EE4791 Database Systems -Tutorial 7 and Sample Answer

1. Consider a disk with block size B=512 bytes. A file has r=30,000 EMPLOYEE records of fixed-length. Each record has the following fields: NAME (30 bytes), SSN (9 bytes), DEPARTMENTCODE (9 bytes), ADDRESS (40 bytes), PHONE (9 bytes), BIRTHDATE (8 bytes), SEX (1 byte), JOBCODE (4 bytes), SALARY (4 bytes, real number). An additional byte is used as a deletion marker.

   (a) Calculate the record size R in bytes.
   (b) Calculate: (i) the blocking factor bfr and the number of file blocks b assuming an unspanned organization; (ii) the total disk space needed (no of bytes).
   (c) If the file is unordered and each record has a unique value of SSN, calculate the average number of block accesses needed for searching a record based on SSN value.
   (d) If the file is ordered by its key field SSN and no indexes are defined for the file, calculate the average number of block accesses needed for searching a record based on SSN value.

Answer:

(a) Record length R = (30 + 9 + 9 + 40 + 9 + 8 + 1 + 4 + 4) + 1 = 115 bytes

(b) Blocking factor bfr = $\lfloor B/R \rfloor$
= $\lfloor 512/115 \rfloor$
= 4 records per block

Number of blocks needed for file = $\lceil r/bfr \rceil$
= $\lceil 30000/4 \rceil$
= 7500

Disk space needed = 7500*512 = 3,840,000 bytes

(c) Since the file is unordered, linear search will be used for searching a record based on SSN value.
Hence, the average number of block accesses = (No of file blocks)/2
= 7500/2
= 3750

(d) Since the file is by its key field SSN and no indexes are defined, binary search will be used for searching a record based on SSN value.
Hence, the average number of block accesses = $\lceil \log_2(\text{No of file blocks}) \rceil$
= $\lceil \log_2 7500 \rceil$
= 13

2. For the same file described in question one, assuming that the file is ordered by the key field SSN and a primary index on SSN is defined for the file. Suppose that the disk's block size B is still 512 bytes, and a block pointer is P=6 bytes long. Calculate:

   (a) The index blocking factor (fan-out) bfr_i for the primary index.
   (b) The number of first-level index entries and number of first-level index blocks.
   (c) The number of levels needed if we make it into a multi-level index.
   (d) The total number of blocks required by all the index entries in the multilevel index structure.
   (e) The number of block accesses needed to search for and retrieve a record from the file, given its SSN value, using the primary index if it is implemented as:
      (i) single-level index structure
      (ii) multilevel index structure
   (f) The percentage of improvement on number of block accesses achieved by:
      (i) 1(d) from 1 (c), and
      (ii) Implementing the above-mentioned primary index as a single-level index from 1(d)
      (iii) Implementing the above-mentioned primary index as multilevel index from implementing as a single-level index

Answer:

(a) Index entry size R_i = size of primary index + P
= size of SSN + P = 9 + 6 = 15 bytes
Index blocking factor bfr_i for the primary index (fan-out, *fo*) = $\lfloor B/R_i \rfloor$
$= \lfloor 512/15 \rfloor$
= 34

(b) The number of first-level index entries = number of file blocks
= 7500
The total number of index blocks for the primary index, b_1
$= \lceil$ No of file blocks/Index blocking factor $\rceil$
$= \lceil 7500/34 \rceil$
= 221

(c) Number of second-level index blocks $b_2 = \lceil b_1/fo \rceil = \lceil 221/34 \rceil = 7$ blocks.
Number of third-level index blocks $b_3 = \lceil b_2/fo \rceil = \lceil 7/34 \rceil = 1$ block.
The third level is the top level of the multilevel index, that is, $t = 3$.

(d) The total number of blocks required by all the index entries in the multilevel index structure = 221+ 7+1 = 229.

(e) (i) To search for and retrieve a record from the file, given its SSN value, using the primary index, implemented as single-level index structure , will require: (i) perform a binary search for the required index record; (ii) based on the block pointer in the index record to read the record in the data file.

Hence, the number of block accesses needed to search for and retrieve a record from the file, given its SSN value, using the primary index, implemented as single-level index structure

$= \lceil \log_2(\text{No of index blocks}) \rceil + 1$
$= \lceil \log_2 221 \rceil + 1$
$= 9$

(ii) To search for and retrieve a record from the file, given its SSN value, using the primary index, implemented as multilevel index structure, will require to read one index block at each level and based on the block pointer in the base level (first level) to read the record in the data file.

Hence, the number of block accesses needed to search for and retrieve a record = 3 +1 = 4

(f) (i) The percentage of improvement achieved by 1(d) from 1(c)

= (3750-13)/3750
= 99.65%.

(ii) The percentage of improvement achieved by implementing the above-mentioned primary index as a single-level index from 1(d)

= (13-9)/13
= 30.76%.

(iii) The percentage of improvement achieved by implementing the above-mentioned primary index as multilevel index from implementing as a single-level index

= (9-4)/9
= 55.55%.

3. Suppose the file is not ordered by the key field SSN and we want to construct a secondary index on SSN. Repeat the calculation from 2(a) to 2(e) for the secondary index.

Answer:

(a) Index entry size R_i = size of secondary index + P
= size of SSN + P
= 9 + 6 = 15 bytes
Index blocking factor bfr_i for the secondary index = $\lfloor B/R_i \rfloor$
$= \lfloor 512/15 \rfloor$
= 34

(b) The number of first-level index entries = number of file records
= 30000.
The total number of first-level index blocks for the secondary index, b_1
$= \lceil$ No of EMPLOYEE records/Index blocking factor $\rceil$
$= \lceil 30000/34 \rceil$
= 883.

(c) Number of second-level index blocks $b_2 = \lceil b_1/fo \rceil = \lceil 883/34 \rceil = 26$ blocks.
Number of third-level index blocks $b_3 = \lceil b_2/fo \rceil = \lceil 26/34 \rceil = 1$ block.
The third level is the top level of the multilevel index, that is, t =3.

(d) The total number of blocks required by all the index entries in the multilevel index structure =883+ 26+1 = 910.

(e) (i) To search for and retrieve a record from the file, given its SSN value, using the secondary index, implemented as single-level index structure, will require: (i) perform a binary search for the required index record; (ii) based on the block pointer in the index record to read the record in the data file.

Hence, the number of block accesses needed to search for and retrieve a record from the file, given its SSN value, using the secondary index, implemented as single-level index structure

$= \lceil \log_2(\text{No of index blocks}) \rceil + 1$

$= \lceil \log_2 883 \rceil + 1$

$= 11$

(ii) To search for and retrieve a record from the file, given its SSN value, using the primary index, implemented as multilevel index structure will require to read one index block at each level and based on the block pointer in the base level (first level) to read the record in the data file.

Hence, the number of block accesses needed to search for and retrieve a record = 3 +1 = 4

4. For the same file described in question one, suppose the file is not ordered by the non-key field DEPARTMENTCODE and we want to construct a secondary index on DEPARTMENTCODE with one level of indirection that stores record pointers. Assume there are 1000 distinct values of DEPARTMENTCODE and that the EMPLOYEE records are evenly distributed among these values. Suppose that the disk's block size B is still 512 bytes, a block pointer is P=6 bytes and a record pointer is P_R =7 bytes long. Calculate:

   (a) The index blocking factor bfr_i.
   (b) The number of blocks needed by the level of indirection that stores record pointers.
   (c) The number of first-level index entries and the number of first-level index blocks.
   (d) The approximate number of block accesses needed to search for and retrieve all records in the file having a specific DEPARTMENTCODE value using the secondary index implemented in single-level index structure.
   (e) The number of levels needed if we make it into a multi-level index.
   (f) The total number of blocks required by all the index entries including the blocks required by the level of indirection level in the multilevel index structure.
   (g) The number of block accesses needed to search for and retrieve a record from the file, given its DEPARTMENTCODE value, using the secondary index implemented as multilevel index structure with the one level of indirection.

Answer:

(a) Index entry size R_i = size of secondary index + P
= size of DEPARTMENTCODE + P
= 9 + 6 = 15 bytes

Index blocking factor bfr_i for the primary index = $\lfloor B/R_i \rfloor$
= $\lfloor 512/15 \rfloor$ = 34

(b) The average no. of employee record for each department value
= (30000/1000)
= 30
Since a record pointer is 7 bytes, the number of bytes needed at the level of indirection for each value of DEPARTMENTCODE is 7 * 30 =210 bytes, which fits in one block.
Hence, the no. of blocks needed for the level of indirection = no of DEPARTMENTCODE's value = 1000.

(c) The no of first-level index entries = no of distinct DEPARTMENTCODE's values = 1000
The total number of first-level index blocks for the index
$= \lceil$ No of DEPARTMENTCODE's value/Index blocking factor $\rceil$
$= \lceil 1000/34 \rceil$
= 30

(d) No of block accesses needed for searching the secondary INDEX implemented in single-level index structure
$= \lceil \log_2(\text{No of index blocks}) \rceil$
$= \lceil \log_2 30 \rceil$
= 5
If we assume that the 30 records are distributed over 30 distinct blocks, we need an additional 30 block accesses to retrieve all 30 records.
Hence, total block accesses needed on average to retrieve all the records with a given value for DEPARTMENTCODE = 5 + 1+ 30 = 36

(e) Number of second-level index blocks $b_2 = \lceil b_1/fo \rceil = \lceil 30/34 \rceil = 1$ block.
The second level is the top level of the multilevel index, that is, t =2.

(f) The total number of blocks required by all the index entries in the multilevel index structure including the blocks for the indirection =30+ 1+1000 = 1031.

(g) To search for and retrieve a record from the file, given its SSN value, using the secondary index, implemented as multilevel index structure with one level of indirection, will require to read one index block at each level and based on the block pointer in the base level index (first level) to read the block of record pointers and then based on these pointers to read all the target records in the data file.

As on the average, each department code has 30 employees, there are altogether 30 block accesses to read all the target records in the file.

Hence, the number of block accesses needed to search for and retrieve a record = 2 + 1 + 30=32.

5. For the same file described in question one with block size B = 512, block pointer size P = 6 bytes and record pointer size Pr = 7 bytes, assuming that the file is not ordered by the key field SSN and we want to construct a B+-tree access structure (index) on SSN. Calculate:

   (a) The orders p and p_{leaf} of the B+-tree.
   (b) The number of leaf-level blocks needed if blocks are approximately 69% full.
   (c) The number of levels needed if internal nodes are also 69% full.
   (d) The total number of blocks required by the B+-tree.
   (e) The number of block accesses needed to search for and retrieve a record from the file--given its SSN value--using the B+-tree.

Answer:

(a) Let V = the size of SSN

An internal node of B+-tree can have up to p pointers and p-1 search values SSN. Hence,

$(p*P) + ((p - 1)* V) \leq B$, where V= size of SSN

$(p*6) + ((p - 1)*9) \leq 512$

$15p \leq 521$

We can choose the largest p = 34 satisfying the above inequality

A leaf node has the same number of search values and record pointers with a block pointer to the next node. Hence,

$p_{leaf}*(Pr+V) + P \leq B$

$p_{leaf}*(7+9) + 6 \leq 512$

$16*p_{leaf} \leq 506$

We can choose the largest $p_{leaf} = 31$ satisfying the above inequality

(b) Since leaf nodes are 69% full on the average, the average number of key value and record pointer pairs in a leaf node is $0.69*p_{leaf} = 0.69*31 = 21.39 \approx 21$.

Since the file has 30000 records and hence 30000 values of SSN, the number of leaf-level blocks needed is $b_1 = \lceil 30000/21 \rceil = 1429$ blocks.

(c) Since internal nodes are 69% full on the average, the average number of tree pointers in each internal node = 0.69*(p -1) = 0.69*34 = 23.46 ≈ 23. And, the average no of search values in each internal node = 22.

The number of second-level blocks needed is $b_2 = \lceil 1429/23 \rceil = 63$ blocks.

The number of third-level blocks needed is $b_3 = \lceil 63/23 \rceil = 3$ blocks.

The number of fourth-level blocks needed is $b_4 = \lceil 3/23 \rceil = 1$ blocks.

The number of levels = 4.

(d) The total number of blocks required by the by the B^+-tree
= 1429 +63 +3 + 1 =1496 blocks.

(e) To search for and retrieve a record from the file--given its SSN value--using the B+-tree, will require to read one block at each level and based on the data pointer in the leaf level to read the record in the data file.

Hence, the number of block accesses needed to search for and retrieve a record = 4 +1 = 5

6. In Question 5, if we want to construct a B-tree access structure (index) on SSN instead of B+-tree, calculate the following:

   (a) The order p of the B-tree.
   (b) The number of levels needed if the B-tree is 69% full.
   (c) The total number of blocks required by the B-tree.
   (d) The maximum number of block accesses needed to search for and retrieve a record from the file--given its SSN value--using the B-tree.
   (e) Can we calculate the number of block assesses instead of maximum number of block accesses for 6(e)? If not, why?

Answer:

(a) Let V = the size of SSN
An internal node of B-tree can have up to p pointers and p-1 search values SSN. Hence,
$(p*P) + ((p - 1)*(Pr+V)) \leq B$, where V= size of SSN
$(p*6) + ((p - 1)*(7+9)) \leq 512$
$22p \leq 528$
We can choose the largest p = 24 satisfying the above inequality

(b) Since the B-tree is approximately 69% full, the average number of block pointers in each node = 0.69*p = 0.69*24 = 16.56 ≈ 16 (approximate, we can round up or down, doesn't matter).

Level	No of nodes	No of search values	No of tree pointers(blk pointers
Root:	1 node	15 search values	16 tree pointers (blk pointers)
Level-1:	16 nodes	240 search values (16*15)	256 tree pointers
Level-2:	256 nodes	3840 search values (256*15)	4096 tree pointes (256*16)
Level -4:	4096 nodes	61440 search values (4096*15)	4096*16 tree pointers

Total no of search values stored in the nodes in three-level B-tree (the root, level-1 and level-2) = 15 + 240 + 3840 =4095.

Total no of search values stored in the nodes in four-level B-tree (the root, level-1, level-2 and level-3) = 4095 + 61440 = 65535.

Since the file has 30000 records and hence 30000 values of SSN, hence, altogether, we have 30000 of search values to be stored in a B-tree. Since 4095 < 30000 < 65535, the number of levels = 4.

(c) Since there are altogether 30000 of search values, the total number of blocks required by the by the B-tree = 30000/15 = 2000 blocks.

(d) To search for and retrieve a record from the file--given its SSN value--using the B-tree, at most, we will require to read one block at each level and based on the data pointer in the leaf level to read the record in the data file.

Hence, the maximum number of block accesses needed to search for and retrieve a record = 4 +1 = 5.

(e) No. It is because the search key value might be equal to the search key value of an internal node at any level. In such a case, we will base on the

data pointer in the node to read the required record. Hence, we will not require to read a block at each level.