



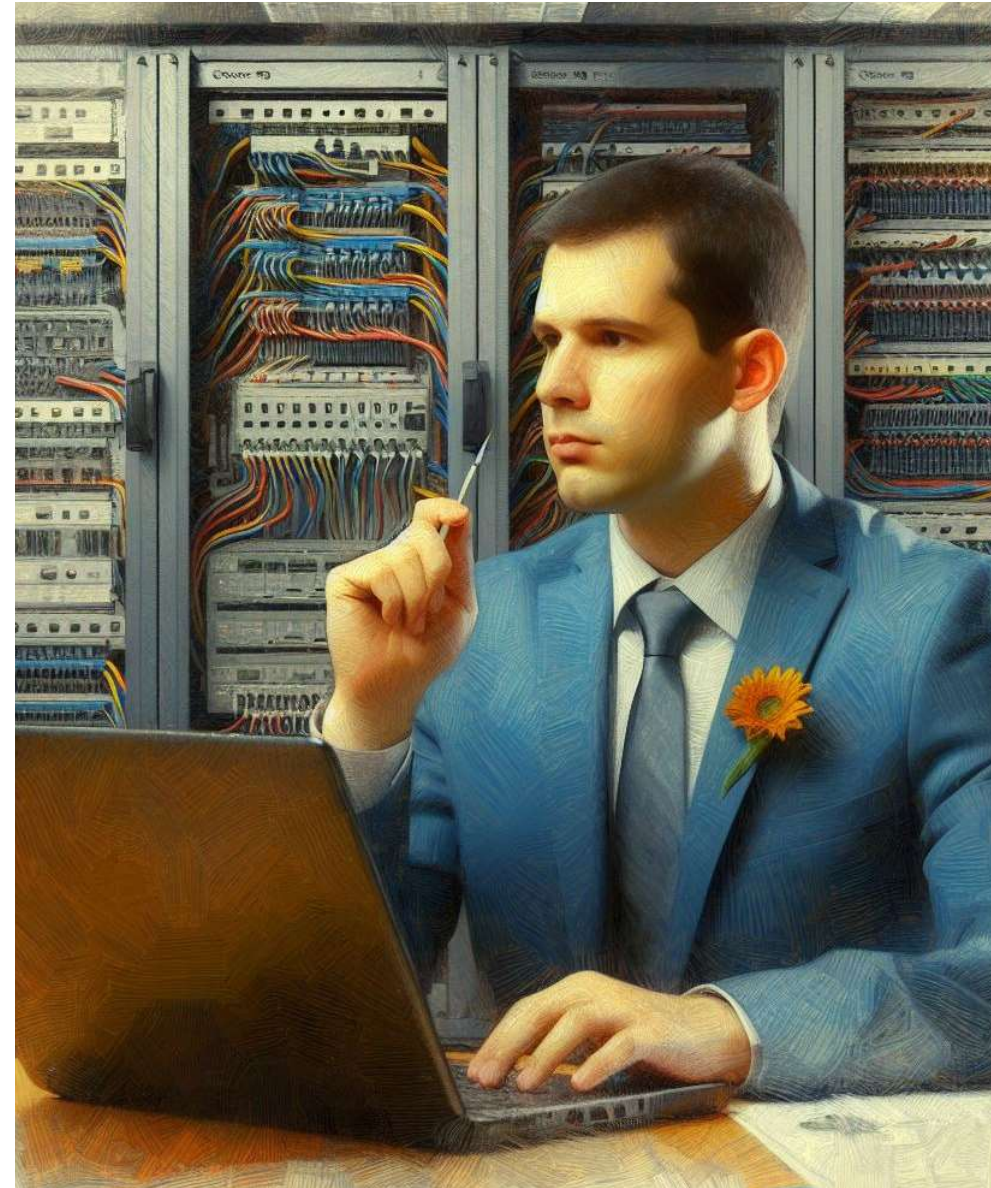
HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

NGUYEN MINH NHUT
INFORMATION SYSTEM ENGINEERING

1

DBMS
noun [C]
UK  /,di:.bi:.em'es/ US  /,di:.bi:.em'es/ [Add to word list](#)

abbreviation for database management system: a set of computer programs for allowing large amounts of information to be put into a computer and for organizing it so that it can be searched, examined, or printed easily and quickly



Database Management and Administration



2

CHƯƠNG 3

QUẢN LÝ GIAO TÁC

Oracle's Large Business Database



Trong chương này chúng ta sẽ học những nội dung:

- Tính ACID trong hệ quản trị CSDL
- Giao tác và Lịch giao tác trong Hệ quản trị CSDL
- Lịch tuần tự, lịch tuần tự
- Lịch khả tuần tự theo View
- Một số kiến thức liên quan khác



PHẦN 1

TÍNH ACID TRONG HỆ QUẢN TRỊ CSDL

Một hệ quản trị cơ sở dữ liệu thường làm việc các giao tác hay còn gọi là Transactions. Mỗi Transaction trong CSDL đảm bảo tính ACID (atomicity, Consistency, Isolation, Durability, đảm bảo tính đồng nhất và bền vững của giao dịch trong cơ sở dữ liệu)



Phần 1. Tính ACID và Transaction trong CSDL



• Định nghĩa Transaction

- Một cơ sở dữ liệu thường làm việc với một *Giao tác (Transaction)*, là tập hợp các hành động nhỏ, nhằm mục đích truy cập và sửa đổi CSDL.
- Trong quá trình làm việc với một *cơ sở dữ liệu* và các giao dịch liên quan của nó, rất quan trọng là các giao dịch được quản lý một cách *mượt mà, liền mạch và đáng tin cậy*.

Định nghĩa Transaction

Một cơ sở dữ liệu thường được sử dụng bởi **nhiều người dùng đồng thời**. Khi làm việc với cơ sở dữ liệu, người dùng sẽ **tương tác với cơ sở dữ liệu** để đạt được kết quả. Sự tương tác này với cơ sở dữ liệu là **một chuỗi các bước logic** được thực thi tuần tự. Đây được gọi là các **giao dịch** (Transaction).

Phần 1. Tính ACID và Transaction trong CSDL



• Quản lý Transactions

– Trên thực tế một CSDL được truy cập bởi *nhiều người dùng* → vào cùng *một thời điểm*. Điều này ngụ ý rằng có nhiều giao dịch *diễn ra đồng thời* cho nhiều người dùng trên một cơ sở dữ liệu duy nhất tại một thời điểm nhất định

– *Các vấn đề thường gặp: Sai dữ liệu, Treo hệ thống*

Định nghĩa Quản lý **Transaction**

Quản lý Transactions là quản lý truy cập nhiều người dùng vào một thời điểm, đảm bảo dữ liệu diễn ra đồng thời và nhất quán, không gây ra tình trạng *sai, mất, treo hệ thống*.

– Có 2 loại giao tác chính trong Transaction:

- Giao tác **Read**: **SELECT**
- Giao tác **Write**: **UPDATE, INSERT, DELETE**

Phần 1. Tính ACID và Transaction trong CSDL



• Ví dụ về một giao tác trong CSDL

Bạn A muốn chuyển cho bạn B 200.000 VNĐ, thì một khoản trừ của A sẽ A sẽ được cộng vào B

– Ví dụ trên sẽ được minh họa từng bước như sau:

- **Read** số dư tài khoản của bạn A
- Thực hiện phép gán số dư $A = \text{số dư A} - \text{khoản tiền chuyển}$
- **Write** dữ liệu gán số dư A vào CSDL
- **Read** số dư tài khoản của bạn B
- Thực hiện phép gán số dư $B = \text{số dư B} + \text{khoản tiền chuyển}$
- **Write** dữ liệu gán số dư B vào CSDL

– Tất cả những bước trên được gọi là giao tác trong CSDL.

– Các bước trên có thể dẫn đến trường hợp mất nhất quán dữ liệu. Việc giải quyết nhất quán dữ liệu có 2 hành động cụ thể:

- **COMMIT**: Nếu giao dịch đạt đến cuối của thứ tự được xác định, nếu nó lưu vào CSDL
- **ROLLBACK**: Thu hồi một giao dịch, hoàn tác các thay đổi mà các giao dịch không hoàn thành có thể đã thực hiện.

Phần 1. Tính ACID và Transaction trong CSDL

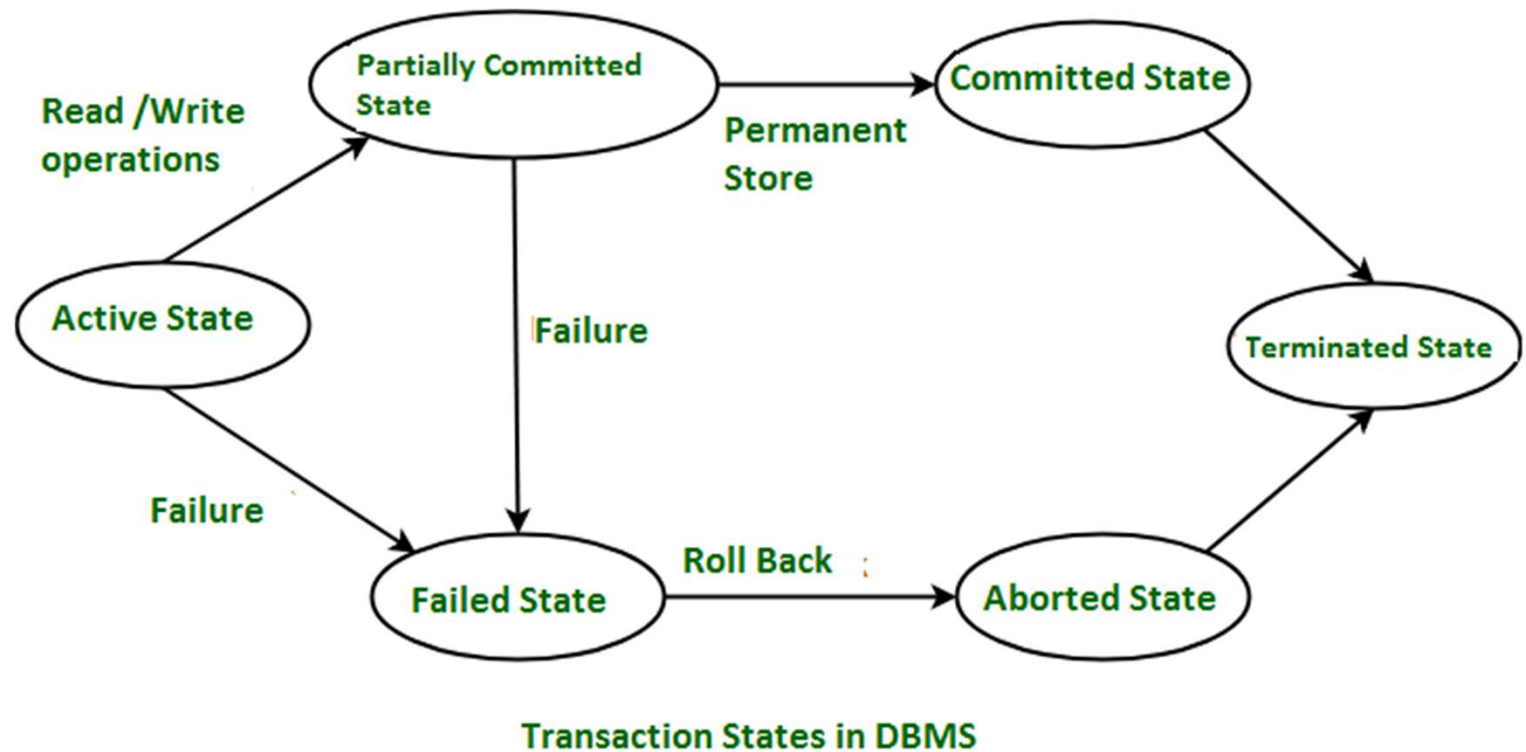


• Tính ACID trong Transaction

- **Tính nguyên tử (Atomicity):** Giao dịch được xem là hoàn thành hoặc không được thực hiện chút nào. Các hoạt động **COMMIT** và **ROLLBACK** được thiết kế để duy trì tính nguyên tử của cơ sở dữ liệu.
- **Tính nhất quán (Consistency):** Đảm bảo tính chính xác của cơ sở dữ liệu. Cơ sở dữ liệu phải ở trong một trạng thái nhất quán trước và sau giao dịch. Ví dụ, nếu một số tiền được trừ từ tài khoản người dùng, thì số tiền đó phải được thêm vào nơi khác để duy trì trạng thái tổng thể của hệ thống.
- **Tính cô lập (Isolation):** Cho phép một giao dịch thực thi độc lập mà không có sự can thiệp từ các giao dịch khác. Đảm bảo rằng nhiều giao dịch xảy ra đồng thời sẽ dẫn đến cùng một hiệu ứng như các giao dịch được thực thi tuần tự trên cơ sở dữ liệu.
- **Tính bền vững (Durability):** Đảm bảo rằng mỗi giao dịch hoàn thành được duy trì trong cơ sở dữ liệu và thông tin không bị mất ngay cả khi hệ thống gặp sự cố. Các giao dịch đã hoàn thành không được phép bị rollback. Thay vào đó, tính bền vững đảm bảo rằng nếu một giao dịch đã hoàn thành, thì nó phải có mặt trong cơ sở dữ liệu, và nếu giao dịch đã được thực hiện sai, thì một giao dịch khác phải được thực hiện để hoàn tác các thay đổi.

Phần 1. Tính ACID và Transaction trong CSDL

- Trạng thái trong Transaction





PHẦN 2

LỊCH GIAO TÁC, VÀ CÁC LOẠI LỊCH GIAO TÁC

Lịch giao tác là thứ tự thực thi của các giao dịch trong cơ sở dữ liệu. Có 4 loại lịch giao tác: Lịch tuần tự, Lịch Song Song, Lịch không truyền dẫn, lịch khả phục hồi.



Phần 2. Giới thiệu về lịch giao tác (Transaction Schedule)

• Lịch giao tác (Transaction Schedule)

- Kí hiệu đọc trên đơn vị dữ liệu $X \rightarrow R(X)$, ký hiệu viết trên dữ liệu $X \rightarrow W(X)$
- Khi nhiều giao dịch đang chạy đồng thời, thì cần phải có một chuỗi trong đó các hoạt động được thực hiện vì vào một thời điểm, chỉ có một hoạt động có thể được thực hiện trên cơ sở dữ liệu. Chuỗi các hoạt động này được gọi là lịch. Chuỗi hoạt động sau đây là một lịch. Ở đây, chúng ta có hai giao dịch T1 và T2 đang chạy đồng thời
- Ví dụ hai Transaction T1, T2

T1	T2
R(X)	
W(X)	
R(Y)	
	R(Y)
	R(X)
	W(Y)

Phần 2. Giới thiệu về lịch giao tác (Transaction Schedule)

- Các loại lịch giao tác:

- **Lịch giao tác tuần tự (Serial Transaction)**, một giao dịch được thực thi hoàn toàn trước khi bắt đầu thực thi giao dịch khác. Nói cách khác, chúng ta có thể nói rằng trong một lịch giao tác tuần tự, một giao dịch không bắt đầu thực thi cho đến khi giao dịch đang chạy kết thúc thực thi
- Ví dụ hai Transaction T1, T2 theo Serial Transaction

T1	T2
R(X)	
W(X)	
R(Y)	
	R(Y)
	R(X)
	W(Y)

Phần 2. Giới thiệu về lịch giao tác (Transaction Schedule)

- Các loại lịch giao tác:

- Lịch giao tác đồng thời (Concurrent Transaction), nhiều hơn một giao dịch được thực thi đồng thời.
- Ví dụ hai Transaction T1, T2 theo Concurrent Transaction

T1	T2
R(X)	R(Y)
W(Y)	R(Y)
	R(X)
	W(Y)

Phần 2. Giới thiệu về lịch giao tác (Transaction Schedule)

- Các loại lịch giao tác:

- **Lịch giao tác không truyền dẫn (Cascasdeless Transaction)**, nếu một giao dịch sẽ thực hiện hoạt động đọc trên một giá trị, nó phải chờ cho đến khi giao dịch đang thực hiện ghi vào giá trị đó được **COMMIT**.
- Ví dụ hai Transaction T1, T2, T3 theo Concurrent Transaction

T1	T2	T3
R(X)		
W(X)		
COMMIT		
	R(X)	
	W(X)	
	COMMIT	
		R(X)
		W(X)
		COMMIT

Phần 2. Giới thiệu về lịch giao tác (Transaction Schedule)

- Các loại lịch giao tác:

- **Lịch giao tác khả phục hồi (Recoverable Transaction)**, nếu một giao dịch đang đọc một giá trị đã được cập nhật bởi một giao dịch khác, thì giao dịch này chỉ có thể **COMMIT** sau khi giao dịch khác đó đã **COMMIT** để cập nhật giá trị.

Ví dụ hai Transaction T1, T2 theo Recoverable Transaction

T1	T2
R(X) W(X)	
	R(X) W(X) R(X)
COMMIT	COMMIT

Phần 2. Giới thiệu về lịch giao tác (Transaction Schedule)

- **Lịch khả tuần tự (Serializability Schedule):**

- *Một lịch giao tác S gồm n giao tác* được gọi là tuần tự hóa nếu nó có thể được thực hiện giống như một lịch giao dịch tuần tự của *n giao dịch tương tự*. Hai lịch S và S' ở dưới được gọi là đương nhau khi:

- S và S' phải cùng một tập hợp giao dịch
- Thứ tự mỗi cặp xung đột trong S và S' là giống nhau
- Sau khi giao hoán thứ tự các giao tác thì không xảy ra xung đột.

Nếu cả hai giao dịch đều xử lý các mục dữ liệu khác nhau, thì sẽ không có xung đột nên chúng ta có thể hoán đổi chúng để đạt được lịch giao dịch tuần tự. Ví dụ, nếu T1 đang đọc giá trị của mục dữ liệu A và T2 đang đọc giá trị của mục dữ liệu B, vì cả hai đều là khác nhau, chúng ta có thể hoán đổi chúng.

T1	T2	T1	T2
R(X)		R(X)	
W(X)		W(X)	
R(Y)			R(Y)
W(Y)			W(Y)
	R(X)	R(Y)	
	W(X)	W(Y)	
	R(Y)		R(X)
	W(Y)		W(X)

Lịch S

Lịch S'

Phần 2. Giới thiệu về lịch giao tác (Transaction Schedule)



- **Các trường hợp gây mất tính nhất quán dữ liệu:**

- Nếu cả hai giao tác T_i và T_j đang xử lý cùng một dữ liệu X , chúng phải kiểm tra chúng xung đột các trường hợp sau:

- Nếu T_i thực hiện $R(X)$ và T_j thực hiện $R(X)$, thì không có xung đột vì cả hai đang thao tác đọc
- Nếu T_i thực hiện $R(X)$ và T_j thực hiện $W(X)$, thì có xung đột
- Nếu T_i thực hiện $W(X)$ và T_j thực hiện $R(X)$, thì có xung đột
- Nếu T_i thực hiện $W(X)$ và T_j thực hiện $W(X)$, thì có xung đột

- Tóm lại trường hợp có xung đột xảy ra **khi và chỉ khi** có ít nhất một **giao tác ghi** trên cùng dữ liệu.



PHẦN 3

ĐỒ THỊ CHỜ VÀ PHƯƠNG PHÁP KHẢ TUẦN TỰ ¹⁷

Đồ thị chờ là một công cụ để phân tích tính tuần tự hóa trong lịch giao dịch đồng thời. Phương pháp tuần tự hóa đồ thị sử dụng loại bỏ các chu trình trong đồ thị chờ để xác định tính tuần tự hóa của lịch.



Phần 3. Đồ thị chờ và phương pháp khả tuần tự



• Đồ thị chờ

– Đồ thị chờ là một công cụ để phân tích tính tuần tự hóa trong lịch giao dịch đồng thời. Phương pháp tuần tự hóa đồ thị sử dụng loại bỏ các chu trình trong đồ thị chờ để xác định tính tuần tự hóa của lịch.

- Nếu T_i thực hiện $R(X)$ và T_j thực hiện $R(X)$, thì không có xung đột vì cả hai đang thao tác đọc
- Nếu T_i thực hiện $R(X)$ và T_j thực hiện $W(X)$, thì có xung đột
- Nếu T_i thực hiện $W(X)$ và T_j thực hiện $R(X)$, thì có xung đột
- Nếu T_i thực hiện $W(X)$ và T_j thực hiện $W(X)$, thì có xung đột

– Đồ thị chờ là đồ thị **có hướng** có:

- Đỉnh là các giao tác (Transaction)
- Cạnh từ T_i đến T_j nếu tồn tại xung đột khi có $W(X)$ trên T_i , T_j (T_i trước T_j)
- Cạnh từ T_j đến T_i nếu tồn tại xung đột khi có $W(X)$ trên T_j , T_i (T_j trước T_i)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

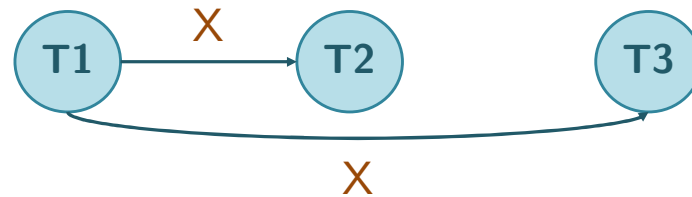
- Ví dụ về đồ thị chờ

Ta có ...W1(X)...R2(X)... Vẽ từ T1 \rightarrow T2 trên đơn vị

Dữ liệu X

Ta có ...W1(X)... R3(X)... Vẽ cung từ T1 \rightarrow T3 trên đơn

Vị dữ liệu X



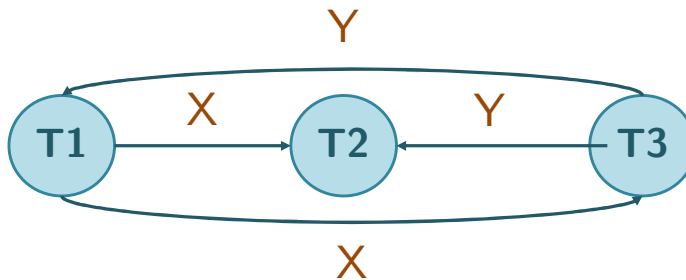
T1	T2	T3
R(X) W(X)	R(X) W(X)	R(Y) W(Y)
R(Y) W(Y)	R(Y) W(Y)	R(X) W(X)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

- Ví dụ về đồ thị chờ

Ta có ...W3(Y) ... R2(Y)... vẽ đồ thị từ $T3 \rightarrow T2$ trên Y

Ta có ...W3(Y) ... R1(Y) vẽ đồ thị từ $T3 \rightarrow T1$ trên Y.



T1	T2	T3
R(X) W(X)	R(X) W(X)	R(Y) W(Y)
R(Y) W(Y)	R(Y) W(Y)	
		R(X) W(X)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

• Lịch khả tuần tự Conflict Serializable, View Serializable

- **Lịch khả tuần tự theo Conflict Serializable:** Khi lịch S hoán vị và tương đương với một lịch tuần tự S', thì lịch đó gọi là lịch Conflict Serializable
- Đánh giá đồ thị chờ **KHÔNG** có chu trình → Lịch khả tuần tự Conflict

T1	T2	T1	T2
R(X)		R(X)	
W(X)		W(X)	
R(Y)			R(Y)
W(Y)			W(Y)
	R(X)	R(Y)	
	W(X)	W(Y)	
	R(Y)		R(X)
	W(Y)		W(X)

Lịch S

Lịch S'

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

• Lịch khả tuần tự Conflict Serializable, View Serializable

– **Lịch View-Equivalent:** Hai lịch S và S' được gọi là View-Equivalent khi và chỉ khi:

- Nếu S có $W_j(A) \rightarrow R_i(A)$ thì trong S' cũng có $W_j(A) \rightarrow R_i(A)$
- Nếu S kết thúc bằng $W_i(A)$ thì S' cũng kết thúc bằng $W_i(A)$
- Nếu S bắt đầu bằng $R_i(A)$ thì S' cũng bắt đầu bằng $R_i(A)$

Xét lịch S và S':

- Điều kiện 1 cả hai lịch đều không có $W_j(A) \rightarrow R_i(A)$ nên cả hai đều thỏa
- Điều kiện 2 lịch S kết thúc bằng $W_1(A)$ và S' kết thúc bằng $W_1(A)$, nên thỏa điều kiện 2
- Điều kiện 3 lịch S bắt đầu $R_1(A)$ và S' cũng bắt đầu bằng $R_1(A)$, nên thỏa điều kiện 2

→ Nên lịch S và S' tương đương theo View

S		
T1	T2	T3
R(A)		
W(A)	W(A)	
		W(A)

S'		
T1	T2	T3
R(A)		
W(A)		
	W(A)	
		W(A)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

• Lịch khả tuần tự Conflict Serializable, View Serializable

- **Lịch khả tuần tự theo View Serializable:** Lịch S được gọi là View Serializable khi tồn tại lịch S' **tuần tự** tương đương với S theo chuẩn **View-Equivalent**.

Xét lịch S và S':

- S và S' tương đương theo View
- S' tuần tự

→ Nên S khả tuần tự theo View

S		
T1	T2	T3
R(A)	W(A)	
W(A)		
		W(A)

S'		
T1	T2	T3
R(A)		
W(A)		
	W(A)	
		W(A)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự



• Cách xác định lịch khả tuần tự theo View bằng đồ thị chờ

Cho lịch S có 3 Transaction T1, T2, T3.

- **Bước 1:** Thêm Tb và Tf vào lịch S
 - Bước 1.1 Thêm Tb vào đầu lịch S sao cho Tb ghi hết tất cả dữ liệu khởi tạo ban đầu
 - Bước 1.2 Thêm Tf vào cuối lịch S sao cho Tf đọc hết tất cả đơn vị dữ liệu trên S

S		
T1	T2	T3
W(A)	R(B) W(A)	W(A) W(B)
W(B)		

S				
Tb	T1	T2	T3	Tf
W(A) W(B)	W(A) W(B)	R(B) W(A)	W(A) W(B)	R(A) R(B)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự



• Cách xác định lịch khả tuần tự theo View bằng đồ thị chờ

Cho lịch S có 3 Transaction T1, T2, T3.

- **Bước 2:** Vẽ đồ thị Phức (Poly Graph) được định nghĩa như sau:
 - **Đỉnh** là các Transaction T_b, T_f, T_i ($i=1,2,3,\dots$)
 - **Cạnh**
 - Nếu $\dots W_i(A) \dots R_j(A) \dots$ thì vẽ cung từ $T_i \rightarrow T_j$ trên đơn vị dữ liệu A (1)
 - Xét từng cặp $\dots W_i(A) \dots R_j(A)$ tại (1) và T_k nằm trước T_i và sau T_j , giao tác $W_k(A)$
 - Nếu $T_b = T_i$ và $T_j \neq T_f$, chèn cung $T_j \rightarrow T_k$
 - Nếu $T_b \neq T_i$ và $T_j = T_f$, chèn cung $T_k \rightarrow T_i$
 - Nếu $T_b \neq T_i$ và $T_j \neq T_f$, chèn cung $T_k \rightarrow T_i$ và $T_j \rightarrow T_k$

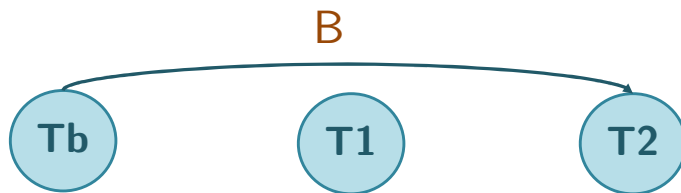
S				
Tb	T1	T2	T3	Tf
W(A) W(B)	W(A)	R(B) W(A)	W(A) W(B)	R(A) R(B)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

• Cách xác định lịch khả tuần tự theo View bằng đồ thị chờ

Cho lịch S có 3 Transaction T1, T2, T3.

- Ta có $Wb(B) \dots R2(B) \dots$ nên $Tb \rightarrow T2$ trên đơn vị dữ liệu B
- Ta có $\dots W3(A) \dots Rf(A) \dots$ nên $T3 \rightarrow Tf$ trên đơn vị dữ liệu A
- Ta có $\dots W3(B) \dots Rf(B) \dots$ nên $T3 \rightarrow Tf$ trên đơn vị dữ liệu B



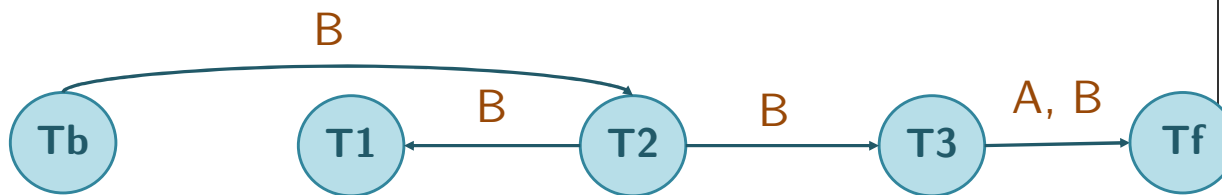
S				
Tb	T1	T2	T3	Tf
W(A) W(B)	W(A) W(B)	R(B) W(A)	W(A) W(B)	R(A) R(B)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

• Cách xác định lịch khả tuần tự theo View bằng đồ thị chờ

Cho lịch S có 3 Transaction T1, T2, T3.

- Xét $T_b \rightarrow T_2$ trên đơn vị dữ liệu B
 - Ta có $T_k = T_3$ có $W_3(B)$ và $T_i = T_b, T_j \neq T_f$ ($T_j = T_2$) nên vẽ cung từ $T_2 \rightarrow T_3$
 - Ta có $T_k = T_1$ có $W_1(B)$ và $T_i = T_b, T_j \neq T_f$ ($T_j = T_1$) nên vẽ cung từ $T_2 \rightarrow T_1$



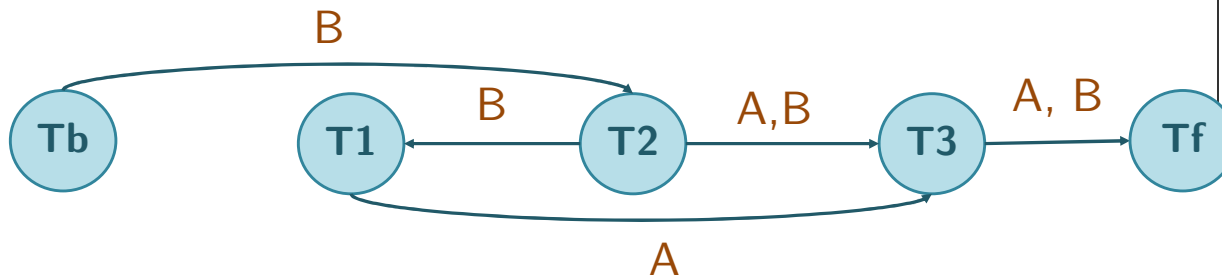
S				
Tb	T1	T2	T3	Tf
W(A) W(B)	W(A) W(B)	R(B) W(A)	W(A) W(B)	R(A) R(B)

Phần 3. Đồ thị chờ và phương pháp khả tuần tự

• Cách xác định lịch khả tuần tự theo View bằng đồ thị chờ

Cho lịch S có 3 Transaction T1, T2, T3.

- Xét cặp $T3 \rightarrow T_f$ trên đơn vị dữ liệu A:
 - Ta có $T_k = T1$ có $W1(A)$ trước $T3$ và $T_i \neq T_b$ ($T_i = T_k$) và $T_j = T_f$ nên vẽ từ $T1 \rightarrow T3$
 - Ta có $T_k = T2$ có $W2(A)$ trước $T3$ và $T_i \neq T_b$ ($T_i = T_k$) và $T_j = T_f$ nên vẽ từ $T2 \rightarrow T3$



S				
Tb	T1	T2	T3	Tf
W(A) W(B)	W(A)	R(B) W(A)	W(A) W(B)	R(A) R(B)

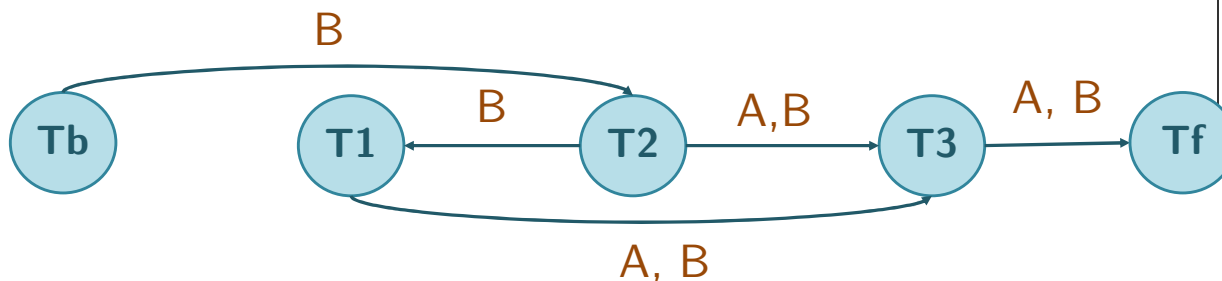
Phần 3. Đồ thị chờ và phương pháp khả tuần tự

• Cách xác định lịch khả tuần tự theo View bằng đồ thị chờ

Cho lịch S có 3 Transaction T1, T2, T3.

- Xét cặp $T3 \rightarrow T_f$ trên đơn vị dữ liệu B
 - Ta có $T_k = T1$ có $W1(B)$ trước $T3$ à $T_i \neq T_b$ ($T_i = T_k$) và $T_j = T_f$ nên vẽ từ $T1 \rightarrow T3$
- Đồ thị KHÔNG CÓ chu trình \rightarrow Lịch khả tuần tự theo View theo thứ tự T2; T1; T3

Nhận xét: Một lịch khả tuần tự theo Conflict \Rightarrow Lịch khả tuần tự theo View



S				
Tb	T1	T2	T3	Tf
W(A) W(B)	W(A) W(B)	R(B) W(A)	W(A) W(B)	R(A) R(B)

Bài tập Chương 3

• Bài tập Chương 3

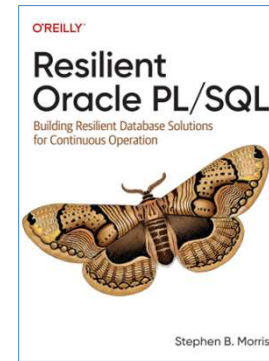
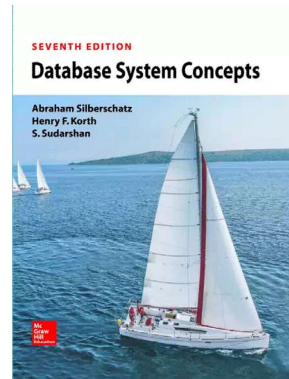
Cho các lịch S sau đây:

1. S: $r_2(B)$ $w_2(A)$ $r_1(A)$ $r_3(A)$ $w_1(B)$ $w_2(B)$ $w_3(B)$
2. S: $w_1(A)$ $r_3(A)$ $r_2(A)$ $w_2(A)$ $r_1(A)$ $w_3(A)$
3. S: $r_2(A)$ $r_1(A)$ $w_1(C)$ $r_3(C)$ $w_1(B)$ $r_4(B)$ $w_3(A)$ $r_4(C)$ $w_2(D)$ $r_2(B)$ $w_4(A)$ $w_4(B)$
4. S: $w_1(A)$ $r_2(A)$ $w_2(A)$ $r_1(A)$
5. S: $r_1(A)$ $r_3(D)$ $w_1(B)$ $r_2(B)$ $w_3(B)$ $r_4(B)$ $w_2(C)$ $r_5(C)$ $w_4(E)$ $r_5(E)$ $w_5(B)$
6. S: $w_1(A)$ $r_2(A)$ $w_3(A)$ $r_4(A)$ $w_5(A)$ $r_6(A)$
7. S: $r_1(X)$ $r_2(X)$ $w_1(X)$ $w_2(X)$

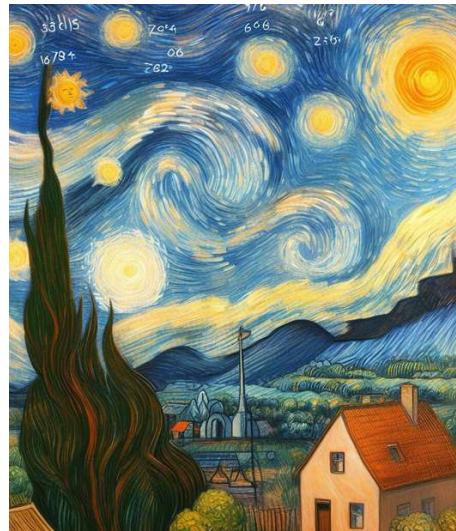
Yêu cầu:

1. Lịch nào khả tuần tự theo Conflict. Giải thích
2. Lịch nào khả tuần tự theo View. Giải thích

Reference



- [1] Abraham Silberschatz, Henry F. Korth, S. Sudarshan, Database System Concepts, 7th Edition, 2020.
- [2] Stephen B. Maris, "Resilient Oracle PL/SQL 1st Edition", O'Reilly, 2023



CẢM ƠN ĐÃ THEO DÕI



ftisu.vn



minhnhut.ftisu@gmail.com



0939013911