CS411 Database Systems

Spring 2009, Prof. Chang

Department of Computer Science
University of Illinois at Urbana-Champaign

Final Examination

May 8, 2009

Time Limit: 180 minutes

- Print your name and NetID below. In addition, print your NetID in the upper right corner of every page.

  Name: _____ **NetID:** _____

- Including this cover page, this exam booklet contains **15** pages. Check if you have missing pages.
- The exam is closed book and closed notes. You are allowed to use scratch papers. No calculators or other electronic devices are permitted. Any form of cheating on the examination will result in a zero grade.
- Please write your solutions in the spaces provided on the exam. You may use the blank areas and backs of the exam pages for scratch work.
- Please make your answers clear and succinct; you will lose credit for verbose, convoluted, or confusing answers. *Simplicity does count!*
- Each problem has different weight, as listed below– So, plan your time accordingly. *You should look through the entire exam before getting started, to plan your strategy.*

Problem	1	2	3	4	5	6	7	8			Total
Points	32	16	10	10	10	12	17	13			120
Score											
Grader											

Problem 1 (*32 points*) Misc. Concepts

For each of the following statements, indicate whether it is *TRUE* or *FALSE* by circling your choice. You will get *2 point* for each correct answer, *-1.0 point* for each incorrect answer, and *0 point* for each answer left blank.

(1) *True* *False*

If two relations are both in BCNF, their join must also be in BCNF.

(2) *True* *False*

Transaction management consists of two main functional components: *concurrency control* and *failure recovery*.

(3) *True* *False*

SQL Injection attacks a Web site by manipulating the user input to cause harmful SQL commands to be executed at the backend database.

(4) *True* *False*

Relational algebra was invented to formalize the underlying operations of the SQL language.

(5) *True* *False*

When translating an E-R diagram to the relational model, there are multiple ways to translate a sub-class relationship.

(6) *True* *False*

With respect to a set of integers, there exists a unique structure to index them in a B+ tree.

(7) *True* *False*

In determining a query plan involving joins, by focusing on only left-deep join trees, we are *not* guaranteed to generate the optimal query plan.

(8) *True* *False*

Regardless of UNDO or REDO, a logging system must write the corresponding log entry *before* any update of database values on disk.

(9) *True* *False*

In cost-based optimization, dynamic programming is a technique that helps us to estimate the cost of a query plan.

(10) *True* *False*

For the same operations (*e.g.*, sorting, grouping), two-pass algorithms generally require less memory buffer than their corresponding one-pass algorithms.

(11) *True* *False*

An important requirement for database indexing is the ability of the index to maintain an appropriate structure as the database changes over time.

(12) *True* *False*

The *Explain Plan* facility in Oracle SQL shows the execution plan and its actual cost by executing the plan.

(13) *True* *False*

We can use *Hints* to tell Oracle query optimizer what to do– Further, a Hint can adapt to the changes of a database to lead to good query plans over time.

(14) *True* *False*

PostGIS implements the OpenGIS SFSQL and thus provides geo-spatial data types such as *Point* and *Polygon* as well as functions like *Distance* and *Within*.

(15) *True* *False*

A typical "denial of service" attack is to attack a database server by sending complex SQL queries that would cause the underlying query optimizer to hang and thus result in the intended denial.

(16) *True* *False*

Schema normalization is a technique that will lead to more efficient query processing.

Problem 2 (*16 points*) Short Answer Questions

For each of the following questions, write your answer in the given space. You will get *2 points* for each correct answer.

(1) Answer: _____
Consider relations $R(a,\ b)$ and $S(a,\ c)$, for the following query. Would a B+ tree index on $S.a$ help in query processing? Briefly explain.

SELECT a FROM R, S WHERE b<10 and $R.a = S.a$ and c>20

(2) Answer: _____
For the above SQL query, is there a *unique* way to write it in relational algebra? If so, give such an expression. If not, explain why.

(3) Answer: _____
For the following table, show a decomposition (into two tables) that is lossless.

A	B	C	D
1	3	2	2
2	3	2	4
3	1	3	6
3	1	1	6

(4) Answer: _____
Consider relation $R(a,\ b,\ c,\ d)$. Give an E-R diagram with *at least two* entities E_1 and E_2, such that the diagram, when translated into the relational model, will result in R.

(5) Answer: _____
Given a query that joins n relations, considering only left-deep join trees, how many different join orders are there? Briefly explain.

(6) Answer: _____
Briefly explain what "I" stands for *and* what it means in the "ACID" properties.

(7) Answer: _____
Give one advantage of extensible hash table indexing over linear hash table.

(8) Answer: _____
Consider relation $R(a,\ b,\ c)$, where a is the primary key with values in the range of [11, 110] and $T(R)$=200. Determine the size of $\sigma_{a=100}R$.

Problem 3 (*10 points*) Schema Decomposition

Consider a relation R with five attributes A, B, C, D, and E. The following dependencies are given: $AB \rightarrow C$, $BC \rightarrow D$, $CD \rightarrow E$, $DE \rightarrow A$.

(a) List all keys for R. Do not list superkeys that are not a key. (*3 points*)

(b) Is R in 3NF? Briefly explain why. (*3 points*)

(c) Is R in BCNF? If yes, please explain why. Otherwise, decompose R into relations that are in BCNF. (*4 points*)

Problem 4 (*10 points*) Query Languages

The following questions refer to the database schema below: *Product*(*pid, price, color*), *Order*(*cid, pid, quantity*), *Customer*(*cid, name, age*).

(a) Write a query, in *relational algebra*, to return the names of customers who order at least one product with color "Red." (*2 points*)

(b) Write an SQL query, to return the total quantity of products ordered by customers with age greater than 70. (*3 points*)

(c) Write an SQL query, to return the pid(s) of the most ordered product(s) (i.e. the product(s) with the highest total ordered quantities). (*5 points*)

Problem 5 (*10 points*) Indexing: B+tree

Consider the B+tree of order 4 (*i.e.*, $n = 4$, each index node can hold at most n keys and $n + 1$ pointers) shown in Figure 1.

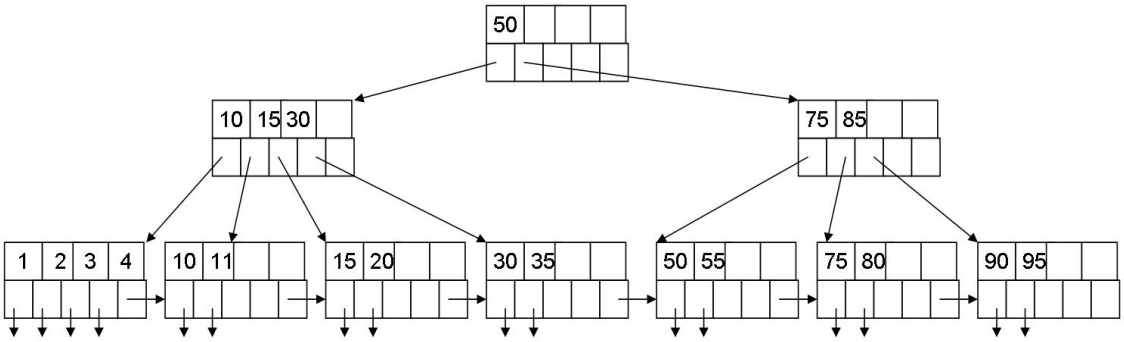

Figure 1: B+ tree.

(a) Based on the tree in Figure 1, show the resulting tree after inserting key 5. (*4 points*)

(b) Based on the tree in Figure 1, show the resulting tree after deleting key 90. (*4 points*)

(c) Based on the tree in Figure 1, show the steps in executing the following operation: Look up all records in the range 40 to 100 (including 40 and 100). (*2 points*)

Problem 6 (*12 points*) Query Processing

Consider joining two relations $R(x, y)$ and $S(x, z)$ on their common attribute x. The size of relation R is 1000 blocks and the size of relation S is 500 blocks. Attribute x of relation R has 50 different values and the values are evenly distributed in R. Attribute x of relation S also has the same 50 different values and the values are evenly distributed in S. Suppose that both relations are not sorted by attribute x.

(a) Suppose the memory buffer has 101 blocks, compute the cost of join using a sort-merge join. (*3 points*)

(b) Suppose the memory buffer has 101 blocks, compute the cost of join using a partitioned hash join. (*3 points*)

(c) We analyzed the I/O requirements of the sort-merge join algorithm in the class. However, the algorithm needs additional disk I/O's if there are so many tuples with the same value in the join attribute that those tuples cannot fit in the main memory.

Suppose the memory buffer has 101 blocks. Assume that attribute x of relation R has two distinct values (x_1 and x_2) and the values are evenly distributed in R. Similarly, attribute x of relation S also has the same two values (x_1 and x_2) and the values are evenly distributed in S. Compute the total number (in average) of disk I/Os that are needed for the sort-merge join algorithm. (*6 points*)

Problem 7 (*17 points*) Query Optimization

(a) Give an example to show: Projection cannot be pushed below bag difference. (*3 points*)

(b) Consider a relation $R(a,\ b,\ c,\ d)$ that has a clustering index on a and non-clustering indexes on each of the other attributes. The relevant parameters are:

$B(R) = 1000$, $T(R) = 5000$, $V(R, a) = 20$, $V(R, b) = 1000$, $V(R, c) = 5000$, $V(R, d) = 500$.

Give the best query plan and the disk I/O cost for each of the following selection queries:

(1) $\sigma_{(a=1)AND(b=2)AND(c=3)}R$. (*4 points*)

(2) $\sigma_{(a=1)AND(b=2)AND(c<3)}R$. (*4 points*)

(c) Consider the following query that joins *Author*(*aid, aname*), *Write*(*aid, bid*), *Book*(*bid, btitle, pid*), *Publisher*(*pid, pname, paddr*).

select *aname*, *btitle*, *pname*
from *Author A*, *Write W*, *Book B*, *Publisher P*
where $A.aid = W.aid$ AND $W.bid = B.bid$ AND $B.pid = P.pid$.

Consider dynamic programming for generating the optimal join order. If we do not want to consider Cartesian products, without actually working through the whole process, calculate how many subqueries each iteration needs to consider. (Note: we are asking for logical queries, **NOT** physical plans) (*6 points*)

Problem 8 (*13 points*) Failure Recovery

Consider the following log sequence.

Log ID	Log
1	$\langle \text{START } T1 \rangle$
2	$\langle T1, A, 1 \rangle$
3	$\langle \text{START } T2 \rangle$
4	$\langle T1, B, 2 \rangle$
5	$\langle \text{COMMIT } T1 \rangle$
6	$\langle T2, B, 2 \rangle$
7	$\langle \text{COMMIT } T2 \rangle$
8	$\langle \text{START } T3 \rangle$
9	$\langle T3, A, 3 \rangle$
10	$\langle \text{START } T4 \rangle$
11	$\langle T3, B, 4 \rangle$
12	$\langle \text{COMMIT } T3 \rangle$
13	$\langle T4, C, 5 \rangle$
14	$\langle \text{START } T5 \rangle$
15	$\langle \text{COMMIT } T4 \rangle$
16	$\langle T5, A, 6 \rangle$
17	$\langle \text{COMMIT } T5 \rangle$

Note: For the questions below, assume the given log sequence is a *UNDO* log.

(a) Suppose we want to start checkpointing right after logID 11. In the space below, indicate *where* the start checkpointing record would be, and *what* it would look like. Then, indicate *where* the earliest end checkpoint record would be, and *what* it would look like. (*2 points*)

(b) Continue from (a). Suppose the system crashes right after logID 16. What is the portion of the log we would need to inspect and which transactions need to be undone? (*3 points*)

Note: For the questions below, assume the given log sequence is a *REDO* log.

(c) Suppose we want to start checkpointing right after logID 4. In the space below, indicate *where* the start checkpointing record would be, and *what* it would look like. Then, indicate *where* the earliest end checkpoint record would be, and *what* it would look like. (*2 points*)

(d) Continue from (c). Suppose the system crashes right after logID 16. If ⟨END CKPT⟩ is written to the log, indicate the portion of the log we would need to inspect and which transactions need to be redone. (*3 points*)

(e) Now, suppose we start checkpointing right after logID 4 and the system crashes right after logID 6. If ⟨END CKPT⟩ is *not* written to the log, indicate the portion of the log we would need to inspect and which transactions need to be redone. (*3 points*)