CSE 562: Assignment 6

(2 pages)

Due Date: No later than 11:59 **PM** on Sunday, March 31
Submission Instructions: Submit your assignment as a single PDF or plain text file (no Word files please) using the departmental submit script. Upload your file to `timberlake.cse.buffalo.edu`, then log in and run the command:

```
/util/bin/submit_cse562 [filename]
```

replacing `[filename]` with your submission file. For additional details see:
`https://wiki.cse.buffalo.edu/services/content/submit-script`
`https://wiki.cse.buffalo.edu/services/content/student-servers`

Question 1 (Based on R&G Exercise 17.2)

Consider the following classes of schedules: *serializable*, *conflict-serializable*, *view-serializable*, *recoverable*. 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 an abort or commit must follow all the listed actions.

1. $T_1 : R(X),\ T2 : R(X),\ T_1 : W(X),\ T2 : W(X)$
2. $T_1 : W(X),\ T2 : R(Y),\ T_1 : R(Y),\ T2 : R(X)$
3. $T_1 : R(X),\ T2 : R(Y),\ T3 : W(X),\ T2 : R(X),\ T_1 : R(Y)$
4. $T_1 : R(X),\ T_1 : R(Y),\ T_1 : W(X),\ T2 : R(Y),\ T3 : W(Y),\ T_1 : W(X),\ T2 : R(Y)$
5. $T_1 : R(X),\ T2 : W(X),\ T_1 : W(X),\ T2 : Abort,\ T_1 : Commit$
6. $T_1; R(X),\ T2 : W(X),\ T_1 : W(X),\ T2 : Commit,\ T_1 : Commit$
7. $T_1 : W(X),\ T2 : R(X),\ T_1 W(X),\ T2 : Abort,\ T_1 : Commit$
8. $T_1 : W(X),\ T2 : R(X),\ T_1 : W(X),\ T2 : Commit,\ T_1 : Commit$
9. $T_1 : W(X),\ T2 : R(X),\ T_1 : W(X),\ T2 : Commit,\ T_1 : Abort$
10. $T2 : R(X),\ T3 : W(X),\ T3 : Commit,\ T_1 : W(Y),\ T_1 : Commit,$
    $T2 : R(Y),\ T2 : W(Z), T2 : Commit$

11. $T_1 : R(X),\ T2 : W(X),\ T2 : Commit,\ T_1 : W(X),\ T_1 : Commit,\ T3 : R(X),\ T3 : Commit$

12. $T_1 : R(X),\ T2 : W(X),\ T_1 : W(X),\ T3 : R(X),\ T_1 : Commit,\ T2 : Commit,\ T3 : Commit$

Question 2 (Based on R&G Exercise 17.4)

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

- **Sequence S1**: $T_1 : R(X),\ T2 : W(X),\ T2 : W(Y),\ T3 : W(Y),\ T_1 : W(Y),\ T_1 : Commit,\ T2 : Commit,\ T3 : Commit$
- **Sequence S2**: $T_1 : R(X),\ T2 : W(Y),\ T2 : W(X),\ T3 : W(Y),\ T_1 : W(Y),\ T_1 : Commit,\ T2 : Commit,\ T3 : Commit$

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 T_i is i. For lock-based concurrency control mechanisms, add lock and unlock 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.
4. Optimistic concurrency control.
5. Timestamp concurrency control with buffering of reads and writes (to ensure recoverability) and the Thomas Write Rule (If TS(T) ¡ WTS(O), discard T's write to O).
6. Multiversion concurrency control.

Question 3 (Based on R&G Exercise 18.2)

1. What are the properties required of LSNs.
2. What are the fields in an update log record? Explain the use of each field.
3. What are redouble log records?
4. What are the differences between update log records and CLRs?