CS 6360.002/003 Assignment-5

Due Date: Nov 17, 2017, 11:59PM

1. Consider a disk with block size B=512 bytes. A block pointer is P=6 bytes long,
and a record pointer is P R =7 bytes long. A file has r=3000 EMPLOYEE records
of fixed-length. Each record has the following fields: NAME (30 bytes), SSN (10
bytes), DEPARTMENTCODE (10 bytes), ADDRESS (30 bytes), PHONE (10 bytes),
BIRTHDATE (10 bytes), GENDER (1 byte), JOBCODE (4 bytes), SALARY (4 bytes, real
number). An additional byte is used as a deletion marker.

(70 points)

(a) Calculate the record size R in bytes.
110 bytes

(b) Calculate the blocking factor bfr and the number of file blocks b assuming an
unspanned organization.
Bfr = 4
Number of blocks = 750

(c) Suppose the file is ordered by the key field SSN and we want to construct a primary index on SSN. Calculate (i) the index blocking factor bfr i; (ii) the number of first-level index entries and the number of first-level index blocks; (iii) the number of levels needed if we make it into a multi-level index; (iv) the total number of blocks required by the multi-level index; and (v) the number of block accesses needed to search for and retrieve a record from the file--given its SSN value--using the primary index.
Bfri = 32
Number of levels = 2
Number of blocks required by the multi-level index = 25
Number of block accesses = 3

(d) Suppose the file is not ordered by the key field SSN and we want to construct a
secondary index on SSN. Repeat the previous exercise (part c) for the secondary
index and compare with the primary index.
Number of levels = 3

Number of blocks required by the multi-level index = 98
Number of block accesses = 4

(e) Suppose the file is not ordered by the non-key field DEPARTMENTCODE and we want to construct a secondary index on DEPARTMENTCODE using Option 3 of Section 18.1.3, with an extra level of indirection that stores record pointers. Assume there are 100 distinct values of DEPARTMENTCODE, and that the EMPLOYEE records are evenly distributed among these values. Calculate (i) the index blocking factor bfr i;
(ii) the number of blocks needed by the level of indirection that stores record pointers; (iii) the number of first-level index entries and the number of first-level index blocks; (iv) the number of levels needed if we make it a multi-level index; (v) the total number of blocks required by the multi-level index and the blocks used in the extra level of indirection; and (vi) the approximate number of block accesses needed to search for and retrieve all records in the file having a specific DEPARTMENTCODE value using the index.

Number of levels = 2
Number of blocks required by the multi-level index = 5
Number of block accesses = 33

(f) Suppose the file is ordered by the non-key field DEPARTMENTCODE and we want to construct a clustering index on DEPARTMENTCODE that uses block anchors (every new value of DEPARTMENTCODE starts at the beginning of a new block). Assume there are 100 distinct values of DEPARTMENTCODE, and that the EMPLOYEE records are evenly distributed among these values. Calculate (i) the index blocking factor bfr i; (ii) the number of first-level index entries and the number of first-level index blocks; (iii) the number of levels needed if we make it a multi-level index; (iv) the total number of blocks required by the multi-level index; and (v) the number of block accesses needed to search for and retrieve all records in the file having a specific DEPARTMENTCODE value using the clustering index (assume that multiple blocks in a cluster are either contiguous or linked by pointers).

Number of levels = 2
Number of blocks required by the multi-level index = 5
Number of block accesses = 10

(g) Suppose the file is not ordered by the key field Ssn and we want to construct a B+ tree access structure (index) on SSN. Calculate (i) the orders p and p leaf of the
B+ tree; (ii) the number of leaf-level blocks needed if blocks are approximately
69% full (rounded up for convenience); (iii) the number of levels needed if internal nodes are also 69% full (rounded up for convenience); (iv) the total number of blocks required by the B+ tree; and (v) the number of block accesses needed to search for and retrieve a record from the file --given its SSN value-- using the B+ tree.

2. A PARTS file with Part# as key field includes records with the following Part# values: 23, 65, 37, 60, 46, 92, 48, 71, 56, 59, 18, 21, 10, 74, 78, 15, 16, 20, 24, 28, 39, 43, 47, 50, 69, 75, 8, 49, 33, 38. Suppose the search field values are inserted in the given order in a B+ tree of order p=4 and p-leaf =4; show how the final tree looks like. **(20 points)**

3. Optimize the following SQL query on the Company Database to find names of employees earning over $80,000 per year, names of projects in which they work more than 30 hours, where the project is located in Chicago and the manager of the project's controlling department started after January 1, 2009.

```
Select Lname, Fname, Pname, Hours
From   Project P, Employee E, Department D, Works_on W
Where  E.Ssn = W.Essn
  and  P.Dnum = D.Dnumber
  and  W.Pno = P.Pnumber
  and  Plocation = 'Chicago'
  and  Hours > 30
  and  Salary > 80000
  and  Mgr_start_date >= '1/1/2009'
```

Show the final query tree. **(10 points)**