

University of Illinois at Urbana-Champaign
Midterm Examination
CS511 Advanced Database Management Systems

Time Limit: 75 minutes
Exam Date: Oct. 11, 2006

- Closed notes; closed book; no sheet of formulas permitted.
- Calculators are not permitted.
- Please write your answers directly on the exam sheet.
- The space we left for your answers is often more than what you actually need, so please do not judge how much you need to write based on the amount of space.
- In case you find a question ambiguous, please write down your assumption and answer the question accordingly.
- Please use the back side of the exam as scratch paper.

Your Name:

Your NetID:

Good Luck!

1. [15 points] For each of the following statements, indicate whether it is *TRUE* or *FALSE* by circling your choice. You will get 1 point for each correct answer, -1 point for each incorrect answer, and 0 point for each answer left blank.

1. True ~~False~~
Among Codd's contributions is the definition of the SQL language which enables declarative querying over a database.
2. True ~~False~~
Codd's original relational operators (*i.e.*, permutation, projection, join, composition, and restriction) are *minimal* - None of the operators can be expressed in terms of the others.
3. True ~~False~~
System R's cost-based optimization is also called "Selinger's Style" optimization.
4. True ~~False~~
Many of System R's techniques/concepts are long lasting and still used in DBMS today: *e.g.*, cost-based query optimization and degrees of consistency.
5. True ~~False~~
KD-tree is a balanced search tree.
6. True ~~False~~
2-Phase Locking guarantees serializability and prevents deadlock.
7. True ~~False~~
If the data values to be indexed can be arranged in a linear order, they can always be indexed with a B-tree structure to support range queries for these values.
8. True ~~False~~
Finer granularity locking can increase concurrency, but it can also increase the overhead of concurrency control.
9. True ~~False~~
A schedule with degree 2 consistency is guaranteed to be serializable.
10. True ~~False~~
ARIES builds upon the idea of "shadow paging" proposed for crash recovery in System R.
11. True ~~False~~
When evaluating predicate $\text{gpa} > 3.5$ with a non-clustered index on *gpa*, a data page can be touched more than once.
12. True ~~False~~
System R was developed at the IBM San Jose Labs (*i.e.*, today's Almaden Labs), and became the first relational DBMS product on the market.
13. True ~~False~~
The Wisconsin benchmark is influential because it introduces the performance metric TPS.
14. True ~~False~~
In an R-tree, *unlike* B-tree, the indexed region of a node can overlap with that of its siblings.
15. True ~~False~~
The issue of *steal* vs. *no-steal* policies in page replacement suggests that the operating system needs to be aware of the state of transactions in order to correctly manage their buffer.

2. [25 points] Indexing

a [5/25 points] Briefly explain why keeping an index/search tree balanced is important in a DBMS. In particular, explain what undesirable result may be caused by an un-balanced index tree.

A balanced index tree minimizes the maximum path from the root to a leaf node. For example, for a skewed tree which is an un-balanced tree, the cost of the maximum path is n , where n is the number of nodes, while $\log n$ for a balanced tree.

Grading Criteria:

- 5 points for any correct argument
- -3 points for other arguments

b [5/25 points] Briefly describe a scenario where a non-balanced index tree may be better than a balanced index tree.

If there is a *frequent access pattern*, a non-balance tree is better. For example, assume that we have 8 records. If we frequently access only 2 records among them, then it is better to move these 2 records to higher nodes instead of keeping them at the same level with other records.

Grading Criteria:

- 5 points for any correct argument
- -1 point for the scenario where data distribution is not uniform
- -3 points for other arguments

c [15/25 points] We want to implement a GiST to support queries about identifying pictures that contain some given colors. Suppose the query predicate to be supported is $HasColor(x, c)$ where x is an image and c is a set of colors, and a query with the predicate $HasColor(x, c)$ is supposed to return all the images that have *at least one* of the colors in c . What key predicate would you use in your GiST implementation? How would you implement the $Consistent(E, q)$ method? How would you implement the $Union(E_1, \dots, E_n)$ method? (E, E_1, \dots, E_n are entries with your key predicates and q is a query predicate.)

Key predicate: $HasColor(x, c)$ where x is an image and c is a set of colors, *i.e.*, TRUE if the image has *at least one* of the colors in c and FALSE otherwise.

$Consistent(E, q)$: When $p = HasColor(x, c1)$ and $q = HasColor(x, c2)$, return TRUE if $c1$ and $c2$ overlap; FALSE otherwise.

$Union(E_1, \dots, E_n)$: When $p_i = HasColor(x, c_i)$, $c_1 \cup \dots \cup c_n$.

Grading Criteria:

- 5 points for each
- -2 points: if you say that $c1$ contains $c2$ (or vice versa) for the Consistent method

3. [10 points] Degrees of Consistency

Given the following three transactions:

T1: read1(A); writel(A); commit1;

T2: write2(A); commit2;

T3: read3(B);read3(A);write3(A);write3(B);commit3;

Give an interleaved schedule in which every transaction sees degree 3 of consistency. Draw the precedence graph for this schedule.

Many answers are possible. Here is one sample answer.

Schedule:

	T1	T2	T3
1			read3(B);
2	read1(A);		
3	write(A);		
4	commit1;		
5		write2(A);	
6		commit2;	
7			read3(A);
8			write3(A);
9			write3(B);
10			commit3;

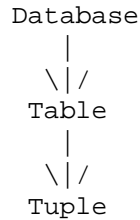
Precedence graph: T1 → T2 → T3

Grading Criteria:

- 8 points for the schedule
 - -1 point, if the schedule is NOT interleaved
 - -5 points, if every transaction does not see degree 3 of consistency
- 2 points for the precedence graph, which will be graded based on the given schedule
 - -1 point for the wrong direction in the precedence graph
- 0 points, if a precedence graph is given without a schedule

4. [15 points] Lock Management

Suppose we have the following hierarchy of objects to be locked in our database:



For each of the following actions, give a sequence of lock acquisitions. Your sequence may contain S, X, IS, IX, SIX locks and references to the objects and their names where appropriate. All the actions refer to table “Students(StudentID,Name,GPA)”.

a [5/15 points] Delete the record with StudentID=3 in table Students (we don’t need to find the tuple, i.e., we have already found the tuple).

```
IX(Database)
IX(Table)
X(The tuple corresponding to StudentID=3)
```

Grading Criteria: -1 point for each missing/unnecessary/wrong lock

b [5/15 points] Get the Name of the student with StudentID=15 in table Students (we don’t need to find the tuple, i.e., we have already found the tuple).

```
IS(Database)
IS(Table)
S(The tuple corresponding to StudentID=15)
```

Grading Criteria: -1 point for each missing/unnecessary/wrong lock

c [5/15 points] Change the GPA of the student with Name=“Frannzee” (assume Frannzee is unique) to 4.0 in table Students. (FIND and change involved here!)

```
SIX(Database)
SIX(Table)
X(The tuple matching Name=“Frannzee”)
```

Grading Criteria:

- 2 points for the SIX lock on Database
- 2 points for the SIX lock on Table
- -1 point for the S lock on each tuple that we touch in order to find right tuple (since this lock is implied in the SIX lock)
- -1 point for each any other missing/unnecessary/wrong lock

5 [10 points] Query Optimization: System R

a [5/10 points] What are the three components in the cost-based optimization method used in System R?

- Plan space
- Cost estimation
- Search algorithm

Grading Criteria:

- -1 point for each missing/incorrect component

b [5/10 points] Give two reasons why the System R query optimizer may miss the true optimal query plan.

- The System R query optimizer only considers left-deep join trees.
- The Cost estimation is approximate.
- The greedy search algorithm can miss the global optimal plan.
- ...

Grading Criteria:

- -1 point for each missing/incorrect reason

6 [15 points] Query Optimization: Cost Estimation

Consider relations $r(A, B)$ and $s(B, C)$. Assume that r contains 2,000 tuples, and that s contains 5,000 tuples. We want to compute $v = r \text{ JOIN}_{r.B=s.B} s$.

a [5/15 points] Without any further assumptions, what is the maximum number of tuples that v may contain?

$$2000 \times 5000 = 10,000,000$$

b [5/15 points] Now assume that we know that $V(B, r) = 500$. (That is, in r the attribute B takes on 500 different values.) What is now a reasonable estimate on the size of v ?

$$2000/500 \cdot 5000 = 20,000$$

c [5/15 points] Finally, assume we know that s satisfies the functional dependency $B \rightarrow C$. What is now a reasonable estimate on the size of v ?

2000, since B is a key

Grading Criteria (for each sub-question):

- 0 point for an incorrect answer without any explanation
- 2-3 points for an incorrect answer with *reasonable* explanation
- 1 point for an incorrect answer with some explanation
- No deduction for calculation errors

7 [10 points] Logging/Recovery

Use no more than 2 sentences to answer each of the following questions.

a [4/10 points] In ARIES, what does checkpointing do?

It helps to minimize the time to recover from system crash.

b [3/10 points] In ARIES, what is the purpose of creating a compensation log record (CLR) in the undo phase?

It helps not to undo transactions, which are already undone.

c [3/10 points] In ARIES, how does the system know what transactions to undo?

Using the transaction table constructed by the Analysis phase, it identifies all transactions active at the time of the crash (*i.e.*, uncommitted transactions).

Grading Criteria:

- Any reasonable explanation gets the full points.