University of Waterloo
Midterm Examination #1

Spring, 2007

Student Name:	
Student ID Number:	

Course Abbreviation and Number	CS448/648
Course Title	Database Systems Implementation
Sections	01 (09:30)
Instructor	K. Salem

Date of Exam	June 11, 2007
Time Period	09:30-10:20
Duration of Exam	50 minutes
Number of Exam Pages (including this cover sheet)	6 pages
Exam Type	Closed Book
Additional Materials Allowed	None

Question 1: (12 marks)	Question 2: (8 marks)	Question 3: (10 marks)	Question 4: (5 marks)	Question 5: (5 marks)
Total: (40 marks)				

1. (12 marks)

Suppose that a database includes two relations, R and S. Relation R has N_R tuples and occupies B_R blocks on the disk. Relation S has N_S tuples and occupies B_S blocks on the disk.

The database system needs to process a query which involves an equijoin of R and S. The system is considering 4 different plans for processing this query. For each of these plans, which are listed below, give an estimate of the total I/O cost of the plan, ignoring the cost of materializing the join output. In each case, you should assume that the join operator has sufficient memory available to hold M blocks of data, where $B_R < M < B_S$ and $B_S < M^2$. Express your answers in terms of B_R, B_S, N_R, N_S, and M. Briefly explain your answers.

Plan A: (3 marks) Block-oriented nested-loop join with S as the inner relation. Both R and S are accessed using table scans.

Plan B: (3 marks) Block-oriented nested-loop join with R as the inner relation. Both R and S are accessed using table scans.

Plan C: (3 marks) Basic hash join (i.e., Grace hash join, not hybrid hash join) with R as the build relation and S as the probe relation. Both R and S are accessed using table scans.

Plan D: (3 marks) Basic hash join with S as the build relation and R as the probe relation. Both R and S are accessed using table scans.

2. (8 marks)

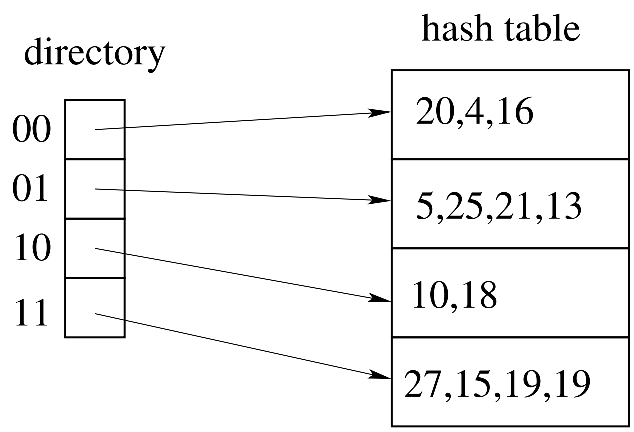
For this question, consider a B+-tree defined on a database relation. The B+-tree's leaves hold RIDs, not the actual tuples. The size of the index key values is such that N (key,pointer) or (key,RID) pairs can be stored in each block of the index.

a. (3 marks)
What is the maximum relation size (i.e., the maximum number of tuples) that can be indexed using a B+-tree of height 3, and what is the total number of index blocks in such a B+-tree?

b. (5 marks)
Suppose that the B+-tree is modified to that it implements prefix compression of the index keys. The effect of the compression is that αN ($\alpha > 1$) (key,pointer) values can fit into an index node. Suppose that such a prefix-compressed index is used to index a relation of the same maximum size that you determined in part (a). Assuming that all index nodes are completely full (except possibly the root node), what is the the total number of index blocks in the prefix-compressed index?

3. (10 marks)

The diagram below shows an extensible hash table with four hash buckets. Each number x in the buckets represents an entry for a record for which the hashed key value is x. For example, the number 20 in the first hash bucket represents a record for which the hashed key value is 20.



Draw a similar diagram showing what this extensible hash table will look like after the following sequence of records is inserted: 17, then 24, then 7. Again, each number in the insertion sequence represents the hashed key value of a record to be inserted.

4. (5 marks)

a. (3 marks)
Briefly explain the distinctions between the RAID0, RAID1, and RAID5 disk organizations.

b. (2 marks)
Briefly explain the difference between a clustered index and an unclustered index.

5. (5 marks)

Suppose that a database management system uses a B+-tree index of height 3 to index a relation. Assume that the leaves of the index hold tuples from the relation, not RIDs. In the *worst case*, what is the I/O cost of inserting a single record into the relation using this B+-tree? Indicate separately how many block read operations and how many block write operations will be required, and justify your answer. Assume that the system has a large buffer cache, but that initially the cache does not contain any of the B+-tree's nodes.