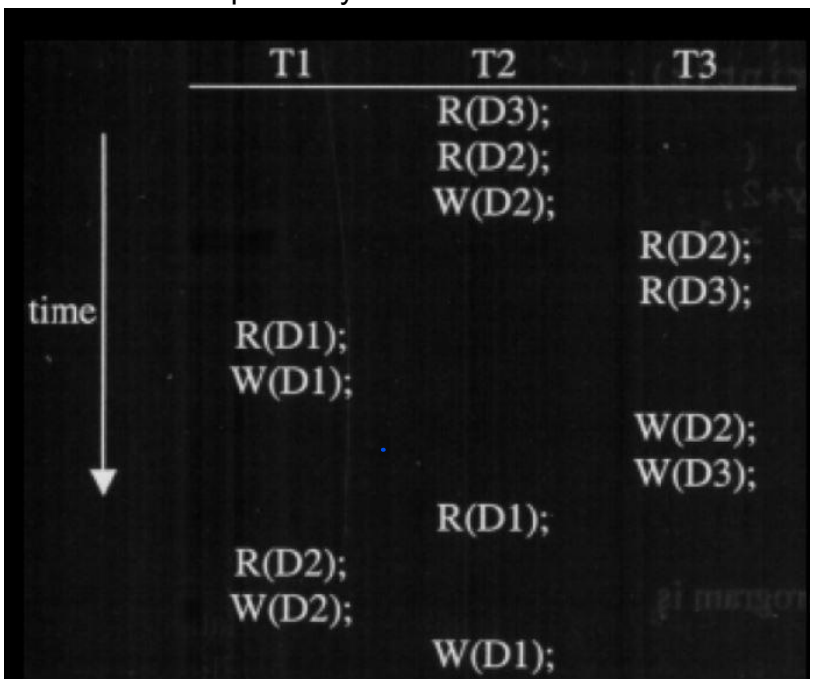
Transactions and Concurrency Control

1) Which of the following scenarios may lead to an irrecoverable error in a database system?

A) A transaction writes a data item after it is read by an uncommitted transaction
B) A transaction read a data item after it is read by an uncommitted transaction
C) A transaction read a data item after it is written by an committed transaction
D) A transaction read a data item after it is written by an uncommitted transaction

2) Consider three data items D1,D2 and D3 and the following execution schedule of transactions T1,T2 and T3. In the diagram, R(D) and W(D) denote the actions reading and writing the data item D respectively.



Which of the following statements is correct?

A) The schedule is serializable as T2; T3;T1;
B) The schedule is serializable as T2; T1;T3;
C) The schedule is serializable as T3; T2; T1;
D) The schedule is not serializable.

3) Consider a database with objects X and Y and assume that there are two

transactions T1 and T2. Transaction T1 reads objects X and Y and then writes object X. Transaction T2 reads objects X and Y, then reads X once more, and finally writes objects X and Y (i.e. T1: R(X), R(Y), W(X); T2: R(X), R(Y), R(X), W(X), W(Y))

1. Give an example schedule with actions of transactions T1 and T2 on objects X and Y that results in a write--read conflict.

2. Give an example schedule with actions of transactions T1 and T2 on objects X and Y that results in a read--write conflict.

3. Give an example schedule with actions of transactions T1 and T2 on objects X and Y that results in a write--write conflict.

4. For each of the three schedules, show that Strict 2PL disallows the schedule.

4) Consider the following sequences of actions, listed in the order they are submitted to the DBMS:

• 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

For each sequence and for each of the following concurrency control mechanisms, describe
how the concurrency control mechanism handles the sequence.

Assume that the timestamp of transaction Ti is i. For lock--based concurrency control mechanisms, add lock and unlock requests to the previous sequence of actions as per the locking protocol. The DBMS processes actions in the order shown. If a transaction is blocked, assume that all its actions are queued until it is resumed; the DBMS continues with the next action (according to the listed sequence) of an unblocked transaction.

1. Strict 2PL with timestamps used for deadlock prevention.

2. Strict 2PL with deadlock detection. (Show the waits--for graph in case of deadlock.)

3. Conservative (and Strict, i.e., with locks held until end--of--transaction) 2PL.

5) Consider the following sequence of actions S, and answer the following questions:

S: r1(A), r2(A), r3(B), w1(A), r2(C), r2(B), w2(B), w1(C)

1. Is the schedule S view-serializable? If so, provide a view-equivalent serial schedule

2. Is the schedule S conflict-serializable? If so, describe all the conflict-equivalent serial schedules

3. Is the schedule S a 2PL schedule (with exclusive locks)?

4. Is the schedule S a 2PL schedule (with shared and exclusive locks)?

6) Consider the following classes of schedules: ***serializable, conflict-serializable, view serializable, recoverable, avoids-cascading-aborts, and strict***. For each of the following schedules, state which of the preceding classes it belongs to. If you cannot decide whether a schedule belongs in a certain class based on the listed actions, explain briefly.

The actions are listed in the order they are scheduled and prefixed with the transaction name. If a commit or abort is not shown, the schedule is incomplete; assume that abort or commit must follow all the listed actions.

1. T1: R(X), T2: R(X), T1: W(X), T2: W(X) View serializability also depends on final commits of transaction.
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) View serializability cannt be decided
5. T1: R(X), T2: W(X), T1: W(X), T2: Abort, T1: Commit T2 final write is discarded because of abortion
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

7) Consider the following sequence S of actions, and answer these questions:

S: r2(A), r3(B), w1(A), r2(C), r2(D), w1(D)

1. Is the schedule S view-serializable? If so, provide a view-equivalent serial schedule

2. What is the precedence graph associated to S? Is the schedule S conflict-serializable? If so, describe all the conflict-equivalent serial schedules

3. Is the schedule S a 2PL schedule (with exclusive locks)?