Quiz 2 Solutions

1) Let l be the fixed length of the row of a relation R with N rows, and B be the block size. Let l be smaller than B, and let unspanned placement of rows in blocks be used. There is no header or footer in the block. Then, Select one or more:

a. Number of blocks needed to store R is exactly ceil[(N/floor[B/l])]

b. Number of rows per block is exactly floor[B/l]

c. Number of rows per block is exactly ceil[B/l]

d. None of the others

e. Number of blocks needed to store R is exactly ceil[(N/ceil[B/l])]

CORRECT ANSWERS – A,B

2) Consider R(K, A1, A2, …, An) with a composite clustered index on (A1, A2), R has N rows, and A1 has p distinct values and A2 has q distinct values for a populated relation R. Select one or more:

a. None of the others

b. R is ordered on K

c. R is ordered in (A1, A2) (first A1 and then A2)

d. Composite Clustered index on (A1, A2) has (p+q) rows

e. A row of clustered index is (A1-value, A2-value, BlockPointer)

CORRECT ANSWERS – C,E

EXPLANATION

- B is wrong because as it is a clustered index on (A1,A2) it has to be ordered on (A1,A2) and it cannot be ordered on K.

- C is correct because it is a clustered index on (A1,A2) so it must be ordered on (A1,A2)

- D is wrong because, A1 as p distinct values and A2 has q distinct values, so total number of possible combinations of (A1,A2) is p*q.

- E is correct by definition

3) Consider R(K, A1, A2, …, An) with N rows, and a secondary index on K Select one or more:

a. None of the others

b. A row of secondary index on K is (K-value, BlockPointer)

c. There are N rows for the secondary index on K

d. R is ordered on K

e. R is ordered on any attribute other than K

CORRECT ANSWER – A

EXPLANATION

- B is wrong because it should be recordPointer not blockpointer

- If K was mentioned as Key then C would have been correct, as there would have been N distinct values and thus N rows. But it is not mentioned.

- D is wrong because, It is a secondary index on K so it must not be ordered on K

- E is wrong because, It is not necessary that R has to be ordered. It may not be ordered on any attribute.

4) Consider a relation R(K, A) with N rows stored in nR blocks and attribute A takes only two values (x,y), there are (N-1) rows with value A=x, and 1 row with value A=y. To retrieve all rows for the query Select * From R where A=y. To get this result - Select one or more:

a. If R is hashed on A, it will take exactly one block access.

b. If R has clustered index on A it will take exactly two block accesses

c. If R is ordered on K it can take [floor([log2(nR)]/2)] block accesses

d. None of the others

e. If R is unordered it can take [floor(nR/2)] block accesses

CORRECT ANSWERS – A,B,E

EXPLANATION

- A is correct by definition

- For clustered index it will take:

BA = log2(number of distinct keys) + No. of occurrences

BA = log2(2) + 1

BA = 1 + 1 = 2

- Option C is Wrong If R is ordered on K then binary search cannot be done for A(So logarithm cannot come into the picture)

- Option E is correct as R is unordered, the best case would be 1 block access, worst case would be nR block accesses. On average floor(nR/2) block accesses would be required.

5) If consider a relation with >10 rows, the block size B is doubled, then Note - (approximately halve is number in range [(number/2) -3], to [(number/2)+3]), not necessarily exactly half - but very close to it. Select one or more:

a. The number of blocks for secondary index on Key will remain the same

b. None of the others

c. The number of blocks for primary index will remain the same.

d. The number of leaf nodes of B-tree will approximately halve

e. The number of blocks for clustered index on non-key attribute A will approximately halve

CORRECT ANSWERS – D,E

EXPLANATION

- Option A is wrong because as block size increases, more keys could be held in a block, so basically number of blocks for SI will half.
- Option C is wrong using the same explanation for Option A.
- Option D is correct because as block size doubles, fanout also doubles, so minimum <key,ptr> pairs in each block will double. So finally the overall number of blocks required in the B-Tree will also half. Therefore no. of Leaf Nodes will be halved.
- Option E is true because block size increases, number of rows per block increases, double the number of keys could be stored in one block. Therefore number of blocks will half.