CS 631: ITDBMS: Mid Semester Exam, 25 Feb 2012

Time: 3.00 PM – 5.00 PM Marks: 60

Instructions: **Answer all parts of a question together.**
In case of any ambiguity, make any required assumptions, but state them clearly.

1. Storage and File Structures

   (a) Normally a null bitmap is used to identify attributes that are null. Give two separate conditions under which a bit need not be maintained for a particular attribute. ...2

   (b) In the context of flash memory, what is wear-leveling, and why is it required? ...3

   (c) A trick often used to reduce seek time on magnetic disk is to use only a few tracks for storing data, so the disk arm never has to move very far. Would it be better to use the inner tracks, the middle tracks, or the outer tracks to implement this idea? Briefly explain why. ...3

   (d) What is a latent failure on a disk, and how can RAID systems minimize the impact of such failures? ...2+2

2. Indexing

   (a) When using a B^+-tree file organization what should secondary indices use in place of physical record pointers, and why? ...3

   (b) Suppose an SQL statement deletes 10% of the records from a large relation. What would be the best way to update a secondary B^+-tree index on the relation (assuming that the deleted records are scattered around the index). ...4

3. Query processing

   (a) Give an example of a situation where an indexed nested loops join could be much worse than a hybrid merge join (where we use the index to find record pointers, but sort on record pointers and complete the merge join). ...3

   (b) It is possible to dynamically switch from indexed nested loops join to hybrid merge join at any point when a given outer tuple has been completely processed. Briefly suggest on what basis such a decision may be made dynamically, rather than at optimization time. ...5

4. Query optimization

   (a) Give conditions under which ${}_L\mathcal{G}(r \bowtie s) \equiv \Pi_L(r \rtimes s)$, where all operations are in the multiset relational algebra; note that the groupby operation ${}_L\mathcal{G}$ without any aggregation outputs the distinct values of attributes L. ...4

   (b) Using the query $r \bowtie_{r.A=s.A} \sigma_{s.B>10}(s)$, explain why pushing selections down is not always a good idea. Also describe the small modification to the recursive *findbestplan* algorithm (presented separately from the algorithm in the book) to handle this case. ...2+3

5. Transaction processing and locking

   (a) Give a schedule involving just one data item, which results in a deadlock using two-phase locking. ...3

   (b) Very briefly explain why the serialization order matches the commit order using rigorous two-phase locking. ...3

   (c) Give an example of a schedule with strict two-phase locking where the serialization order does not match the commit order. ...4

6. Timestamp Protocol

   (a) Explain what would happen in the following schedule, if (a) the basic timestamp ordering protocol is used without Thomas' write rule and (b) with Thomas' write rule.

   Ti denotes a transaction with timestamp i. ...4

```
     T1           T2           T3
    ------------------------------
1.  R(X)
2.               R(X)
3.  W(X)
4.               W(X)
5.                            W(X)
6.               W(X)
```

   (b) Give an example of what could go wrong if a read and a write transactions both do a timestamp check at exactly the same time. How does a database system prevent such a race condition? ...3

   (c) Now suppose we use an incr(X,V) operation, which adds value V to X, and which does not conflict with another incr operation, but does conflict with read and write operations. To handle this, we can add an extra incr-timestamp, set initially to the write timestamp when the item is created.

   Give a modified version of the TSO rules for read, write and incr, showing the modified tests for rejecting an operation using the incr-timestamp, and rules on when and how the incr-timestamp is updated. ...7

Total Marks = 60