

**Assignment 4: Transaction Management and Database Design**

Due Thursday April 13, 3pm.

10% of your grade will be for presentation.

No late assignments will be accepted.

1. *Serializability.* (10 points)

Which of the schedules below (if any) is conflict serializable? For each serializable schedule, show an equivalent serial schedule. Show your work.

Schedule 1	Schedule 2
T1: write(A)	T2: write(B)
T1: read(F)	T1: write(A)
T2: write(B)	T2: read(F)
T3: increment(C)	T3: increment(C)
T1: increment(E)	T4: increment(E)
T4: increment(E)	T1: read(F)
T2: read(F)	T2: increment(E)
T4: read(D)	T1: read(D)
T3: read(A)	T4: write(B)
T4: write(B)	T2: write(C)
T2: write(C)	T3: read(A)
T1: increment(D)	T4: increment(D)
T3: read(F)	T1: increment(E)
T2: increment(E)	T3: read(F)

2. *Two-Phase Locking.* (5 points) Show a schedule of reads and writes that is conflict serializable but which is not possible under two-phase locking (using read and write locks). Explain why it is not possible. Short, simple schedules are preferred, and will receive more marks.
3. *Two-Phase Locking.* (10 points) A *lock point* of a transaction is any moment when it holds all of its locks. Show that every schedule of two-phase transactions is equivalent to a serial schedule in which the transactions are ordered by their lock points. *i.e.*, If transaction T1 reaches its lock point before transaction T2, then T1 is before T2 in the equivalent serial schedule.
4. *Tree Locking.* (10 points) Prove that the simple tree-locking protocol presented in class guarantees freedom from deadlock.

5. *Timestamps.* (10 points) Show that any schedule,  $S_1$ , that satisfies the time-stamp ordering rule is conflict equivalent to a serial schedule,  $S_2$ , in which transactions are ordered by their start time in  $S_1$ .
6. *Concurrency Control.* (15 points) In a tree of data items, a transaction must first read the root item, and then work its way down the tree to other items. That is, to read or write an item in the tree, a transaction must first read the item's parent. Consider the following three concurrency control methods applied to a tree of data items: (a) two-phase locking, (b) tree locking, and (c) timestamp ordering. For each *pair* of methods, do one of the following:
- show that the two methods have the same set of legal schedules for all trees of data items (where, here, a schedule includes read and write operations, but not locks); or
  - show a tree of data items and a schedule of read and/or write operations that is legal under one method but not the other.
7. *Functional Dependencies.* (4 points total) Let  $U$  be a finite set of attributes, and let  $D$  be a finite set of functional dependencies on the attributes of  $U$ . Define  $SAT(D)$  to be the set of relations over  $U$  that satisfy all the dependencies in  $D$ . Prove the following:
- (a) (2 points)  $SAT(D_1 \cup D_2) = SAT(D_1) \cap SAT(D_2)$
- (b) (2 points) If each dependency in  $D_2$  can be derived from  $D_1$  using Armstrong's axioms, then  $SAT(D_1) \subseteq SAT(D_2)$ .
8. *Functional Dependencies.* (10 points) Let  $F$  be the following set of functional dependencies:

$$\{AB \rightarrow CD, B \rightarrow DE, C \rightarrow F, E \rightarrow G, A \rightarrow B\}$$

Use Armstrong's axioms to show that  $A \rightarrow FG$  is logically implied by  $F$ . Use *only* Armstrong's axioms. In particular, do not use any lemmas or theorems based on Armstrong's axioms.

9. *Decompositions.* (10 points total). Suppose we are given the relation scheme  $ABCD$  with functional dependencies  $\{A \rightarrow B, B \rightarrow C, A \rightarrow D, D \rightarrow C\}$ . Let  $\rho$  be the decomposition  $\{AB, AC, BD\}$ .
- (a) Find the projected dependencies for each of the relation schema in  $\rho$ . (3 points)

(b) Does  $\rho$  preserve the given dependencies? (2 points)

(c) Does  $\rho$  have a lossless join with respect to the given dependencies? (5 points)

10. *Normal Forms.* (10 points total) Consider a database of ship voyages with the following attributes:  $S$  (ship name),  $T$  (type of ship),  $V$  (voyage identifier),  $C$  (cargo),  $P$  (port), and  $D$  (day). We assume that a voyage consists of a sequence of events where one ship picks up a single cargo, and delivers it to a sequence of ports. A ship can visit only one port in a single day. Thus, the following functional dependencies may be assumed:  $S \rightarrow T$ ,  $V \rightarrow SC$ ,  $SD \rightarrow PV$ .

(a) (5 points) Find a lossless-join decomposition into BCNF.

(b) (5 points) Explain why there is no lossless-join decomposition into BCNF that preserves dependencies.