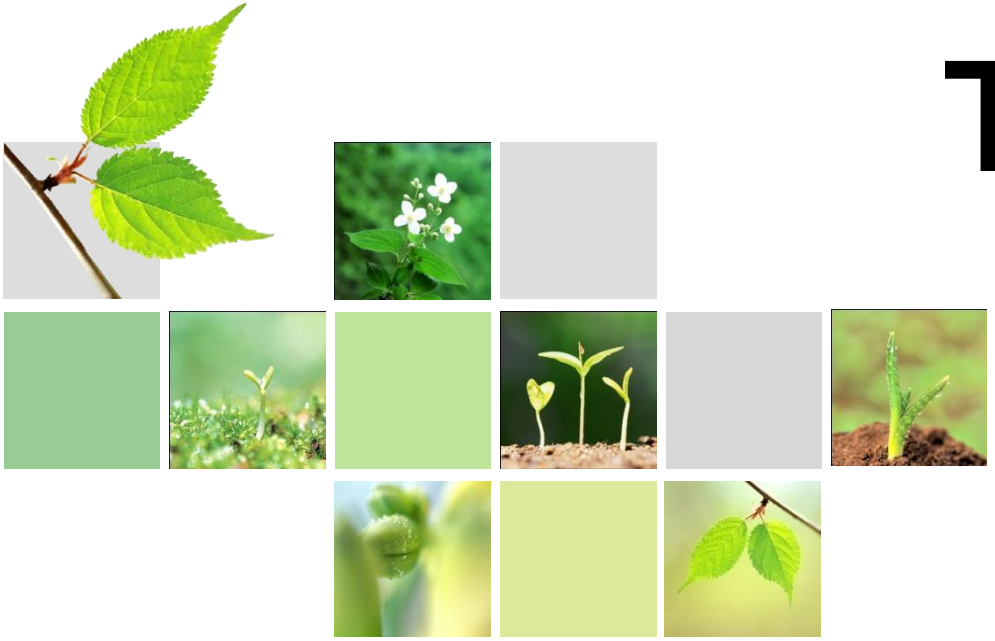
Cycle deadlock

Transaction 1	Transaction 1
Insert/ Update/Delete (A)	
	Insert/ Update/Delete (B)
Insert/ Update/Delete (B)	
	Insert/ Update/Delete (A)
Commit	Commit

Deadlock

- **Khi dead lock xảy ra**
  - SQL Server sẽ chọn 1 trong 2 transaction gây dead lock để hủy bỏ, khi đó transaction còn lại sẽ được tiếp tục thực hiện cho đến khi hoàn tất.
  - Transaction bị chọn hủy là transaction mà SQL ước tính chi phí cho phần việc đã làm được ít hơn transaction còn lại.

Thank You!



Bài tập

STT	T1	T2
1	**Begin tran**	
2		**Begin tran**
3	SELECT * FROM SACH	
4		INSERT SACH (MaSach) Values (‘S003’)
5	SET @Msach = (SELECT Top 1 MaSach FROM SACH ORDER BY MaSach DESC)	
6		**Rollback**
7	…	
8	**Commit**	

Bài tập 2

STT	T1	T2
1	**Begin tran**	
2		**Begin tran**
3	SELECT * FROM SACH	
4		INSERT SACH (MaSach) Values (‘S003’)
5	SET @Msach = (SELECT Top 1 MaSach FROM SACH ORDER BY MaSach DESC)	
6		**Commit**
7	…	
8	**Commit**	

Bài tập 3

STT	T1	T2
1	**Begin tran**	
2		**Begin tran**
3	SELECT * FROM SACH	
4		UPDATE SACH Set TenSach = ‘ABC’ WHERE MaSach = ‘S003’
5	SELECT * FROM SACH	
6		**Commit**
7	…	
8	**Commit**	

Bài tập 4

STT	T1	T2
1	**Begin tran**	
2		**Begin tran**
3	SELECT * FROM PHIEUDATHANG	
4		SELECT * FROM PHIEUDATHANG
5		INSERT PHIEUDATHANG (MaPD, MaKH) VALUES (‘PD004’,’KH003’)
6	INSERT PHIEUDATHANG (MaPD, MaKH) VALUES (‘PD004’,’KH002’)	
7		**Commit**
8	**Commit**	

Bài tập 5

STT	T1	T2
1	**Begin tran**	
2		**Begin tran**
3	SELECT * FROM PHIEUDATHANG WHERE MaKH = ‘KH002’	
4		INSERT PHIEUDATHANG (MaPD, MaKH) VALUES (‘PD004’,’KH002’)
5	SELECT * FROM PHIEUDATHANG	
6	INSERT PHIEUDATHANG (MaPD, MaKH) VALUES (‘PD005’,’KH002’)	
7		**Commit**
8	**Commit**	

Bài tập 6

STT	T1	T2
1	**Begin tran**	
2		**Begin tran**
3	INSERT PHIEUDATHANG (MaPD, MaKH) VALUES (‘PD005’,’KH002’)	
	SELECT * FROM PHIEUDATHANG WHERE MaKH = ‘KH002’	
4		SELECT * FROM PHIEUDATHANG WHERE MaKH = ‘KH002’
5		UPDATE PHIEUDATHANG SET MaKH = ‘KH003’ WHERE MaPD = ‘PD005’
6	SELECT * FROM PHIEUDATHANG WHERE MaKH = ‘KH002’	
7		**Commit**
8	**Commit**	