

# **Nguyên lý Hệ quản trị CSDL**

## **Chương 5: Hệ thống phục hồi (Recovery System)**

# Nội dung

- ◆ Phân lớp hư hỏng
- ◆ Cấu trúc lưu trữ
- ◆ Phục hồi dựa trên sổ ghi lô trình
  - Sự cập nhật có trì hoãn
  - Sự cập nhật tức thời
  - Điểm kiểm soát
- ◆ Phân trang bóng
- ◆ Phục hồi dành cho các giao dịch song song

# Phân lớp hư hỏng

## ◆ Các loại hư hỏng:

- Giao dịch:
  - ◆ Lỗi luận lý: dữ liệu đầu vào, tràn giá trị,...
  - ◆ Lỗi hệ thống: deadlock, cạnh tranh,...
- Hệ thống: hệ điều hành, RAM,...
- Đĩa

◆ Phục hồi: Xác định loại hư hỏng  $\Rightarrow$  đánh giá sự ảnh hưởng đến dữ liệu  $\Rightarrow$  đề xuất giải pháp đảm bảo tính bền vững và nguyên tử  $\Rightarrow$  **giải thuật phục hồi lỗi**

## ◆ Giải thuật phục hồi: bao gồm

- Các hoạt động trong quá trình các giao dịch thực hiện
- Các hoạt động sau khi lỗi phát sinh

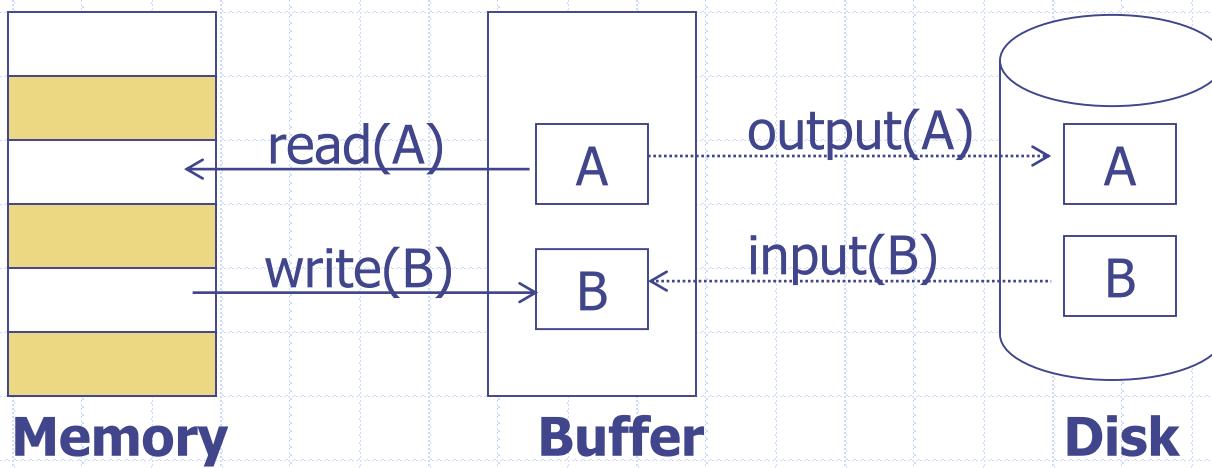
# Cấu trúc lưu trữ

## ◆ Các loại lưu trữ:

- Lưu trữ bị phai (bay hơi)
- Lưu trữ không phai (không bay hơi)
- Lưu trữ bền: thông tin *không bao giờ* bị mất (!??)

## ◆ Thực hiện lưu trữ bền.

## ◆ Phương pháp truy cập dữ liệu:



# Phục hồi dựa trên Sổ ghi lô trình (log)

- ◆ Dùng sổ ghi lô trình ghi nhận lại các thay đổi trên CSDL
- ◆ Một thao tác cập nhật  $\Rightarrow$  một log record
- ◆ Các loại log record:

Loại Log Record	Ý nghĩa
$<T_i, \text{Start}>$	GD $T_i$ đã khởi động.
$<T_i, X, V_1, V_2>$	GD $T_i$ thay đổi giá trị của $X$ từ $V_1$ thành $V_2$
$<T_i, \text{commit}>$	GD $T_i$ đã bàn giao.
$<T_i, \text{abort}>$	GD $T_i$ đã hủy bỏ.

- ◆ Sổ ghi lô trình ghi vào các thiết bị “bên”
- ◆ Có 2 giải thuật phục hồi dựa trên Sổ ghi lô trình

# Sự cập nhật có trì hoãn

- ◆ Dùng khi các GD được thực hiện tuần tự
- ◆ Tất cả các thao tác cập nhật CSDL sẽ bị trì hoãn
- ◆ Sự thực thi của các giao dịch tiến hành như sau:
  - Trước khi  $T_i$  khởi động:  $\langle T_i, \text{Start} \rangle$
  - Trước khi  $T_i$  Write(X):  $\langle T_i, X, V_2 \rangle$
  - Khi  $T_i$  bàn giao một phần:  $\langle T_i, \text{commit} \rangle$

- ◆ Phục hồi:  
 $\forall T_i: \langle T_i, \text{start} \rangle$  và  $\langle T_i, \text{commit} \rangle$  có trong sổ ghi lô trình  
 $\Rightarrow \text{redo}(T_i)$

**redo( $T_i$ )**: cập nhật giá trị mới cho tất cả các hạng mục dữ liệu được cập nhật bởi  $T_i$

# Sự cập nhật có trì hoãn

❖ Ví dụ:

$T_1$	$T_2$
R(A)	R(C)
A=A-50	C=C-100
W(A)	W(C)
R(B)	
B=B+50	
W(B)	

$A = 1000, B = 2000, C = 700$

$\langle T_1, \text{start} \rangle$   
 $\langle T_1, A, 950 \rangle$   
 $\langle T_1, B, 2050 \rangle$   
 $\langle T_1, \text{commit} \rangle$   
 $\langle T_2, \text{start} \rangle$   
 $\langle T_2, C, 600 \rangle$   
 $\langle T_2, \text{commit} \rangle$

# Sự cập nhật tức thời

- ◆ Các thao tác thay đổi giá trị các hạng mục CSDL sẽ **thể hiện ngay** lên CSDL
- ◆ Sự thực thi các GD được tiến hành như sau:
  - Trước khi  $T_i$  khởi động,  $\langle T_i, \text{Start} \rangle$
  - Trước khi  $T_i$  Write(X),  $\langle T_i, X, V_1, V_2 \rangle$
  - Khi  $T_i$  hoàn thành,  $\langle T_i, \text{commit} \rangle$
- ◆ Phục hồi:
  - $\forall T_i: \langle T_i, \text{start} \rangle$  và  $\langle T_i, \text{commit} \rangle$  có trong sổ ghi lô trình  
 $\Rightarrow \text{redo}(T_i)$
  - $\forall T_i: \langle T_i, \text{start} \rangle$  có trong sổ ghi lô trình nhưng không có  
 $\langle T_i, \text{commit} \rangle \Rightarrow \text{undo}(T_i)$
  - Thực hiện sự phục hồi từ dưới lên

# Sự cập nhật tức thời

## ◆ Ví dụ:

	$T_1$	$T_2$
$R(A)$		$R(C)$
$A=A-50$		$C=C-100$
$W(A)$		$W(C)$
$R(B)$		
$B=B+50$		
$W(B)$		

$A = 1000, B = 2000, C = 700$

$\langle T_1, \text{start} \rangle$   
 $\langle T_1, A, 1000, 950 \rangle$   
 $\langle T_1, B, 2000, 2050 \rangle$   
 $\langle T_1, \text{commit} \rangle$   
 $\langle T_2, \text{start} \rangle$   
 $\langle T_2, C, 700, 600 \rangle$   
 $\langle T_2, \text{commit} \rangle$

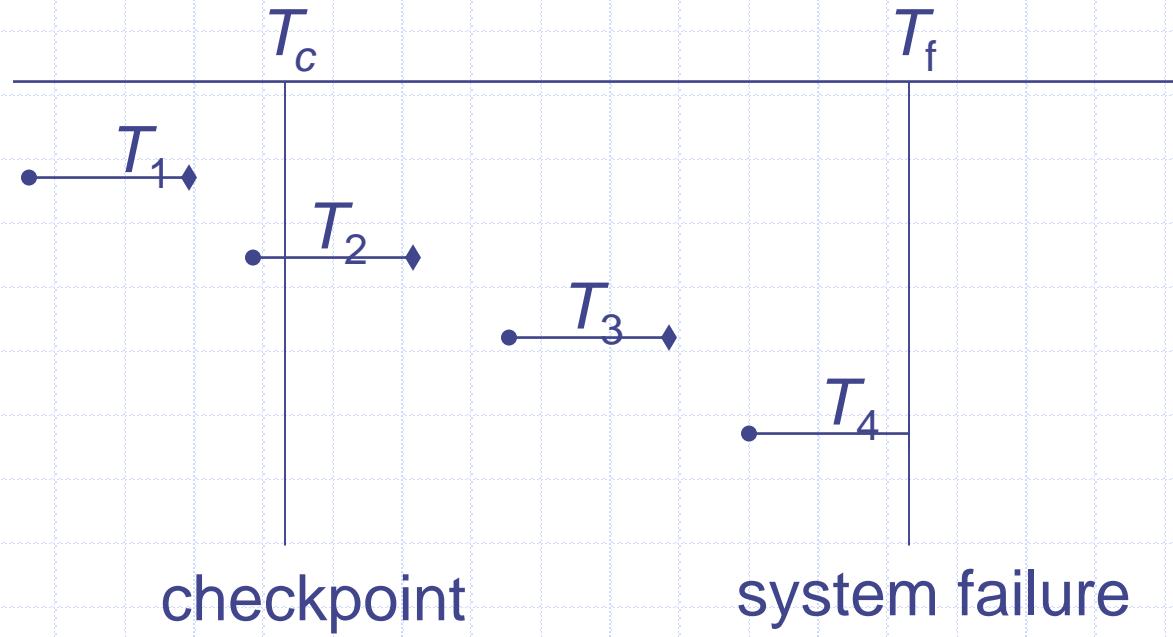
- ◆ **Redo( $T_i$ )**: đặt giá trị mới cho các hạng mục CSDL
- ◆ **Undo( $T_i$ )**: đặt giá trị cũ cho các hạng mục CSDL

# Điểm kiểm soát (Checkpoint)

- ◆ Dùng để cải thiện hiệu năng của quá trình khôi phục
- ◆ Muốn đặt điểm kiểm soát:
  1. Ghi log record vào thiết bị lưu trữ
  2. Cập nhật các khôi đệm đã cập nhật lên CSDL
  3. Thêm **<checkpoint>** vào sổ ghi lô trình
- ◆ Phục hồi:
  1. Từ điểm checkpoint cuối cùng, dò ngược lên tìm **<T<sub>i</sub>, start>** gần nhất.
  2. Gọi **T** là tập các GD gồm **T<sub>i</sub>** và các GD diễn ra sau **T<sub>i</sub>**
    - $\forall T_k \in T: <T_k, \text{commit}>$  không có trong SGLT
      - Nếu Cập nhật tức thời:  $\Rightarrow \text{undo}(T_k)$
      - Nếu Cập nhật trì hoãn:  $\Rightarrow$  bỏ qua **T<sub>k</sub>** (không undo)
    - $\forall T_k \in T: <T_k, \text{commit}>$  có trong SGLT  $\Rightarrow \text{redo}(T_k)$

# Điểm kiểm soát

❖ Ví dụ:



# Phục hồi cho các giao dịch song song

- ◆ Dùng sổ ghi lô trình với sự cập nhật tức thời
- ◆ Checkpoint: **<checkpoint L>** với L là tập các GD đang hoạt động tại thời điểm đặt checkpoint
- ◆ Phục hồi:
  1. **Tạo hai danh sách redo-list và undo-list:**
    - a. Redo-list =  $\emptyset$ ; undo-list =  $\emptyset$
    - b. Dò ngược SGLT đến khi gặp mẫu tin **<checkpoint L>** đầu tiên:
      - Nếu thấy **< $T_i$ , commit>**:  
⇒ thêm  $T_i$  vào redo-list
      - Nếu thấy **< $T_i$ , start>** và  $T_i \notin$  redo-list:  
⇒ thêm  $T_i$  vào undo-list
    - c.  $\forall T_i \in L$  nhưng  $T_i \notin$  redo-list và undo-list thêm  $T_i$  vào undo-list

# Phục hồi cho các giao dịch song song

## 2. Tiến trình phục hồi:

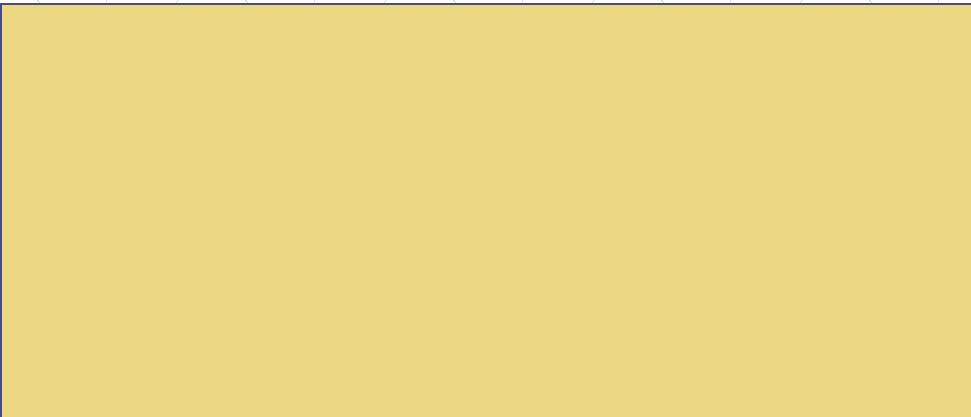
- a. Dò ngược sổ ghi đến khi tìm thấy  $\langle T_i, \text{start} \rangle$  cho tất cả các  $T_i$  trong undo-list, tiến hành undo đối với mỗi record-log thuộc undo-list
- b. Định vị  $\langle \text{checkpoint } L \rangle$  cuối cùng
- c. Dò xuôi sổ ghi lịch trình cho đến cuối sổ ghi, thực hiện redo với mỗi record-log thuộc giao dịch  $T_i$  nằm trong redo-list

# Phục hồi cho các giao dịch song song

Cho một sổ ghi lịch trình sau:

- Hãy xác định L1, L2.
- Xác định giá trị của A – E trên đĩa sau khi sự cố và trước khi phục hồi
- Xác định giá trị của A – E trên đĩa sau khi phục hồi.

Giải:



```
<T1, start>
<T2, start>
<T1, A, 10, 30>
<T1, B, 15, 35>
<T2, C, 110, 160>
<checkpoint L1>
<T1, commit>
<T2, D, 210, 260>
<T3, start>
<T3, E, 1010, 1510>
<checkpoint L2>
<T2, commit>
~~~ crash ~~~
```



8/16/2017

5.15