CS54100 - Homework 2

FALL 2015

Due: Tuesday, October 20, 2015 in class, before class starts.

(There will be a 10% penalty for each late day. After 5 late days, the homework will not be accepted.)

**Part 1: Tree-Based Indexing (30 Points)**

1. Show the result of inserting 17, 35, 20, 34, 6, 23, 8, 18, 40, 23, 32 and 13 into an initially empty B+-tree of order 3.
2. Show the result of deleting 20, 13, 35 from the previous B+-tree. Show the B+-tree after each deletion.
3. Assume that you have just built a dense B+ tree index using Alternative (2) on a heap file containing 20,000 records. The key field for this B+ tree index is a 40-byte string, and it is a candidate key. Pointers (i.e., record ids and page ids) are (at most) 10-byte values. The size of one disk page is 1000 bytes. The index was built in a bottom-up fashion using the bulk-loading algorithm, and the nodes at each level were filled up as much as possible. Answer the following:
   1. How many levels does the resulting tree have? Show your steps.
   2. For each level of the tree, how many nodes are at that level? Show your steps.
   3. How many levels would the resulting tree have if key compression is used and it reduces the average size of each key in an entry to 10 bytes? Show your steps.
4. If your database system supported both a static and a dynamic tree index (e.g, ISAM and B+ trees), would you ever consider using the static index in preference to the dynamic index? why?

**Part 2: Hash-based Indexing (25 Points)**

Give examples of the following by showing the insertions step-by-step:

1. A Linear Hashing index and an Extendible Hashing index with the same data entries, such that the Linear Hashing index has more pages.
2. A Linear Hashing index and an Extendible Hashing index with the same data entries, such that the Extendible Hashing index has more pages.

State clearly any assumptions you may assume.

**Part 3: Query Evaluation (20 Points)**

Consider the following schema with the Employees relation:

Employees( EmpId: Integer, EmpName: String, Salary: Real)

For each of the following indexes, list whether the index matches the given selection conditions. If there is a match, list the primary conjuncts. Please state clearly any assumptions you may assume.

1. A hash index on the search key (Employees.EmpId).

(a) *σ Employees.EmpId < 40,000 (Employees)*

(b) *σ* Employees*.EmpId = 20,000 (Employees)*

2. A B+-tree index on the search key (Employees.EmpId).

(a) *σ* Employees.EmpId *< 10,000 (Employees)*

(b) *σ* Employees.EmpId *= 10,000 (Employees)*

3. A hash index on the search key (Employees.EmpId, Employees.Salary).

(a) *σ Employees.EmpId < 10,000 And Employees.Salary = 4000 (Employees)*

(b) *σ Employees.EmpId=1500 And Employees.Salary > 3500 (Employees)*

(c) *σ Employees.EmpId = 10,000 (Employees)*

(d) *σ Employees.Salary =4000 (Employees)*

4. A B+-tree index on the search key (Employees.EmpId, Employees.Salary).

(a) *σ Employees.EmpId < 55,000 And Employees.Salary = 7000 (Employees)*

(b) *σ Employees.EmpId = 3,000 And Employees.Salary > 5000 (Employees)*

(c) *σ Employees.EmpId = 4 (Employees)*

(d) *σ Employees.Salary =5500 (Employees)*

**Part 4: External Sorting (25 Points)**

Assume you have a file of 200,000,000 records, each of size 128 Bytes. A disk page is of size 2048 bytes. We want to sort this file.

1. Calculate the total number of disk page accesses needed to perform the sort when using:
   1. Two-way external Merge Sort. State the number of passes needed. Show your steps.
   2. The general external sort-merge algorithm, assuming a buffer of 12 pages. What is the number of passes needed? Show your steps.
2. What is the minimum number of buffer pages do you need to sort this file in just three passes using the general external sort-merge algorithm? Show your steps.