Name: AndrewId:

15-415 Homework #3
(Fall 2011)
Due: Monday, November 28, 2011

Concurrency

1. Consider the following schedules:

```
S1: T1:R(X), T2:W(Y), T2:R(X), T1:W(Y), T1:Commit, T2:Commit 1
S2: T1:R(X), T2:R(Y), T1:W(Z), T1:Commit, T3:R(Y), T3:R(Z), T2:W(Y), T3:W(X),
      T2:Commit, T3:Commit
```

   For each schedule, state which of the following concurrency control protocols allows it, that is, allows the actions to occur in exactly the order shown.
   - 2PL
   - Strict 2PL

   Please provide a brief explanation for your answer...If YES, show where the lock requests could have happened; If NO, explain briefly.

XML/Semi-Structured Data

2. In what way can a large XML file be "searched"?

3. Why did we study XML technologies in 15-415? (*Hint:* It isn't because they are databases)

Distributed Databases

4. Traditional databases are based on relations, essentially tables, where each row is a fixed-structure record, such that each column represents a different attribute of that record. The distributed databases we considered, Cassandra and HBase, had a different data model, essentially a "map of maps", where the structure of the individual mapped items could be independent of other associated items. Why did distributed databases depart from the traditional model?

   In your answer please consider the benefits and limitations of each representation – but be sure to also consider the nature of distributed vs monolithic storage and distributed vs single-source access.

5. Give an example of a query that is easier to express in SQL upon relations than in HBase or Cassandra. What makes this a good example?

6. Give an example of a query that is easier to express in HBase or Cassandra than in SQL upon relations. What makes this a good example?

7. What is *consistent hashing*? Why is it important in both distributed an monolithic databases?

8. Consider *Chords* as compared to other distributed hashing schemes for example, *LH** (The distributed version of linear hashing). Why is the *Chord* approach overwhelmingly the solution used in practice for distributed hashing? What is the big advantage?

Normal Forms and Normalization

Consider the following relation and functional dependencies:

R(A, B, C, D, E, F)

A->BCD
BC->DE
B->D
D->A

9. Please provide one example of a super key. How do you know?

10. Please provide a BCNF decomposition, preserving only the expressed functional dependencies (not the canonical cover)

11. Please derive the canonical cover.

(See next page)

Note: You'll need to wait until after Tuesday's lecture to answer these two questions (or read ahead)

12. If you can, please provide a BCNF decomposition of the relation, preserving the canonical cover. If impossible, please provide a 3NF decomposition and explain the intuition behind why a BCNF decomposition isn't possible.

13. Please decompose the following schema into 3NF, given the provided functional dependencies:

R (A, B, C, D, E)

A->BC
CD->E
B->D
E->A