

1. Review the following terms.

Transaction

ACID properties

Atomicity

Consistency

Isolation

Durability

Serial schedules

Conflict serializable

Conflict/Precedence graph

Locking/Unlocking

Two-Phase locking

Shared lock

Exclusive lock

Wait-for graph

Deadlock

2. Consider the following schedules:

- 1) T_1 : $W(A);$ $W(C);$ commit.
 T_2 : $W(B);$ $R(C);$ commit.
 T_3 : $R(B);$ commit.
- 2) T_1 : $R(A);$ $W(A); R(C);$ commit.
 T_2 : $R(C);$ $W(A);$ $W(B);$ commit.
 T_3 : $R(A);$ $W(C);$ $W(B);$ commit.
- 3) T_1 : $R(A); W(A);$ $W(B); W(C);$ commit.
 T_2 : $R(A);$ $R(B); W(B); W(C);$ commit.
 T_3 : $R(C);$ $W(C);$ commit.

For each of these schedules, please create a precedence graph and decide if the schedule is conflict serializable. If it is conflict serializable, give one example of a conflict equivalent serial schedule?

3. Consider the following two transactions:

T_1 : $R(A);$
 $R(B);$
 if $A = 0$ then $B := B + 1;$
 $W(B).$

T_2 : $R(B);$
 $R(A);$
 if $B = 0$ then $A := A + 1;$
 $W(A).$

Add lock and unlock instructions to transactions T_1 and T_2 so that they observe the two-phase locking protocol. Can the execution of these transactions result in a deadlock?