CPSC 310: Database Systems / CSPC 603: Database Systems and Applications
Exam 2
November 16, 2005

Name: _____

Instructions:

1. This is a closed book exam. Do not use any notes or books, other than your two 8.5-by-11 inch review sheets. Do not confer with any other student. Do not use any computer equipment.
2. Show your work. Partial credit will be given. Grading will be based on correctness, clarity and neatness.
3. I suggest that you read the whole exam before beginning to work any problem. Budget your time wisely—according to the point distribution.
4. There are 4 questions worth a total of 100 points, on 7 pages (including this page).

DO NOT BEGIN THE EXAM UNTIL INSTRUCTED TO DO SO. GOOD LUCK!

Please sign the academic integrity statement:

"On my honor, as an Aggie, I have neither given nor received unauthorized aid on this academic work. In particular, I certify that I have not received or given any assistance that is contrary to the letter or the spirit of the guidelines for this exam."

Signature: _____

References for these problems:

- web site for our textbook

1. (30 pts total) True/False.

(a) (3 pts) On average, repeated random disk I/O's are faster than repeated sequential disk I/O's because random I/O's tend to access different cylinders and therefore cause less contention.

(b) (3 pts) For any data file, it is possible to construct two separate sparse indexes on different keys.

(c) (3 pts) There is a benefit to constructing a two-level index that has a dense first level and a dense second level.

(d) (3 pts) RAID Level 5 (having each data disk serve as the spare disk for some blocks) allows the system to tolerate up to 5 disk crashes at the same time.

(e) (3 pts) Since there is no downside to having indexes, we may as well build an index on every attribute of every relation to speed up as many queries as possible.

(f) (3 pts) A drawback of hashing indexes is that they are slower than B-trees for queries of the form "attribute equals constant".

(g) (3 pts) In order to perform bag union of relations R and S with M memory buffers, the size of the smaller relation must be at most M^2.

(h) (3 pts) Say a randomly chosen block is being fetched from disk. The overall access delay per byte decreases as the block size increases.

(i) (3 pts) Consider relation $R(A, B, C, D)$ and a query that asks for all tuples of R with attribute A between 30 and 100. Suppose relation R is stored in a file using 100 blocks, and there is a B-tree index of height 2 over R with key A that has 30 leaf nodes. It would take fewer disk I/O's to use the index to execute the query than to perform a sequential scan on the file.

(j) (3 pts) Since natural join is associative and commutative, the same number of disk I/O's are executed no matter what order a series of joins are done in.

2. (25 pts total) Disk Storage Organization.

Suppose we have a schema with three relations, all with fixed length, fixed format records.

- `Company(Number,...)` : Contains 16,000 tuples. Each tuple is 250 bytes. `Number` is key and is 20 bytes.
- `Product(Number,Id,...)` : Contains 54,000 tuples. Each tuple is 200 bytes. `Number` refers to the number of the company that makes the product and is 20 bytes. `Id` is the product identifier and is 25 bytes.
- `Model(Id,Price,...)` : Contains 144,000 tuples. Each tuple is 100 bytes. `Id` is the id of the product involved and is 25 bytes. `Price` is the price of this model and is 10 bytes.

Assume each company has 3 products and each product by a given company has 3 models.

Compare two different strategies for organizing the records on disk. Assume a disk block holds 4096 bytes, with 96 bytes per block devoted to header information. Records are not split across blocks.

Sequential Organization: All `Company` records are placed sequentially, in order of company number, in one set of blocks. All `Product` records are placed sequentially, in order of company number, in another set of blocks. All `Model` records are placed sequentially, in order of product id, in a third set of blocks.

Clustered Organization: For each `Company` record, the three `Product` records for that company and the nine `Model` records for that company reside in the same block. `Company` records are sequenced by company number.

(a) (6 pts) How many disk blocks are needed to store the three relations using the sequential organization? Justify your answer.

(b) (6 pts) How many disk blocks are needed to store the three relations using the clustered organization? Justify your answer.

(c) (6 pts) Which storage organization is better for queries in which all `Company` records need to be scanned in order of company id? Justify your answer.

Now consider indexing these relations. Suppose we want a primary index on `Company.Number`. Each index entry associates a 10 byte pointer with a key. The blocks available for use by the index hold 4096 bytes each, with 96 bytes reserved for header information.

(d) (7 pts) For the sequential organization described above, how many index blocks are needed for a *sparse* primary index? Justify your answer.

3. (20 pts total) Query Execution.

Suppose we have two unary relations, R and S with these tuples:

R: 7, 2, 9, 8, 3, 9, 1, 3, 6

S: 8, 4, 2, 1, 3, 2, 7, 3

Show the result of joining R and S using each of the following algorithms. List the results in the order that they would be output by the join algorithm.

(a) (10 pts) Naive (one tuple at a time) nested loops join. Use R for the outer loop and S for the inner loop.

(b) (10 pts) Hash join. Assume there are two hash buckets, numbered 0 and 1, and that the hash function sends even values to bucket 0 and odd values to bucket 1. In the second phase, assume that bucket 0 is processed before bucket 1 and that the contents of a bucket are read in the same order as they were written.

4.(25 pts total) Query Compilation.

Consider this schema for an on-line bookstore:

```
Cust (CustID, Name, Address, State, Zip)
Book (BookID, Title, Author, Price, Category)
Order (OrderID, CustID, BookID, ShipDate)
```

(a) (8 pts) For the following relational algebra expression (Find books by JK Rowling bought by customers in Texas for under $20), show the initial logical query plan, drawn as a tree of relational operators with relations at the leaves.

$\pi_{Title,BookID}(\sigma_{State='TX'ANDAuthor='JKRowling'ANDPrice>20}(Cust \bowtie Order \bowtie Book))$

(b) (8 pts) Transform your logical query plan by pushing selections and projections down as far as you can. Show the result as another logical query plan, drawn as a tree of relational operators with relations at the leaves

(c) (9 pts) Consider the following logical query plan, which finds the names of all customers who ordered a book that was shipped after Jan 1, 2005.

Label each relational operator in the tree with the expected result size in terms of the number of tuples. Assume Containment of Value Sets and Preservation of Value Sets. Use the heuristics discussed in lecture, which are from the textbook.

Use these statistics:

- $T(\texttt{Order}) = 30,000.$
  $V(\texttt{Order,OrderID}) = 1,000.$
  $V(\texttt{Order,CustID}) = 3,000.$
  $V(\texttt{Order,BookID}) = 1,000.$
  $V(\texttt{Order,ShipDate}) = 2,500.$
- $T(\texttt{Cust}) = 3,000.$
  $V(\texttt{Cust,CustID}) = 3,000.$
  $V(\texttt{Cust,Name}) = 2,750.$
- $T(\texttt{Book}) = 7,000.$