**CÂU HỎI ÔN TẬP CHƯƠNG 4**

**1. Transaction là gì?**

Trả lời: Một Giao dịch (Transaction) là một thực thi của một chương trình người dùng, và được DBMS xem như một chuỗi các thao tác. Các thao tác này có thể được thực hiện bằng một giao dịch nào đó bao gồm: các phép đọc và viết lên các đối tượng cơ sở dữ liệu, trong khi đó các thao tác trong một chương trình thông thường có thể bao gồm dữ liệu đầu vào của người dùng, truy cập các thiết bị mạng, xây dựng giao diện người dùng, …

**2. Transaction giống hay Transaction khác với một chương trình được thực thi bên ngoài DBMS?**

Trả lời: Transaction khác mới một chương trình được thực thi bên ngoài DBMS.

**3. Phát biểu nào dưới đây đúng**

(1)DBMS xử lý transaction như một loạt các thao tác đọc ghi các đối tượng CSDL

(2)Thao tác đọc một đối tượng CSDL là đưa đối tượng đó từ đĩa vào bộ nhớ (buffer bool), sau đó copy giá trị của nó vào biến trong chương trình.

(3)Thao tác ghi một đối tượng CSDL là ghi bản sao của đối tượng trong bộ nhớ lên đĩa.

**4. Nêu 4 tính chất quan trọng của transaction? Cho ví dụ**

Trả lời:

- **Atomic (Nguyên tử)**: Tất cả các thao tác trong 1 Transaction được thực hiện thành công hoặc thất bại hoàn toàn. Người dùng không phải lo lắng về ảnh hưởng của các Transaction chưa được thành công (Giả sử có sự cố xảy ra trong quá trình Transaction thực hiện).

Ví dụ: Khi chuyển tiền từ A -> B. Nếu giao dịch không thành công trongmột hoặc nhiều bước trong Transaction, hệ thống phải đảm bảo rằng các bản cập nhật của nó không được phản ánh trong cơ sở dữ liệu, nếu không sẽ dẫn đến sự không nhất quán.

- **Consistent (Nhất quán)** : Mỗi giao dịch được thực thi không tranh chấpvới các giao dịch khác, phải đảm bảo tính chất nhất quán của cơ sở dữ liệu. DBMS thừa nhận rằng tính nhất quán được đảm bảo trên mỗi giao dịch. Việc đảm bảo tính chất này của giao dịch là trách nhiệm của người dùng.

Ví dụ: Khi chuyền tiền từ A 🡪B. Thì tổng tiền trong 2 tài khoản A và B sau khi chuyển và trước khi chuyển phải bằng nhau.

**- Isolated (Cô lập):** Một Transaction khi được người dùng thực hiện thì nó thực hiện như thể là chỉ có một Transaction được thực thi trong 1 thời điểm. Nhưng thực tế thì trong 1 thời điểm có thể có nhiều Transaction cùng được thực hiện.

Ví dụ: Khi chuyển tiền từ A🡪B. Nếu giữa các bước 3 và 6, một giao dịch khác được phép truy cập vào CSDL được cập nhật một phần, nó sẽ thấy một CSDL không nhất quán (tổng A + B sẽ nhỏ hơn mức cần thiết). Có thể được đảm bảo một cách nhỏ nhặt bằng cách chạy các giao dịch nối tiếp nhau, cái này đến cái kia. Tuy nhiên, thực hiện đồng thời nhiều giao dịch có những lợi ích đáng kể, như chúng ta sẽ thấy.

- **Durable (Bền vững):** Khi DBMS thông báo cho người dùng biết rằng giao dịch đã thành công hoàn toàn, những ảnh hưởng của nó nên được duy trì ngay cả khi hệ thống gặp sự cố trước khi tất cả những thay đổi này kịp lưu lại trên đĩa.

Ví dụ: Khi chuyển tiền từ A 🡪 B. Sau khi người dùng được thông báo rằng giao dịch đã hoàn tất (tức là quá trình chuyển tiền đã diễn ra), các cập nhật cho cơ sở dữ liệu của giao dịch phải vẫn tồn tại bất chấp lỗi.

**5. Cho 1 chương trình gồm các hành động theo thứ tự sau: (B1) Read(A); (B2)A = A – 50; (B3) Write(A); (B4) Read(B); (B5) B = B + 50; (B6) Write(B). Phát biểu nào dưới đây mô tả tính nguyên tố, tính nhất quán, tính cô lập, tính bền vững**

(1)Tổng A và B không được thay đổi bởi sự thực thi chương trình – Tính nhất quán.

(2)Nếu transaction không thành công sau bước 3 và trước bước 6, DBMS phải đảm bảo là kết quả các thao tác cập nhật của transaction không được ghi lên CSDL. – Tính nguyên tử.

(3)Khi người dùng đã được thông báo là đã chuyển 50$ sang B thành công thì các cập nhật lên CSDL bởi transaction phải vẫn tồn tại dù hệ thống có gặp sự cố. –Tính bền vững.

(4)Hai người thực thi chương trình trên đồng thời thì mỗi người vẫn thấy như chỉ có một mình chạy chương trình đó. – Tính cô lập

**6. Hệ thống cho phép nhiều transaction chạy đồng thời để:**

Ví dụ: Khi chuyền tiền từ A 🡪 B. Thì tổng tiền trong 2 tài khoản A và B sau khi chuyển và trước khi chuyển phải bằng nhau.

- **Isolated (Cô lập)**: Một Transaction khi được người dùng thực hiện thì nó thực hiện như thể là chỉ có một Transaction được thực thi trong 1 thời điểm. Nhưng thực tế thì trong 1 thời điểm có thể có nhiều Transaction cùng được thực hiện.

Ví dụ: Khi chuyển tiền từ A 🡪 B. Nếu giữa các bước 3 và 6, một giao dịch khác được phép truy cập vào CSDL được cập nhật một phần, nó sẽ thấy một CSDL không nhất quán (tổng A + B sẽ nhỏ hơn mức cần thiết). Có thể được đảm bảo một cách nhỏ nhặt bằng cách chạy các giao dịch nối tiếp nhau, cái này đến cái kia. Tuy nhiên, thực hiện đồng thời nhiều giao dịch có những lợi ích đáng kể, như chúng ta sẽ thấy.

- **Durable (Bền vững):** Khi DBMS thông báo cho người dùng biết rằng giao dịch đã thành công hoàn toàn, những ảnh hưởng của nó nên được duy trì ngay cả khi hệ thống gặp sự cố trước khi tất cả những thay đổi này kịp lưu lại trên đĩa.

Ví dụ: Khi chuyển tiền từ A 🡪 B. Sau khi người dùng được thông báo rằng giao dịch đã hoàn tất (tức là quá trình chuyển tiền đã diễn ra), các cập nhật cho cơ sở dữ liệu của giao dịch phải vẫn tồn tại bất chấp lỗi.

**7. Cho 1 chương trình gồm các hành động theo thứ tự sau: (B1) Read(A); (B2)A = A – 50; (B3) Write(A); (B4) Read(B); (B5) B = B + 50; (B6) Write(B). Phát biểu nào dưới đây mô tả tính nguyên tố, tính nhất quán, tính cô lập, tính bền vững.**

(1)Tổng A và B không được thay đổi bởi sự thực thi chương trình – Tính nhất quán.

(2)Nếu transaction không thành công sau bước 3 và trước bước 6, DBMS phải đảm bảo là kết quả các thao tác cập nhật của transaction không được ghi lên CSDL. – Tính nguyên tử.

(3)Khi người dùng đã được thông báo là đã chuyển 50$ sang B thành công thì các cập nhật lên CSDL bởi transaction phải vẫn tồn tại dù hệ thống có gặp sự cố. – Tính bền vững.

(4)Hai người thực thi chương trình trên đồng thời thì mỗi người vẫn thấy như chỉ có một mình chạy chương trình đó. – Tính cô lập

**6. Hệ thống cho phép nhiều transaction chạy đồng thời để:**

(1)Tăng khả năng sử dụng bộ xử lý và ổ đĩa: một transaction có thể đang sử dụng CPU trong khi một giao dịch khác đang đọc hoặc ghi vào đĩa.

(2)Giảm thời gian phản hồi trung bình của các transaction: các transaction ngắn không cần chờ đợi sau các transaction dài

**7. Phát biểu nào dưới đây đúng về Lịch biểu**

(1)Là một lịch trình thực thi các câu lệnh trong các transaction khác nhau theo thời gian

(2)Một lịch biểu cho một tập các transaction phải bao gồm tất cả các câu lệnh trong các transaction đó.

(3)Lịch biểu phải duy trì thứ tự thực hiện các câu lệnh trong một transaction riêng biệt.

**8. Lịch biểu tuần tự là gì?**

Trả lời: Lịch biểu tuần tự (Serial schedule): là lịch biểu chỉ cho phép thực hiện xong Transaction này, rồi mới tiếp tục thực hiện Transaction khác.

**9. Cho 2 lịch biểu L1 và L2. L1 được nói là tương đương với L2 nếu**

(1) Tác động của việc thực thi L1 lên CSDL giống với tác động của việc thực thi L2 lên CSDL

(2) Kết quả của việc thực thi L1 giống với kết quả thực thi L2

**10. Lịch biểu khả tuần tự?**

Trả lời: Lịch biểu khả tuần tự (Serializable schedule): là 1 lịch biểu tương đương với 1 lịch biểu tuần tự nào đó.

**11. Cho 2 transaction Ti và Tj. Hành động Li thuộc Ti và Lj thuộc Tj. Phát biểu nào dưới đây đúng**

(1) Li và Lj là 2 hành động xung đột nếu Li và Lj cùng truy cập đến một đối tượng CSDL

(2) Li và Lj là 2 hành động xung đột nếu ít nhất một trong 2 hành động Li và Lj là hành động ghi

(3) Li và Lj là 2 hành động xung đột nếu Li và Lj cùng truy cập đến một đối tượng CSDL và ít nhất một trong 2 hành động Li và Lj là hành động ghi

**12. Lịch biểu L1 có 2 transaction gồm các hành động theo thứ tự sau T1: R(X), T1:R(Y), T2:R(X), T1:W(X), T2:W(X), T1:W(Y). Phát biểu nào dưới đây đúng**

(1)T1: R(X) và T2:R(X) là 2 hành động không xung đột

(2)T2:R(X) và T1:W(X) là 2 hành động xung đột

(3)T2:W(X) và T1:W(Y) là 2 hành động không xung đột

**13. Phát biểu nào dưới đây đúng**

(1)Nếu đổi thứ tự thực hiện 2 hành động xung đột thì kết quả của lịch biểu sẽ thay đổi

(2)Nếu đổi thứ tự thực hiện 2 hành động không xung đột thì kết quả của lịch biểu sẽ không thay đổi

**14. Định nghĩa 2 lịch biểu L1 và L2 được gọi là tương đương xung đột**

Trả lời: L1 và L2 được gọi là tương đương xung đột nếu L2 là L1 sau khi thực hiện một loạt các lệnh hoán đổi không xung đột. Hoặc ngược lại.

**15. Định nghĩa lịch biểu khả tuần tự xung đột**

Trả lời: Lịch biểu khả tuần tự xung đột (Conflict Serializable Schedule): là lịch biểu tương đương xung đột với một lịch biểu tuần tự.

**16. Lịch biểu T1:R(X), T2:R(Y), T1: R(Y), T2:W(Y), T2:R(X), T1:W(X), T1:W(Y), T2:W(X) tương đương xung đột với lịch biểu nào dưới đây:**

(1)T1:R(X), T2:R(Y), T1: R(Y), T2:W(Y), T2:R(X), T1:W(X), T2:W(X), T1:W(Y)

(2)T2:R(Y), T1:R(X), T1: R(Y), T2:W(Y), T2:R(X), T1:W(X), T1:W(Y), T2:W(X)

**17. Cho lịch biểu T1:R(X), T2:R(Y), T1: R(Y), T2:W(Y), T2:R(X), T1:W(X), T1:W(Y), T2:W(X). Lịch biểu trên là lịch biểu gì?**

Trả lời: Lịch biểu xung đột – RW conflict

|  |  |
| --- | --- |
| T1 | T2 |
| Rx |  |
|  | Ry |
| Ry |  |
|  | Wy |
|  | Rx |
| Wx |  |
| Wy |  |

**18. Cho lịch biểu T1:R(X), T2:R(Y), T1: W(X), T2:R(X), T2:W(X), T2:W(Y)**

|  |  |
| --- | --- |
| **T1** | **T2** |
| **Rx** |  |
|  | **Ry** |
| **Wx** |  |
|  | **Rx** |
|  | **Wx** |
|  | **Wy** |

Phát biểu nào dưới đây đúng về lịch biểu này

(1) Lịch biểu này khả tuần tự xung đột

(2) Lịch biểu này khả tuần tự

**19. Lịch biểu T1:R(X), T2:R(Y), T1: W(X), T2:R(X), T2:W(X), T2:W(Y) tương đương xung đột với lịch biểu nào dưới đây**

(1)T1:R(X), T1: W(X), T2:R(Y), T2:R(X), T2:W(X), T2:W(Y)

(2)T2:R(Y), T1:R(X), T1: W(X), T2:R(X), T2:W(X), T2:W(Y)

**20. Định nghĩa 2 Lịch biểu tương đương view**

Trả lời: Hai lịch biểu được gọi là tương đương view (View Equivalent) nếu đạt **ba điều kiện sau:**

1. Với mỗi đối tượng Q, nếu Tracsaction Ti đọc giá trị ban đầu của Q trong lịch biểu S thì Transantion Ti trong lịch biểu S’ cũng phải đọc giá trị ban đầu của Q.

2. Với mỗi đối tượng Q, nếu Transaction Ti thực hiện đọc Q trong lịch biểu S, mà giá trị Q này trước đó được ghi bởi Transaction Tj (nếu có) thì Transaction Ti trong lịch biểu S’ cũng phải đọc giá trị Q, mà giá trị Q trước đó cũng được ghi bởi Transaction Tj của S’.

3. Với mỗi đối tượng Q, nếu Transaction thực hiện việc ghi giá trị Q cuối cùng (nếu có) trong lịch biểu S, thì Transaction đó cũng phải thực hiện việc ghi giá trị Q cuối cùng trong lịch biểu S’.

**21. Định nghĩa lịch biểu khả tuần tự view**

Trả lời: Lịch biểu khả tuần tự view (View Serializable): nếu lịch biểu này tương đương view với lịch biểu tuần tự.

**22. Giải thích các khái niệm ghi mù (blind write), đọc bẩn (dirty read)**

Trả lời:

- Blind Write (Ghi mù): Là việc một Transaction thực hiện ghi lên một đối tượng khi chưa đọc đối tượng đó.

- Dirty Read (Đọc bẩn): Xảy ra khi một Transaction đọc một đối tượng mà đối tượng này đang được thay đổi giá trị bằng một Transaction chưa thành công khác.

**23. Lịch biểu: T1:R(X), T2:R(Y), T1:W(X), T2:W(Y) tương đương view với lịch biểu nào dưới đây**

(1)T1:R(X), T1:W(X), T2:R(Y), T2:W(Y)

(2)T1:R(X), T2:R(Y), T2:W(Y), T1:W(X)

(3)T1:R(Y), T2:R(X), T1:W(X), T2:W(Y)

**24. Lịch biểu: T1:R(X), T2:R(Y), T1:W(X), T2:W(Y) tương đương xung đột với lịch biểu nào dưới đây**

(1)T1:R(X), T1:W(X), T2:R(Y), T2:W(Y)

(2)T1:R(X), T2:R(Y), T2:W(Y), T1:W(X)

(3)T1:R(Y), T2:R(X), T1:W(X), T2:W(Y)

**25. Lịch biểu nào dưới đây có hành động ghi mù**

(1)T1:R(X), T1:W(X), T2:R(Y), T2:W(Y)

(2)T1:R(X), T2:R(Y), T2:W(Y), T1:W(X)

(3)T1:R(Y), T2:R(X), T1:W(X), T2:W(X)

**26. Định nghĩa lịch biểu được gọi là khả phục hồi**

Trả lời: Lịch biểu được gọi là khả phục hồi (Recoverable Schedule): nếu 1 Transaction Tj đọc 1 đối tượng dữ liệu mà trước đó đối tượng này được ghi bởi Transaction Ti thì thao tác Commit của Ti phải xuất hiện trước thao tác Commit của Tj.

**27. Lịch biểu nào dưới đây không khả phục hồi**

(1)T1:R(X), T1:W(X), T2:R(Y), T2:W(Y), T1: commit, T2: commit

(2)T1:R(X), T2:R(X), T1: W(X), T2:W(X), T1: commit, T2: commit

(3)T1:R(X), T1:W(X), T2:R(X), T2:W(X), T2: commit, T1: commit

(4)T1:R(X), T1:W(X), T2:R(X), T2:W(X), T1: commit, T2: commit

**28. Phát biểu nào dưới đây đúng**

(1) Trong 1 lịch biểu khi 1 transaction thực hiện không thành công dẫn đến phải phục hồi lại các transaction khác thì lịch biểu đó được gọi là khả phục hồi lan truyền

(2) Phục hồi lan truyền có thể dẫn đến phải phục hồi một lượng đáng kể công việc.

(3) Cần phải tránh việc phục hồi lan truyền.

**29. Lịch biểu nào dưới đây dẫn đến việc phục hồi lan truyền**

(1)T1:R(X), T1:W(X), T2:R(X), T2:W(X), T1: commit, T2: commit

(2)T1:R(X), T1:W(X), T2:R(Y), T2:W(Y), T1: commit, T2: commit

(3)T1:R(X), T1:W(X), T2:R(X), T2:W(X), T2: commit, T1: commit

(4)T1:R(X), T2:R(X), T1: W(X), T2:W(X), T1: commit, T2: commit

**30. Một lịch biểu là khả phục hồi không lan truyền nếu**

(1)Transaction Tj đọc đối tượng X mà trước đó X được ghi bởi Ti thì hành động commit của Ti phải thực hiện trước hành động đọc X của Tj

(2)Transaction Tj đọc đối tượng X mà trước đó X được ghi bởi Ti thì hành động commit của Tj phải thực hiện trước commit của Ti

**31. Lịch biểu nào dưới đây dẫn đến việc phục hồi không lan truyền**

(1)T1:R(X), T1:W(X), T2:R(X), T2:W(X), T1: commit, T2: commit

(2)T1:R(X), T1:W(X), T1: commit, T2:R(X), T2:W(X), T2: commit

(3)T1:R(X), T1:W(X), T2:R(X), T1: commit, T2:W(X), T2: commit

**32. Phát biểu nào dưới đây đúng với nghi thức khóa 2 giai đoạn (2 phase locking)**

(1)Dùng để quan lý sự truy cập đồng thời

(2)Khi một transaction muốn đọc (hoặc chỉnh sửa) một đối tượng, đầu tiên nó

phải yêu cầu share lock (hoặc exclusive lock) trên đối tượng đó,

(3)Chỉ khi nhận được lock transaction mới đọc (hoặc chỉnh sửa đối tượng)

**33. Phát biểu nào dưới đây đúng**

(1)Trong nghi thức khóa 2 giai đoạn nghiêm ngặt, tất cả khóa được giữ bởi 1 transaction sẽ được giải phóng khi transaction đó đã hoàn thành

(2)Trong nghi thức khóa 2 giai đoạn không nghiêm ngặt, các khóa được giữ bởi 1 transaction có thể được giải phóng bất kỳ lúc nào

**34. Phát biểu nào dưới đây đúng**

(1)Trong SQL Server có 3 loại transaction: tự động commit, tường minh và không tường minh

(2)Trong SQL Server có 2 loại transaction: Tường minh và không tường minh

(3)Trong SQL Server chỉ có 1 loại transaction duy nhất

**35. Phát biểu nào dưới đây đúng về Autocommit transaction**

(1)Là transaction mặc định

(2)Mỗi câu lệnh T-SQL được xem là 1 transaction. Chúng được commit hay rollback tùy vào kết quả của chúng.

**36. Phát biểu nào dưới đây đúng về Implicit transaction**

(1)Được kích hoạt bằng câu lệnh SET IMPLICIT\_TRANSACTIONS ON

(2)Khi đã được kích hoạt, SQl server sẽ bắt đầu transaction bất cứ khi nào gặp câu lệnh thao tác dữ liệu (DML-Data Manipulation Language)

(3)Phải dùng câu lệnh commit hoặc rollback tường minh ở cuối transaction

**37. Phát biểu nào dưới đây đúng về Explicit transaction**

(1)Bắt đầu bằng Begin transaction

(2)Dùng lệnh commit transaction để kết thúc transaction thành công

(3)Dùng Rollback transaction để hủy bỏ transaction

BÀI TẬP TRANSACTION

Câu 1. Trả lời các câu hỏi sau:

1. Transaction là gì? Nó khác gì khi so với một chương trình thông thường (chẳng hạn một chương trình viết bằng ngôn ngữ C)

Một Giao dịch (Transaction) là một thực thi của một chương trình người dùng, và được DBMS xem như một chuỗi các thao tác. Các thao tác này có thể được thực hiện bằng một giao dịch nào đó bao gồm: các phép đọc và viết lên các đối tượng cơ sở dữ liệu, trong khi đó các thao tác trong một chương trình thông thường có thể bao gồm dữ liệu đầu vào của người dùng, truy cập các thiết bị mạng, xây dựng giao diện người dùng, vv.

1. Định nghĩa các thuật ngữ sau: atomicity, consistency, isolation, durability, schedule, blide write, dirty read, unrepeatable read, serializable schedule, recoverable schedule, avoids-cascading-aborts schedule.

* Nguyên tử (atomicity): Tất cả các thao tác nằm trong giao dịch được thực hiện thành công hoặc thất bại hoàn toàn. Người dùng không phải lo lắng về ảnh hưởng của các giao dịch chưa được thành công (giả sử, có sự cố xảy ra khi giao dịch này đang trong quá trình thực hiện)
* Nhất quán (consistency): Mỗi giao dịch được thực thi không tranh chấp với các giao dịch khác, phải đảm bảo tính chất nhất quán của cơ sở dữ liệu. DBMS thừa nhận rằng tính nhất quán được đảm bảo trên mỗi giao dịch. Việc đảm bảo tính chất này của giao dịch là trách nhiệm của người dùng.
* Cô lập (isolation): Người dùng nên có thể hiểu được một giao dịch mà không cần xem xét những ảnh hưởng của các giao dịch tương tranh khác đang chạy, thậm chí DBMS có thể chèn vào các thao tác khác vì những lý do thực thi. Tính chất này đôi khi được nói tới như là tính chất cô lập.Các giao dịch được cô lập, hay còn gọi là được bảo vệ từ những ảnh hưởng của các giao dịch tương tranh khác.
* Bền vững (durability): Khi DBMS thông báo cho người dùng biết rằng giao dịch đã thành công hoàn toàn, những ảnh hưởng của nó nên được duy trì ngay cả khi hệ thống gặp sự cố trước khi tất cả những thay đổi này kịp lưu lại trên đĩa.
* Lịch trình (schedule): là một chuỗi các giao dịch (có thể xếp chồng).
* Viết mù (blind write ): là việc một giao dịch nào đó viết lên một đối tượng mà thậm chí không đọc đối tượng này.
* Đọc bẩn (dirty read ): xảy ra khi một giao dịch nào đó đọc một đối tượng mà đối tượng này đang được thay đổi bằng một giao dịch chưa thành công khác.
* Đọc không thể lặp lại (unrepeatable read ): xảy ra khi một giao dịch nào đó không thể đọc cùng một giá trị đối tượng nhiều hơn một lần, thậm chí giao dịch này không được phép thay đổi giá trị. Giả sử giao dịch T2 thay đổi giá trị của đối tượng A – đối tượng đang được đọc bằng một giao dịch T1 trong khi T1 vẫn đang trong quá trình xử lý. Nếu T1 cố gắng đọc giá trị A một lần nữa, nó sẽ có một kết quả khác, mặc dù nó không thay đổi A.
* Lịch trình tuần tự (serializable schedule ): trên một tập S của các giao dịch là một lịch trình mà đều giống với lịch trình tuần tự hoàn toàn trên tập các giao dịch thành công trong S.
* Lịch trình phục hồi (recoverable schedule): là một lịch trình mà trong đó một giao dịch có thể được thành công chỉ sau khi tất cả các giao dịch khác đọc nó đã thành công.
* Lịch trình tránh hủy bỏ chồng (avoids-cascading-aborts schedule ): là một trong số các giao dịch chỉ đọc những thay đổi của các giao dịch đã thành công. Một lịch trình như vậy không chỉ có khả năng phục hồi, việc hủy bỏ một giao dịch có thể được hoàn thành mà không hủy bỏ chồng các giao dịch khác.

1. Mô tả Strict 2PL.

* Một giao dịch yêu cầu một khóa chia sẻ/ độc quyền trên một đối tượng trước khi nó đọc/ sửa đối tượng đó.
* Tất cả các khóa mà giao dịch nắm bắt được giải phóng khi giao dịch đó thành công.

Câu 2. Xét các hành động được thực hiện bởi transaction T1 trên hai đối tượng CSDL như sau: R(X), W(X), R(Y), W(Y).

1. Hãy cho một ví dụ về transaction T2 sao cho nếu thực hiện đồng thời hai transaction mà không có cơ chế kiểm soát đồng thời thì có thể ngăn cản việc thực hiện T1.

|  |  |
| --- | --- |
| **T1** | **T2** |
| R(X)  W(X)  R(Y)  W(Y) | W(Y)  abort |

1. Giải thích Strict 2PL sẽ thực hiện việc ngăn cản sự ảnh hưởng giữa hai transaction.

Không có 2PL thì T2 thực hiện Lệnh W(Y) rồi

1. Strict 2PL được sử dụng trong nhiều hệ CSDL. Hãy nêu hai lý do tại sao?

Một giao dịch yêu cầu một khóa chia sẻ/ độc quyền trên một đối tượng trước khi nó đọc/ sửa đối tượng đó. Tất cả các khóa mà giao dịch nắm bắt được giải phóng khi giao dịch đó thành công.

Câu 3. Xét một CSDL có hai đối tượng X và Y. Giả sử có hai transaction T1 và T2. Transaction T1 thực hiện : R( X), R(Y) và W(X). Transaction T2 thực hiện : R(X), R(Y), W(X), W(Y).

1. Hãy cho một lịch biểu với các hành động của T1 và T2 trên đối tượng X và Y mà nó gây ra xung đột ghi- đọc (write-read conflict).

|  |  |
| --- | --- |
| **T1** | **T2** |
| **R(X)**  **R(Y)**  **W(X)**  **W(Y)** | **R(X)**  **R(Y)**  **W(Y)**  **W(X)**          **R(X)** |

1. Hãy cho một lịch biểu với các hành động của T1 và T2 trên đối tượng X và Y mà nó gây ra xung đột đọc-ghi (read- write conflict).

|  |  |
| --- | --- |
| **T1** | **T2** |
| **R(X)**  **R(Y)**      **W(X)**  **W(Y)** | **R(X)**  **R(Y)**      **W(Y)**  **W(X)** |

1. Hãy cho một lịch biểu với các hành động của T1 và T2 trên đối tượng X và Y mà nó gây ra xung đột đọc-ghi (write- write conflict).

|  |  |
| --- | --- |
| **T1** | **T2** |
| **R(X)**  **R(Y)**  **W(X)**  **W(Y)** | **R(X)**  **R(Y)**          **W(Y)**  **W(X)** |

1. Hãy lý giải Strict 2 PL sẽ không cho phép lịch biểu nào thực thi.

Sẽ không cho lịch biểu có gây ra xung đột Write – Write và xung đột Write – Read thực hiện.

Write – Write: nếu T1 hoặc T2 thực hiện Write thì hệ thống sẽ cập một Xlock cho T1 hoặc T2 đó. Và trước khi commit thì sẽ không cấp xlock cho tuyến trình khác nên sẽ không thể Write nó. Nên 2PL sẽ không cho lich biểu thực thi.

Write – Read: nếu T1 hoặc T2 thực hiện Write thì hệ thống sẽ cập một Xlock cho T1 hoặc T2 đó. Và trước khi commit thì sẽ không cấp slock cho tuyến trình khác. Nên 2PL sẽ không cho lich biểu thực thi.

Câu 4. Xét lịch biểu S (chưa đầy đủ) sau :

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

Với mỗi yêu cầu dưới đây, hãy chỉnh sửa S để tạo một lịch biểu đầy đủ thỏa mãn các điều kiện đã cho. Nếu có một chỉnh sửa nào là không thể thực hiện hãy giải thích lý do. Nếu nó có thể hãy dùng số lượng hành động có thể nhỏ nhất (Read, Write, Commit hay Abort). Bạn có thể tùy ý thêm hành động ở bất kỳ chỗ nào trong lịch biểu S.

1. Lịch biểu cho kết quả tránh đươc việc hủy bỏ dây chuyền (cascading abort) nhưng không thể phục hồi (not recorverable)

T1: R(X), T1: R(Y), T1: W(X), T2: R(Y), T3: W(Y), T1: W(X), T2: R(Y), T2 : Commit

1. Lịch biểu cho kết quả có thể phục hồi

T1: R(X), T1: R(Y), T1: W(X), T2: R(Y), T3: W(Y), T3 : Commit, T1: W(X), T2: R(Y), T2 : Commit

1. Lịch biểu cho kết quả là xung đột-khả tuần tự (conflict-serializable)

Không thể thực hiện được lịch biểu này. Vì T2 : R(Y) xung đột với T3 : W(Y) do đó không thể đổi chỗ cho nhau

Câu 5. Định nghĩa các thuật ngữ sau: conflict-serializable schedule, View-serializable schedule, strict schedule.

- Conflict- serializable schedule là một lịch trình khả tuần tự xung đột mà trong đó có ít nhất một lệnh thực hiện write.

-View- serializable schedule nghĩa là lịch trình khả tuần tự view. Một lịch trình khả tuần tự view khi nó tương đương với một lịch trình tuần tự

1. Mô tả hai nghi thức lock sau: 2PL, conservative 2PL.

a. Nghi thức lock 2PL

Nghi thức này cung cấp 2 loại khóa là S và X.

Khóa X: khi transaction T1 yêu cầu khóa X để truy cập đối tượng, các transaction T2 sẽ không thể thực hiện thao tác với đối tượng mà transaction T1 đang giữ khóa X

Khóa S: khi transaction T1 yêu cầu khóa S để truy cập đối tượng, các transaction T2 sẽ có thể thực hiện thao tác với đối tượng mà transaction T1 đang giữ khóa S. Đây gọi là khóa dùng chung

b. Nghi thức Conservative 2PL

Transaction phải yêu cầu khóa tất cả các mục dữ liệu cần thiết trước khi transaction bắt đầu thực hiện

1. Tại sao Lock và Unlock phải là các thao tác atomic.

Lock và Unlock phải là các thao tác atomic vì 2 thao tác này yêu cầu 2 trạng thái là thành công hoặc thất bại. Không được phép thực hiện nửa chừng vì như vậy sẽ không thể quyết định cấp quyền hoặc không cấp quyền truy cập dữ liệu cho các transaction

1. Vấn đề phantom (phantom problem) là gì? Có phải nó xảy ra trong CSDL mà tập các đối tượng CSDL là cố định và chỉ có giá trị của đối tượng có thể được thay đổi.

Phantom problem là các vấn đề có thể xảy ra trong lúc thực hiện các lệnh transaction và xảy ra khi các transaction tiến hành thay đổi các giá trị của đối tượng( viết) nhưng không đồng nhất khiến kết quả tổng không đúng

1. Trình bày điểm khác biệt giữa các thời biểu (timestamps) được gán cho các transaction được khởi động lại khi thời biểu được dùng để ngăn cản deadLock so với thời biểu được dùng để kiểm soát đồng thời.

Câu 6. Xác định các lớp lịch biểu dưới đây thuộc các lớp lịch biểu nào trong các lớp lịch biểu: serializable, conflict-serializable, view-serializable, recoverable, avoids-cascading-aborts, strict.

Nếu bạn không thể xác định một lịch biểu nào đó thuộc lớp nào dựa trên danh sách các hành động, hãy giải thích lý do.

Các hành động được liệt kê theo thứ tự chúng được lập lịch. Nếu một lịch không có hành động commit hay abort thì lịch đó không đầy đủ. Giả sử hành động abort/commit phải đứng sau các hành động được liệt kê.



1.         view-serializable

2.         conflict-serializable

3.         conflict-serializable

4.         conflict-serializable

5.         serializable, conflict-serializable, and view-serializable; recoverable and avoid cascading

6.         aborts

7.         Recoverable

8.         Không xác định vì nó có tất cả các tính chất trên

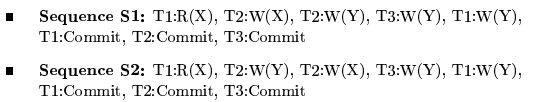
9.         avoids-cascading-aborts

10.       avoids-cascading-aborts

11.       không xác định vì nó có tất cả các tính chất trên

12.       strict

Câu 7. Xét các chuỗi hành động được liệt kê theo thứ tự được đệ trình tới DBMS sau:



Với mỗi chuỗi và với mỗi cơ chế kiểm soát đồng thời (Wait-die policy, deadLock detection, Conservative and strict 2PL, Optimistic concurrency control), hãy mô tả cơ chế kiểm soát đồng thời xử lý chuỗi hành động như thế nào.

**1.**      Với S1

T1 yêu cầu khóa S(X) và nhận được để đọc dữ liệu từ X

T2 yêu cầu khóa X(X), X(Y) và nhận được để ghi dữ liệu X, Y vào hệ thống

T3 yêu cầu khóa X(Y)

T2 trả khóa X(Y) để T3 ghi dữ liệu

T3 nhận được khóa X(Y)

T1 yêu cầu khóa X(Y)

T3 trả khóa X(Y) để T1 ghi dữ liệu

T1 nhận được khóa X(Y)

Khi T1 commit, T1 sẽ trả S(X) và X(Y)

T2 commit, T3 commit

**2.**      Với S2

T1 yêu cầu khóa S(X) và nhận được để đọc dữ liệu từ X

T2 yêu cầu khóa X(X), X(Y) và nhận được để ghi dữ liệu X, Y vào hệ thống

T3 yêu cầu khóa X(Y)

T2 trả khóa X(Y) để T3 ghi dữ liệu

T3 nhận được khóa X(Y)

T1 yêu cầu khóa X(Y)

T3 trả khóa X(Y) để T1 ghi dữ liệu

T1 nhận được khóa X(Y)

Khi T1 commit, T1 sẽ trả S(X) và X(Y)

T2 commit, T3 commit