CS632 Endsem Exam, Spring 2011, May 1, 2011, 9.30 AM -12.30 PM

Question paper pages: 3 Total Marks: 80

NOTE: Answer all subparts of a question together, do **not** split them up.

1. Short answers

   (a) Which two papers did you like the most, and which two papers did you like the least, from among the papers covered this year in CS 632? ...2

   (b) In the MQO paper, a node is considered shared if it occurs beneath two children of an operation node, but is not considered shared if it occurs beneath two children of an equivalence node. Explain why. ...3

   (c) BigTable index only logs updates, but does not apply them immediately (since the files are not updated in place). Explain (a) how BigTable ensures that queries see earlier updates, and (b) the merging step handles update entries. ...5

   (d) BANKS: Consider a database with three relations
   customer(CustID, name, nation),
   part(PartID, name), and
   order(OrderID, CustID, PartID, date)
   where primary key attributes are underlined, and foreign key attributes can be inferred from the attribute names.

      i. Given a query with the two keywords India, Dehydrated-Martians, what would you expect to get as high-ranked answers, assuming the first keyword only matches nations, and the second only matches part names? ...1

      ii. Amongst the above answers with the same structure, how would the answers be ranked? (Just give a brief intuition, don't give formulae). ...2

      iii. Give an example of a low-ranked, and probably meaningless, answer that could get generated in response to the above query. ...2

   (e) Consider the snapshot isolation anomalies paper. Suppose you have only one relation *r(empid, dept, sal)*, and only one kind of transaction, which reads all employees of a particular department, and adds 100 rupees to their salary. Multiple instances of this transaction may run concurrently, with snapshot isolation. (a) Why would the basic column-based analysis flag this transaction as vulnerable to SI anomalies? (b) Why would the later techniques flag it as a false positive? ...2+3

   (f) Consider a Web page where you need to select a city, but instead of a huge dropdown list, the page requires you to first select the state, and then displays the cities in the state. (a) Explain how you would write a query in the syntax of the "Ajax-Based Report Pages..." paper, to support such an interface. (Don't worry about minor syntax details of the query, and don't worry about the presentation layer, which translates the data tree to HTML) (b) Explain how incremental view maintenance results in the display being updated. ...3+2

   (g) The Aurora streams paper describes how to change the network even while it is active. (a) Explain intuitively how this is done without losing any results. (b) The above technique would not actually work with stateful operators such as joins in Eddies. Explain why. ...2+3

Subtotal = 30

2. MapReduce and SCOPE:

   (a) Consider the binary operation $r \cap s$, for relations $r(A, B)$ and $s(A, B)$. Explain (intuitively, no need for code) (a) how you would implement this using the Combine operator of SCOPE and (b) the reduce operation in Map Reduce. ...4

   (b) Why is partial aggregation, mentioned in the SCOPE paper, particularly important in a typical data center architecture, where nodes in a rack are connected to a switch, and that switch is in turn connected to other racks? ...2

   (c) It is easy to translate standard partitioned parallel join into MapReduce (as you did in your assignment). Now consider how to implement asymmetric partitioned join of $r \bowtie s$, where s is replicated to each machine, and joined with whatever r tuples are stored in that machine.

      i. Under what situation would such a join be useful? ...1

      ii. Assume the reducer key can be divided into two parts, and only the first part is used to decide which machine handles a particular group. Now explain how to implement the above join using map-reduce, ensuring that relation s is replicated and r does not move. ...7

3. Test data generation: Suppose you are given the relations $r(A, B), s(B, C)$ and $t(C, D)$, and the query $q = (r \bowtie s) \bowtie t$,

   (a) Give 4 examples of join-type mutations between join and left outerjoin, for this query. ...2

   (b) Give four datasets that would be sufficient to kill **all** join-type mutants for this query. For each dataset, give an example set of tuples, and explain, using first order logic where required, what property it satisfies. ...6

   (c) If a foreign key $s.C \rightarrow t.C$, where $s.C$ is declared as not null, is added, which data set would no longer be generated, and why? ...2

4. Suppose you have a query $q = r \bowtie s \bowtie t$, and two authorized views $r \bowtie s$ and $r \bowtie t$.

   (a) Is q authorized? Explain your answer. ...2

   (b) Assuming the usual transformation rules for associativity and commutativity, would the techniques from the fine-grained authorization paper be able to infer correctly whether q is authorized or not? Explain your answer. ...4

   (c) Why is redundancy removal during simplification more efficient that redudancy removal during transformation? Explain briefly. ...4

5. Consider the Stack-Tree-Desc algorithm, which can be used to compute ancestor/descendant joins efficiently. Extend it to compute a path expression $a//d/c$, i.e. where sorted lists a-list, d-list and c-list are given for a, d and c, and the algorithm should efficiently find each c node in c-list which is a child of some node d in d-list, such that d is a descendant of some node a in a-list. (The Holistic Twig Joins paper presents a general form of such a multi-way join algorithm, but the goal for this question is more limited.) ...8

6. Uncertain Data:

   (a) Suppose you have relations :

   - *courseOffering(year, sem, department, course)*, which lists which courses ran in which semester,
   - *studentInfo(year, sem, course, grade)* which gives grades of some students, and
   - *approxInfo(year, department, grade)*, which gives partial information about the grades of some students, where the semester information is not known, and the exact course is unknown, but the department is known.

   Explain how you would allocate the tuples from *approxInfo* to *studentInfo*, using each of the allocation techniques. In each case, give the intuition, and for uniform and count based, give the formulae you would use. ...6

   (b) Explain why query execution is slower with uniform allocation than with count-based or correlation preserving allocation. ...2

Total Marks = 80