

8.5 Transactions and concurrency

Name: _____ Class: _____ Date: _____

Total: 24 marks

KEY WORDS transaction • serializability

Objective

Classify schedules as serializable or lost-update, and match isolation levels to anomalies.

1 Key knowledge and worked examples

■ The reference

T1 withdraws 10, T2 deposits 20 from a balance of 100 ✓. R1 R2 W2 W1 ends 90 (the deposit lost); R1 W1 R2 W2 ends 110 – equal to serial T1 then T2 ✓.

■ Worked example: the levels

Read-uncommitted allows the dirty read (T2 sees T1's uncommitted value that may roll back); read-committed removes it; repeatable-read stabilises re-reads; serializable removes phantoms ✓.

2 Practice

2.1 ACID: name the C and what it means. [3]

2.2 From 100, R2 R1 W1 W2 (withdraw 10, deposit 20): final value? [3]

2.3 Name 2PL's price and its cure. [3]

3 Exam-style questions

3.1 From balance 200: T1 withdraws 30, T2 deposits 50.

Compute the ending of R1 R2 W2 W1 and say which update vanished. Compute the ending of R1 W1 R2 W2. State the serializability verdict of each, with the reason. State what 2PL changes about the first schedule's outcome. [8]

3.2 T1 updates rows matching a filter; T2 inserts a new matching row between T1's two reads.

Name the anomaly. State which isolation level prevents it and which lowest level already shows consistent re-reads. Give the mechanism serializable uses (range locks or equivalent) and why read-committed cannot stop the insert. [7]

Solutions

2.1

- Consistency: constraints hold before and after every transaction ✓.

2.2

- Both read 100; T2's write lands last: $100 + 20 = 120$ – the withdrawal lost ✓.

2.3

- Deadlock; the detector kills a victim, which retries ✓.

3.1

- R1 R2 W2 W1: both read 200; final = $200 - 30 = 170$ – T2's deposit vanished ✓.
- R1 W1 R2 W2: $170 + 50 = 220$ ✓.
- Verdicts: the first is not serializable (no serial order yields 170); the second equals serial T1-then-T2 ✓.
- 2PL: T2's read of the locked balance waits, so the schedule becomes the serial-equivalent one ending 220 ✓.

3.2

- The phantom read: T1's second read sees a row its first did not ✓.
- Serializable prevents it; repeatable-read already gives consistent re-reads of existing rows ✓.
- Serializable uses range locks over the predicate's key range (or an equivalent) – the insert's key falls inside a held range ✓.
- Read-committed locks only current row reads – no range is held, so the matching insert proceeds ✓.