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.

2. **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 chấtcô 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.

3. Mô tả Strict 2PL.

Strict 2PL là một giao thức khóa được sử dụng rộng rãi nhất trong đó

1) Một giao dịch yêu cầu một khóa chia sẻ/ độc quyền trên một đối tượng trước khi nó đọc/ sửa đối tượng đó.

2) 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

2. **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.

3. **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(Y)
	W(X)
R(X)	
R(Y)	
W(X)	
W(Y)	
	R(X)

2. **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)
R(X)	
R(Y)	
	W(Y)
	W(X)
W(X)	
W(Y)	

3. **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)
R(X)	
R(Y)	
W(X)	
W(Y)	
	W(Y)
	W(X)

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

Sẽ không cho lịch biểu có gay 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
2. **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
3. **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**.

   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.

   Nghi thức Conservative 2PL

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

2. **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.

3. **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.

4. **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. T1:R(X), T2:R(X), T1:W(X), T2:W(X)
2. T1:W(X), T2:R(Y), T1:R(Y), T2:R(X)
3. T1:R(X), T2:R(Y), T3:W(X), T2:R(X), T1:R(Y)
4. T1:R(X), T1:R(Y), T1:W(X), T2:R(Y), T3:W(Y), T1:W(X), T2:R(Y)
5. T1:R(X), T2:W(X), T1:W(X), T2:Abort, T1:Commit
6. T1:R(X), T2:W(X), T1:W(X), T2:Commit, T1:Commit
7. T1:W(X), T2:R(X), T1:W(X), T2:Abort, T1:Commit
8. T1:W(X), T2:R(X), T1:W(X), T2:Commit, T1:Commit
9. T1:W(X), T2:R(X), T1:W(X), T2:Commit, T1:Abort
10. T2: R(X), T3:W(X), T3:Commit, T1:W(Y), T1:Commit, T2:R(Y), T2:W(Z), T2:Commit
11. T1:R(X), T2:W(X), T2:Commit, T1:W(X), T1:Commit, T3:R(X), T3:Commit
12. T1:R(X), T2:W(X), T1:W(X), T3:R(X), T1:Commit, T2:Commit, T3:Commit

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:

- **Sequence S1:** T1:R(X), T2:W(X), T2:W(Y), T3:W(Y), T1:W(Y), T1:Commit, T2:Commit, T3:Commit
- **Sequence S2:** T1:R(X), T2:W(Y), T2:W(X), T3:W(Y), T1:W(Y), T1:Commit, T2:Commit, T3:Commit

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