Name: AndrewId:

15-415 Final Exam
Fall 2010 (Kesden)

Notes:

- For simplicity, all questions are worth an equal number of points
- Questions differ in complexity
- Questions differ in the amount of effort required to answer
- Questions differing in length of answer anticipated; shorter (but fully correct) answers are generally better than longer ones
- Partial credit might be helped by exposing more of your thought process, rather than less
- Call, any hour, with any questions: 412-268-1590 (Greg's cell)

XML/Semi-Structured Data

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

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

Distributed Databases

3. 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.

4. 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?

5. 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?

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

7. 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?

Query Optimization

8. Consider the following query (yes, substr does what you think it does):

```
select * from r,s
where substr(r.lastName,10) = substring(s.lastName,10);
```

   Describe two different query plans that might be generated by the compiler to execute this query. In other words, with respect to the relational algebra portion of the query (not the string munging), please describe solutions to the problem using two different data structures or ways of solving the problem generally available within a database, e.g. hashing, indexing, merging, etc.

9. Given some relation, *r*, with some attributes *size* and *name*, please explain how a histogram can be used to select those *names* associated with sizes greater than some arbitrary *x*.

10. When are the stats stored within the catalogue updated? Why is this strategy generally employed?

Normal Forms and Normalization

11. 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

Consider the following relation and functional dependencies:

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

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

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

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

14. Please derive the canonical cover.

15. 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.