Aisha Muhammad Nawaz (20L-0921)

Advance Database Concepts

Assignment # 3 (BSCS 6A)

Take the following assumptions for the block size and file size to solve the questions:
Q1. Block Size B=4096 bytes and File Records r=10million

A block pointer is P = 6 bytes long and a record pointer is PR = 7 bytes long. A file has above records of fixed length (un-spanned). Each record has the following fields:

NAME (30 bytes), SSN (9 bytes), DEPARTMENTCODE (9 bytes), ADDRESS (40bytes), 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. Suppose that the file is ordered by the key field SSN and we want to construct a primary index on SSN.

ANSWER Q1(a):
(i) the index blocking factor bfri (which is also the index fan-out fa):

Index Record Size, Ri= (Vssn + Pb) = 9+6=15 Bytes
B=4096 bytes
bfri=floor[B/(Ri)]=floor[4096/15]=273 records

(ii) the number of first-level index entries and the number of first-level index blocks

In primary index, number of index entries = number of blocks of base table
ri=b
First calculating number of blocks in base table,
B= 4096 Bytes,
Size of one Records, R = 30+9+9+40+9+8+1+4+4+1 =115 Bytes
Number of records in one block, bfr =floor(B/R)= floor (4096/115)= 35 records
Total file records, r= 10 million (10 000 000)
Number of blocks of base table, b=ceiling[r/bfr]=ceiling(10000000/35) = 285715 blocks
so ri=285715
& Using bfri from previous part, Number of first-level index blocks=ceiling[ri/bfri]=
ceiling[285715/273]=1047 blocks

Hence,
Number of first-level index entries, ri= 285715
Number of first-level index blocks, bi= 1047

(iii) the number of levels needed if we make it into a multilevel index:

First Level has blocks, b1 = 1047
Second Level has number of index entries r2 = b1= 1047
& the number of blocks, b2 = ceiling[b1/bfri]= ceiling[1047/273]= 4 blocks
Third Level has number of index entries = b2 = 4
& the number of blocks , b3 = ceiling [b2/brfi] =[4/273]= 1 Block
As number of blocks =1 so Top level is 3rd

Hence, index has total Number of Levels = 3 Levels

(iv) the total number of blocks required by the multilevel index:
Using Calculations from previous step,
Total number of blocks required by the multilevel index = b1+b2+b3 = 1047 + 4 + 1 = 1052 blocks

(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:
Number of block accesses = x + 1 = 3 + 1 = 4 block accesses
Where x is number of levels.

b. Suppose that the file is not ordered by the key field SSN and we want to construct a secondary index on SSN. Repeat the previous (part a) for the secondary index and compare with the primary index.

ANSWER Q1(b):
(i) the index blocking factor bfri (which is also the index fan-out fa):
Index Record Size, Ri= (Vssn + Pb) = 9+6=15 Bytes
B=4096 bytes
bfri=floor[B/(Ri)]=floor[4096/15]=273 records

(ii) the number of first-level index entries and the number of first-level index blocks
In Secondary index, number of index entries = number of distinct values of record
As SSN is unique so number of distinct values of record will be equal to number of records
i.e r=d=10 000 000 = ri
Using bfri from previous part, Number of first-level index blocks=ceiling[ri/bfri]=
ceiling[10000000/273]=36631 blocks

Hence,
Number of first-level index entries, ri= 10 000 000
Number of first-level index blocks, bi= 36631

(iii) the number of levels needed if we make it into a multilevel index:
First Level has blocks, b1 = 36631
Second Level has number of index entries r2 = b1= 36631
& the number of blocks, b2 = ceiling[b1/bfri] = ceiling [36631/273] = 135 blocks
Third Level has number of index entries = b2 = 135
& the number of blocks, b3 = ceiling [b2/brfi] = ceiling [135/273] = 1 Block

As number of blocks =1 so Top level is 3rd

Hence, index has total Number of Levels = 3 Levels

(iv) the total number of blocks required by the multilevel index:
Using Calculations from previous step,
Total number of blocks required by the multilevel index = b1+b2+b3 = 36631 + 135 + 1 = 36767 blocks

(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:
Number of block accesses = x + 1 = 3 + 1 = 4 block accesses

c. Suppose that the file is not ordered by the non-key field DEPARTMENTCODE and we want to construct a secondary index on DEPARTMENTCODE, with an extra level of indirection that stores record pointers. Assume there are 20000 distinct values of DEPARTMENTCODE and that the EMPLOYEE records are evenly distributed among these values.

ANSWER Q1(C):
(i) the index blocking factor bfr, (which is also the index fan-out fa):
Index Record Size, Ri = (Vdcode + Pb) = 9+6=15 Bytes
B=4096 bytes
bfri=floor[B/(Ri)]=floor[4096/15]=273 records

(ii) the number of blocks needed by the level of indirection that stores record pointers:
Total Number of records, r = 10 000 000
Number of distinct records, d = 20000
Number of same values/Number of records for each department code, s = 10000000/20000 = 500
Pointer for each record size = 7 bytes
Number of bytes required for all records of each value of department code = 7 *500 = 3500 bytes
Bfr for level of indirection = floor[4096/3500]=1
Hence, one block stores all records corresponding to one value of department code
So, Number of blocks needed by the level of indirection = d = 20000

(iii) the number of first level index entries and the number of first-level index blocks:
In Secondary index, number of index entries = number of distinct values of record
ri = d = 20 000
Using bfri from previous part,
Number of first-level index blocks=ceiling[ri/bfri]= ceiling[20000/273]=74 blocks

Hence,
Number of first level index entries = 20000
Number of first-level index blocks, bi= 74

(iv) the number of levels needed if we make it into a multilevel index:
First Level has blocks, b1 = 74
Second Level has number of index entries r2 = b1= 74
& the number of blocks, b2 = ceiling[b1/bfri]= ceiling[74/273]= 1 block

As number of blocks =1 so Top level is 2nd

Hence, index has total Number of Levels = 2 Levels

(v) the total number of blocks required by the multilevel index and the blocks used in the extra level of indirection:

b=b1+b2+bindirection = 74 + 1 + 20000 = 20075 blocks

(vi) the approximate number of block accesses needed to search for and retrieve all records in the file that have a specific DEPARTMENTCODE value, using the index:

Block accesses = x + 1 + s = 2 + 1 + 500 = 503 (Assuming in worst case the 500 records corresponding to one distinct department code are distributed over 500 different blocks)

d. Suppose that 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 20000 distinct values of DEPARTMENTCODE and that the EMPLOYEE records are evenly distributed among these values.

ANSWER Q1(d):
(i) the index blocking factor bfr, (which is also the index fan-out fa):
Index Record Size, Ri = (Vdcode + Pb) = 9+6=15 Bytes
B=4096 bytes
bfri=floor[B/(Ri)]=floor[4096/15]=273 records

(ii) the number of first-level index entries and the number of first-level index blocks:
In Clustering index, number of index entries = number of distinct values of record
ri = d = 20 000
Number of first-level index blocks= ceiling [ri/bfri]=ceiling[20000/273] = 74

Hence,
Number of first level index entries = 20 000
Number of first-level index blocks, bi= 74

(iii) the number of levels needed if we make it into a multilevel index:
First Level has blocks, b1 = 74
Second Level has number of index entries r2 = b1= 74
& the number of blocks, b2 = ceiling[b1/bfri]= ceiling[74/273]= 1 block
As number of blocks =1 so Top level is 2^{nd}

Total Number of Levels for index = 2 Levels

(iv) the total number of blocks required by the multilevel index:

b= b1 + b2 = 74 + 1 = 75

(v) the number of block accesses needed to search for and retrieve all records in the file that have a specific DEPARTMENTCODE value, using the clustering index (assume that multiple blocks in a cluster are contiguous):

Block accesses = x + 1 = 2 + 1 (To get first record)
For the others:
Number of records with the same department code, s = 10 000 000 / 200000 = 500
The 500 records are clustered in ceiling [s/bfr] blocks.
So,
X + ceiling [s/bfr] = 2 + ceiling [500 /35] = 17
Hence,
total block accesses needed on average to retrieve all the records with a given Department code = 17

e. Suppose the file is not ordered by the key field Ssn and we want to construct a B+-tree access structure (index) on SSN.

ANSWER Q1(e):

(i) the orders p and p leaf of the B+-tree:
Block Pointer size, Pb = 6 bytes
Record Pointer size, Pr = 7 bytes
Block size, B = 4096 bytes
Vssn = 9 bytes

Finding Order p:
$p(Pb)+(p-1)(Vssn)<B$
$p(6)+(p-1)(9)<4096$
$6p + 9p - 9 < 4096$
$p<(4096+9)/(6+9)$
$p<273.667$
hence, p= 273

Finding p leaf:
$pleaf(Vssn +Pr)+Pb<4096$
$pleaf (9 + 7) + 6 < 4096$
$pleaf<(4096-6)/(9+7)$
$pleaf<255.625$
hence, pleaf = 255

(ii)the number of leaf-level blocks needed if blocks are approximately 69% full (rounded up for convenience):
Number of records in leaf nodes = 0.69*pleaf = 0.69 * 255 =175.95
= 176 (Rounded up for convenience)
Total Number of records: 10 000 000
Number leaf – level blocks needed = ceiling [10 000 000 /176] = 56819 blocks

(iii) the number of levels needed if internal nodes are also 69% full (rounded up for convenience):
Average fan out, fo, for internal nodes = 0.69 * (p) = 0.69 * (273) = 188.37
= 189 (Rounded up for convenience)

Number of records in 2nd Level = Number of blocks in leaf-level nodes = b1 = 56819 blocks
Number of blocks in 2nd Level, b2 = ceiling [56819/189] = 301

Number of records in 3rd Level = Number of blocks in 2nd Level = b2 = 301
Number of blocks in 3rd Level, b3 = ceiling [301/189] = 2

Number of records in 4th Level = Number of blocks in 3rd Level = b3 = 2
Number of blocks in 4th Level, b4 = ceiling [2/189] = 1
As number of blocks =1, This is the top Level

Number of Levels needed = 4

Alternatively, ceiling [Log189(39216)] + 1 = 3 + 1 = 4

(iv) the total number of blocks required by the B+-tree:
b = b1 + b2 + b3 + b4 = 56819 + 301 + 2 +1 = 57123 blocks

(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:
Will have to traverse till leaf nodes
Block accesses = x +1 = 4 +1 = 5 blocks
Where x is number of levels.

f. Repeat (part e), but for a B-tree rather than for a B+-tree. Compare your results for the B-tree and for the B+-tree.

ANSWER Q1(f):

(i) the orders p and p leaf of the B-tree:
Block Pointer size, Pb = 6 bytes
Record Pointer size, Pr = 7 bytes
Block size, B = 4096 bytes
Vssn = 9 bytes

Finding Order p:
p(Pb)+(p-1)(Vssn+Pr)<B
p(6)+(p-1)(9+7)<4096
6p + 16p -16 < 4096
p<(4096+16)/(6+16)
p<186.90909
hence, p=186

Finding p leaf:
There is no p leaf in B – tree.

COMAPRISION: order of p is less for B-tree compared to B+-tree as internal nodes also store record pointers in B-tree

(ii)the number of leaf-level blocks needed if blocks are approximately 69% full (rounded up for convenience):
Number of records in leaf nodes = 0.69*(p)= 0.69 * (186) = 128.34
= 129 (Rounded up for convenience)
Total Number of records: 10 000 000
Number leaf – level blocks needed = ceiling [10 000 000 /129] = 77520 blocks

COMAPRISION: Number leaf – level blocks needed is more for B-tree compared to B+-tree as bfr is less of B-tree

(iii) the number of levels needed if internal nodes are also 69% full (rounded up for convenience):
Average fan out, fo, for internal nodes = 129 (Rounded up for convenience)

Number of records in 2^{nd} Level = Number of blocks in leaf-level nodes = b1 = 77520 blocks
Number of blocks in 2^{nd} Level, b2 = ceiling [77520/129] = 601

Number of records in 3^{rd} Level = Number of blocks in 2^{nd} Level = b2 = 601
Number of blocks in 3^{rd} Level, b3 = ceiling [601/129] = 5

Number of records in 4^{th} Level = Number of blocks in 3^{rd} Level = b3 = 5
Number of blocks in 4^{th} Level, b4 = ceiling [5/129] = 1
As number of blocks =1, This is the top Level

Number of Levels needed = 4

Alternatively, ceiling [Log129(77520)] + 1 = 3 + 1 = 4
COMAPRISION: Number of Levels needed is same for B-tree compared to B+-tree

(iv) the total number of blocks required by the B-tree:
b = b1 + b2 + b3 + b4 = 77520 + 601 + 5 +1 = 78127 blocks

COMAPRISION: Number blocks needed is more for B-tree compared to B+-tree as bfr is less of B-tree

(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:
In worst case have to traverse all levels
Block accesses = x +1 = 4 +1 = 5 blocks
Where x is number of levels & '+ 1' is for base table access.

COMAPRISION: Number of Block accesses needed is same for B-tree compared to B+-tree

IN CONCLUSION: B+ tree is more efficient

Q2. Block Size B=8192 bytes and File Records r=10billion

a. Suppose that the file is ordered by the key field SSN and we want to construct a primary index on SSN.

ANSWER Q2(a):
(i) the index blocking factor bfri (which is also the index fan-out fa):

Index Record Size, Ri= (Vssn + Pb) = 9+6=15 Bytes
B=8192bytes
bfri=floor[B/(Ri)]=floor[8192/15]=546 records

(ii) the number of first-level index entries and the number of first-level index blocks

In primary index, number of index entries = number of blocks of base table
ri=b
First calculating number of blocks in base table,
B= 8192 Bytes,
Size of one Records, R = 30+9+9+40+9+8+1+4+4+1 =115 Bytes
Number of records in one block, bfr =floor(B/R)= floor (8192/115)= 71 records
Total file records, r= 10 billion (10*10^9)
Number of blocks of base table, b=ceiling[r/bfr]=ceiling(10*10^9/71) = 140845071 blocks
so ri=140845071
& Using bfri from previous part, Number of first-level index blocks=ceiling[ri/bfri]=
ceiling[140845071/546]=257959 blocks

Hence,
Number of first-level index entries, ri= 140845071
Number of first-level index blocks, bi= 257959

(iii) the number of levels needed if we make it into a multilevel index:

First Level has blocks, b1 = 257959
Second Level has number of index entries r2 = b1= 257959
& the number of blocks, b2 = ceiling[b1/bfri]= ceiling[257959/546]= 473 blocks
Third Level has number of index entries = b2 = 473
& the number of blocks , b3 = ceiling [b2/brfi] =[473/546]= 1 Block
As number of blocks =1 so Top level is 3rd

Hence, index has total Number of Levels = 3 Levels

(iv) the total number of blocks required by the multilevel index:

Using Calculations from previous step,
Total number of blocks required by the multilevel index = b1+b2+b3
= 257959 + 473 + 1 = 258433 blocks

(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:
Number of block accesses = x + 1 = 3 + 1 = 4 block accesses
Where x is number of levels.

b. Suppose that the file is not ordered by the key field SSN and we want to construct a secondary index on SSN. Repeat the previous (part a) for the secondary index and compare with the primary index.

ANSWER Q2(b):
(i) the index blocking factor bfri (which is also the index fan-out fa):
Index Record Size, Ri= (Vssn + Pb) = 9+6=15 Bytes
B=8192 bytes
bfri=floor[B/(Ri)]=floor[8192/15]=546 records

(ii) the number of first-level index entries and the number of first-level index blocks
In Secondary index, number of index entries = number of distinct values of record
As SSN is unique so number of distinct values of record will be equal to number of records
i.e r=d=10*10^9= ri
Using bfri from previous part, Number of first-level index blocks=ceiling[ri/bfri]=
ceiling[10*10^9/546]=18315019 blocks

Hence,
Number of first-level index entries, ri= 10 *10^9
Number of first-level index blocks, bi= 18315019

(iii) the number of levels needed if we make it into a multilevel index:
First Level has blocks, b1 = 18315019
Second Level has number of index entries r2 = b1= 18315019
& the number of blocks, b2 = ceiling[b1/bfri] = ceiling [18315019 /546] = 33544 blocks
Third Level has number of index entries = b2 = 33544
& the number of blocks, b3 = ceiling [b2/brfi] = ceiling [33544/546] = 62 Block
Fourth Level has number of index entries = b3 = 62
& the number of blocks, b4= ceiling [b3/bfri] = ceiling [62/546] = 1
As number of blocks =1 so Top level is 4th

Hence, index has total Number of Levels = 4 Levels

(iv) the total number of blocks required by the multilevel index:
Using Calculations from previous step,
Total number of blocks required by the multilevel index = b1+b2+b3+b4
= 18315019 +33544+ 62 +1 = 18348626 blocks

(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:
Number of block accesses = x + 1 = 4 + 1 = 5 block accesses

c. Suppose that the file is not ordered by the non-key field DEPARTMENTCODE and we want to construct a secondary index on DEPARTMENTCODE, with an extra level of indirection that stores record pointers. Assume there are 20000 distinct values of DEPARTMENTCODE and that the EMPLOYEE records are evenly distributed among these values.

ANSWER Q2(C):
(i) the index blocking factor bfr, (which is also the index fan-out fa):
Index Record Size, Ri = (Vdcode + Pb) = 9+6=15 Bytes
B=8192 bytes
bfri=floor[B/(Ri)]=floor[8192/15] = 546 records

(ii) the number of blocks needed by the level of indirection that stores record pointers:
Total Number of records, r = 10 * 10^9
Number of distinct records, d = 20000
Number of same values/Number of records for each department code, s = 10*10^9/20000 = 500000
Since blocks in level of indirection only store record pointer,
bfr of level of indirection, bfrl = floor [8192/7] = 1170 records
Number of blocks required for each distinct value = ceiling [500000/1170] = 428 blocks
Number of distinct values = 20 000
Total Number of blocks needed by the level of indirection is = 428 * 20 000 = 8560000 blocks

(iii) the number of first level index entries and the number of first-level index blocks:
In Secondary index, number of index entries = number of distinct values of record
ri = d = 20 000
Using bfri from previous part(i),
Number of first-level index blocks=ceiling[ri/bfri]= ceiling[20000/546]=37 blocks

Hence,
Number of first level index entries = 20000
Number of first-level index blocks, bi= 37

(iv) the number of levels needed if we make it into a multilevel index:
First Level has blocks, b1 = 37
Second Level has number of index entries r2 = b1= 37
& the number of blocks, b2 = ceiling[b1/bfri]= ceiling[37/546]= 1 block
As number of blocks =1 so Top level is 2nd

Hence, index has total Number of Levels = 2 Levels

(v) the total number of blocks required by the multilevel index and the blocks used in the extra level of indirection:

b=b1+b2+bindirection = 37 + 1 + 8560000 = 8560038 blocks

(vi) the approximate number of block accesses needed to search for and retrieve all records in the file that have a specific DEPARTMENTCODE value, using the index:

Block accesses = x + 1 + s = 2 + 1 + 500000 = 500003 (Assuming in worst case the 500000 records corresponding to one distinct department code are distributed over 500000 different blocks)

d. Suppose that 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 20000 distinct values of DEPARTMENTCODE and that the EMPLOYEE records are evenly distributed among these values.

ANSWER Q2(d):
(i) the index blocking factor bfr, (which is also the index fan-out fa):
Index Record Size, Ri = (Vdcode + Pb) = 9+6=15 Bytes
B=8192 bytes
bfri=floor[B/(Ri)]=floor[8192/15]=546 records

(ii) the number of first-level index entries and the number of first-level index blocks:
In Clustering index, number of index entries = number of distinct values of record
ri = d = 20 000
Number of first-level index blocks= ceiling [ri/bfri]=ceiling[20000/546] = 37

Hence,
Number of first level index entries = 20 000
Number of first-level index blocks, bi= 37

(iii) the number of levels needed if we make it into a multilevel index:
First Level has blocks, b1 = 37
Second Level has number of index entries r2 = b1= 37
& the number of blocks, b2 = ceiling[b1/bfri]= ceiling[37/546]= 1 block
As number of blocks =1 so Top level is 2^{nd}

Total Number of Levels for index = 2 Levels

(iv) the total number of blocks required by the multilevel index:
b= b1 + b2 = 37 + 1 = 38

(v) the number of block accesses needed to search for and retrieve all records in the file that have a specific DEPARTMENTCODE value, using the clustering index (assume that multiple blocks in a cluster are contiguous):

Block accesses = x + 1 = 2 + 1 (To get first record)
For the others:

Number of records with the same department code, s = 10*10^9/ 200000 = 500 000
The 500 000 records are clustered in ceiling [s/bfr] blocks.
So,
X + ceiling [s/bfr] = 2 + ceiling [500 000 /71] = 7043
Hence,
total block accesses needed on average to retrieve all the records with a given Department code = 7045

e. Suppose the file is not ordered by the key field Ssn and we want to construct a B+-tree access structure (index) on SSN.

ANSWER Q2(e):

(i) the orders p and p leaf of the B+-tree:
Block Pointer size, Pb = 6 bytes
Record Pointer size, Pr = 7 bytes
Block size, B = 8192 bytes
Vssn = 9 bytes

Finding Order p:
p(Pb)+(p-1)(Vssn)<B
p(6)+(p-1)(9)<8192
6p + 9p -9 < 8192
p<(8192+9)/(6+9)
p<546.7333
hence, p= 546

Finding p leaf:
pleaf(Vssn +Pr)+Pb<8192
pleaf (9 + 7) + 6 < 8192
pleaf<(8192-6)/(9+7)
pleaf<511.625
hence, pleaf = 511

(ii)the number of leaf-level blocks needed if blocks are approximately 69% full (rounded up for convenience):
Number of records in leaf nodes = 0.69*pleaf = 0.69 * 511 =352.59
= 353 (Rounded up for convenience)
Total Number of records: 10 000 000 000
Number leaf – level blocks needed = ceiling [10 000 000 000 /353] = 28328612 blocks

(iii) the number of levels needed if internal nodes are also 69% full (rounded up for convenience):
Average fan out, fo, for internal nodes = 0.69 * (p) = 0.69 * (546) = 376.74
= 377 (Rounded up for convenience)

Number of records in 2^{nd} Level = Number of blocks in leaf-level nodes = b1 = 28328612 blocks
Number of blocks in 2^{nd} Level, b2 = ceiling [28328612/353] = 80252

Number of records in 3rd Level = Number of blocks in 2nd Level = b2 = 80252
Number of blocks in 3rd Level, b3 = ceiling [80252/353] = 228

Number of records in 4th Level = Number of blocks in 3rd Level = b3 = 228
Number of blocks in 4th Level, b4 = ceiling [228/353] = 1
As number of blocks =1, This is the top Level

Number of Levels needed = 4

Alternatively, ceiling [Log377(28328612)] + 1 = 3 + 1 = 4

(iv) the total number of blocks required by the B+-tree:
b = b1 + b2 + b3 + b4 = 28328612 + 80252 + 228 +1 = 28409093 blocks

(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:
Will have to traverse till leaf nodes
Block accesses = x +1 = 4 +1 = 5 blocks
Where x is number of levels.

f. Repeat (part e), but for a B-tree rather than for a B+-tree. Compare your results for the B-tree and for the B+-tree.

ANSWER Q2(f):

(i) the orders p and p leaf of the B-tree:
Block Pointer size, Pb = 6 bytes
Record Pointer size, Pr = 7 bytes
Block size, B = 8192 bytes
Vssn = 9 bytes

Finding Order p:
p(Pb)+(p-1)(Vssn+Pr)<B
p(6)+(p-1)(9+7)<8192
6p + 16p -16 < 8192
p<(8192+16)/(6+16)
p<373.0909
hence, p=373

Finding p leaf:
There is no p leaf in B – tree.

<u>COMAPRISION: order of p is less for B-tree compared to B+-tree as internal nodes also store record pointers in B-tree</u>

(ii)the number of leaf-level blocks needed if blocks are approximately 69% full (rounded up for convenience):

Number of records in leaf nodes = 0.69 * (p) = 0.69*(373) = 257.37
= 258 (Rounded up for convenience)
Total Number of records: 10 000 000 000
Number leaf – level blocks needed = ceiling [10 000 000 000 /258] = 38759690 blocks

COMAPRISION: Number of leaf -level blocks needed is more for B-tree compared to B+-tree as bfr is less for B-tree

(iii) the number of levels needed if internal nodes are also 69% full (rounded up for convenience):
Average fan out, fo, for internal nodes = 258 (Rounded up for convenience)

Number of records in 2nd Level = Number of blocks in leaf-level nodes = b1 = 38759690 blocks
Number of blocks in 2nd Level, b2 = ceiling [38759690/258] = 150232

Number of records in 3rd Level = Number of blocks in 2nd Level = b2 = 150232
Number of blocks in 3rd Level, b3 = ceiling [150232/258] = 583

Number of records in 4th Level = Number of blocks in 3rd Level = b3 = 583
Number of blocks in 4th Level, b4 = ceiling [583/258] = 3

Number of records in 5th Level = Number of blocks in 4th Level = b4 = 3
Number of blocks in 5th Level, b5 = ceiling [3/258] = 1
As number of blocks =1, This is the top Level

Number of Levels needed = 5

Alternatively, ceiling [Log258(38759690)] + 1 = 4 + 1 = 5

COMAPRISION: Number of levels is more for B-tree compared to B+-tree as blocks are a lot more in B-tree

(iv) the total number of blocks required by the B-tree:
b = b1 + b2 + b3 + b4 = 38759690 + 150232 + 583 +3+1 = 38910509 blocks

COMAPRISION: Number of blocks is more for B-tree compared to B+-tree as bfr is less in B-tree

(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:
In worst case have to traverse all levels
Block accesses = x +1 = 5 +1 = 6 blocks
Where x is number of levels & + 1 is for base table access.
COMAPRISION: Number of block accesses is more for B-tree compared to B+-tree as more levels in B-tree

IN CONCLUSION: B+ tree is more efficient

Q3. Consider a DBMS that has the following characteristics: 1KB fixed-size blocks, 12-byte pointers, 56-byte block headers. We want to build an index on a search key that is 8 bytes long. Calculate the maximum number of records we can index with a

a. 3 Level B+- tree index (including the root level):
ANSWER Q3(a):

ASSUMING 1KB = 1024 BYTES & BLOCK/RECORD POINTER SIZE= 12 BYTES

Block size = 1024 bytes, Pointer size = 12 bytes, Block Header size = 56 bytes, Key size (Vk) = 8 bytes

$P*12 + (P-1)*(8) + 56 < 1024$
$P<48.8$
$P=48$

$Pleaf(8+12)+12+56<1024$
$Pleaf<47.8$
$Pleaf=47$

b1 = ceiling[R/47]
b2= ceiling[b2/48]
b3= ceiling [b2/48] = 1
at max b2 can be 48

so,
48 = ceiling[b2/48]
At max b2 can be 2304

So,
2304 = ceiling[R/47]
At max R can be 108288

Hence, Maximum number of records that can be indexed is = 108288

b. 3 Level B-tree index (including the root level):
ANSWER Q3(b):

ASSUMING 1KB = 1024 BYTES & BLOCK/RECORD POINTER SIZE= 12 BYTES

Block size = 1024 bytes, Pointer size = 12 bytes. Block Header size = 56 bytes, Key size, Vk = 8 bytes

$56+(P-1)*(8+12)+P(12) < 1024$
$P<30.875$
$P=30$

b1 = ceiling[R/30]
b2= ceiling[b2/30]
b3= ceiling [b2/30] = 1
at max b2 can be 30

so,
30 = ceiling[b2/30]

At max b2 can be 900

So,
900= ceiling[R/30]
At max R can be 27000

Hence, Maximum number of records that can be indexed is = 27000

Q4. Assume a relation *R (A, B, C)* is given; Suppose *A, B, C* are integer type values. Relation *R* is stored as an un-ordered file (un-spanned) on key field *A* and contains 5000 data blocks. Assume there is B+- tree access structure (index) on *A* of height x=4 (root, 2 intermediate layer, leaf). Moreover, one node of the B+-tree is stored in one block on the disk. Estimate the number of block fetches needed to compute the following queries:

a. SELECT * FROM R WHERE A = 777:
x+1 = 4+1 = 5
b. SELECT C FROM R WHERE A = 111 AND B = 3:
(x+1) = 4+1 = 5
c. SELECT * FROM R WHERE A = 111 OR A = 3;
2(x+1) = 2(4+1) = 10
d. SELECT * FROM R WHERE A > 100:
x+1 = 4+1 = 5
e. SELECT COUNT(*) FROM R WHERE A > 100:
x = 4