National University of Computer and Emerging Sciences, Lahore Campus

	Course:	**Advance Database Concepts**	**Course Code:**	**CS4064**
	Program:	**BS (Computer Science)**	**Semester:**	**Spring 2025**
	Out Date:	**31-Jan-2025**	**Total Marks:**	
	Due Date:	**Tue 11-Feb-2025** ***(Start of class)***	**Weight:**	
			Page(s):	**2**
	Assignment:	**1 (Transactions & CCT)**		

Instructions:

- Use any valid assumption where needed.
- You are required to submit the hard copy of your assignment at the start of your class.
- For any queries, please contact your TA.

Question 1.

Consider the following classes of schedules: non-recoverable, recoverable, cascadeless and strict. For each of the following schedules, state which of the preceding class it belongs to. Justify your answer.

a. **S1:** r1(X), w3(X), c3, w1(Y), c1, r2(Y), w2(Z), c2

b. **S2:** r1(X), w2(X), w1(X), c2, c1

c. **S3:** r2(X), w3(X), c3, w1(Y), r2(Y), r2(Z), c2, r1(Z), c1

d. **S4:** r1(X), w2(X), w1(X), r3(X), c1, c2, c3

e. **S5:** r1(X); r2(Z); r3(X); r1(Z); r2(Y); r3(Y); w1(X); c1; w2(Z); w3(Y); w2(Y); c3; c2

Question 2.

Consider the following schedule:

a. **S1:** r1(W), r2(X), w1(Y), r3(Z), r2(Y), w4(W), w3(X), r4(Y), w2(Z), w1(X)

b. **S2:** r1(A), r2(C), r3(A), w2(C), r3(B), r1(B), w2(B)

Draw the serializability (precedence) graph for each schedule. State whether the 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.

Question 3.

Consider the following schedule of actions listed in the order they are submitted to the DBMS:

a. **S1:** r2(X), w3(X), w1(Y), r2(Y), r2(Z), r3(Y), c3, c2, r1(Z), c1.
b. **S2:** r1(Z), r1(Y), w1(Y), w2(Y), r2(Z), r3(X), w3(X), w1(X), c1, w2(Z), r3(Y), c2, c3.

For each of the following concurrency control mechanisms, describe how the concurrency control mechanism handles the schedule. Assume that the timestamp of transaction *Ti* is *i*. For lock-based concurrency control mechanisms, add lock and unlock requests to the above schedule 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 schedule) of an unblocked transaction.

i. Basic 2PL with protocol based on a timestamp for deadlock avoidance (use wait-die policy)

ii. Strict 2PL with protocol based on a timestamp for deadlock avoidance (use wound-wait policy)

iii. Rigorous 2PL with protocol based on a timestamp for deadlock avoidance (use wait-die policy)

iv. Rigorous 2PL with protocol based on a timestamp for deadlock avoidance (use wound-wait policy)

v. Rigorous 2PL with protocol based on a deadlock detection (Use wait-for-graph to deal with deadlock)

vi. Basic timestamp ordering (TO) protocol

vii. Strict timestamp ordering protocol

viii. Timestamp ordering using Thomas's write rule (TWR)

ix. Multi-version timestamp ordering protocol

x. Validation (Optimistic) concurrency control technique (use defer the validation until a later time when the conflicting transactions have finished)