|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **National University of Computer and Emerging Sciences, Lahore Campus** | | | | |
| **final design** | **Course:** | **Advance Database Concepts** |  |  |
| **Program:** | **BS (Computer Science)** |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **Practice Problems:** | **Transactions - Solution** |  |  |
|  |  | | | |

**Q1.** Determine whether each schedule is strict, cascadeless, recoverable, or non-recoverable. Provide proper reason.

S1: r1(X); w1(X); r1(Y); w1(Y); r2(X); w2(X); C2; C1;

S2: r1(X); w1(X); r1(Y); w1(Y); r2(X); C1; w2(X); C2;

S3: r1(X); r2(X); w1(X); r1(Y); w1(Y); w2(X); C1; C2;

S4: r1(X); r2(X); w1(X); r1(Y); w1(Y); C1; w2(X); C2;

S5: r2(X); r1(X); w2(X); C2; w1(X); r1(Y); w1(Y); C1;

**ANSWER:**

S1: r1(X); w1(X); r1(Y); w1(Y); r2(X); w2(X); C2; C1; NON RECOVERABLE

S2: r1(X); w1(X); r1(Y); w1(Y); r2(X); C1; w2(X); C2; RECOVERABLE

S3: r1(X); r2(X); w1(X); r1(Y); w1(Y); w2(X); C1; C2; CASCADELESS

S4: r1(X); r2(X); w1(X); r1(Y); w1(Y); C1; w2(X); C2; STRICT

S5: r2(X); r1(X); w2(X); C2; w1(X); r1(Y); w1(Y); C1; STRICT

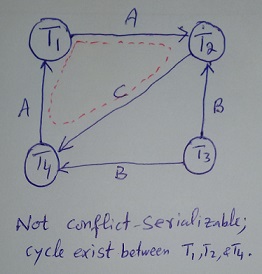
**Q2.** Consider the following schedule of four transactions T1, T2, T3, and T4.

**S:** r1(A); r4(A); w1(A); w3(B); r2(A); r2(B); w2(C); r4(B); r4(C); r2(D); r3(E).

Draw the serializability (precedence) graph for this schedule. State whether this schedule is conflict-serializable or not. If the schedule is conflict-serializable, write down the equivalent serial schedule(s) otherwise explain why it is not.

**ANSWER:**

Not conflict serializable; cycle exist T1🡪T2🡪T4🡪T1



**Q3.** Consider the three transactions T1, T2, and T3, and the schedules S1 and S2 given below.

***S1:*** *r2(X); w3(X); w1(Y); r2(Y); r2(Z); r3(Y); r1(Z);*

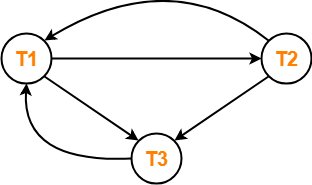
***S2:*** *r1(X); w2(X); r3(X); w1(X); w3(X);*

Draw the serializability (precedence) graphs for S1 and S2, and state whether each schedule is conflict-serializable or not and view-serializable or not. If the schedule is serializable, write down the equivalent serial schedule(s) otherwise explain why it is not.

**ANSWER:**

**Clearly, there exists no cycle in the precedence graph. Therefore, the given schedule *S1* is conflict- serializable. Thus, we conclude that the given schedule *S1* is also view-serializable. Equivalent serial schedule is T1🡪T2🡪T3. Edges in graph: T1--Y-->T2, T1--Y-->T3, & T2--X-->T3.**

**Clearly, there exists a cycle in the precedence graph. Therefore, the given schedule *S2* is not conflict- serializable. The given schedule *S2* is view-serializable and serialization order is T1🡪T2🡪T3.**



**Q4.** Given these transactions find the following schedules (if possible):

T1: r1(A); r1(B); w1(B); w1(A); c1;

T2: r2(B); w2(B); c2;

T3: r3(B); w3(B); B=B+2; w3(B); c3;

**a.** A recoverable schedule with cascade-rollback and lost update problem.

**b.** A cascade-free but not strict schedule.

**ANSWER:**

**a)**

|  |  |  |
| --- | --- | --- |
| r1(A);  r1(B);  **w1(B);**  w1(A);  c1; | **r2(B);**  **w2(B);**  **c2;** | r3(B);  **w3(B);**  **B=B+2; w3(B);**  **c3;** |

**b)**

|  |  |  |
| --- | --- | --- |
| r1(A);  r1(B);  w1(B);  w1(A);  c1; | r2(B);  w2(B);  c2; | r3(B);  w3(B);  B=B+2; w3(B);  c3; |