Homework #6 Answers
due Thursday, December 2 , 2010 at 2 pm

Database Systems, CSCI-4380-01

Each student must work on this homework alone.

1 Homework Description

Question 1. Estimate the cost of following query plans. For each plan, the implementation of each operation, the amount of buffer allocated to each operation and the size of the input relations are given. Assume $R(A, B, C, D, E)$, $S(A, F, G, H)$, $B(R) = 5,000$ and $B(S) = 10,000$, $Tuples(R) = 100,000$ and $Tuples(S) = 10,000,000$.

(a) $R \bowtie S$ using block-nested loop join for $\bowtie$ using $M = 101$.

Answer. Read R once into 100 blocks, and S 5000/100= 50 times using 1 block. Total cost: $5,000 + 50 * 10,000 = 505,000$

(b) $(R \bowtie S) \bowtie T$ using block-nested loop join between R and S using $M = 101$, and the join with T using block-nested loop join also using $M = 101$. Assume $B(T) = 5,000$ and $B(R \bowtie S) = 30,000$.

Answer. We already computed $R \bowtie S$ above. Given $B(R \bowtie S) = 30,000$ which we will read into 100 blocks as it is the outer relation for the second join (with T), we need to read T a total of $30,000/100 = 30$ times. Note that, the cost of reading the join $R \bowtie S$ is already included in the cost of computing this join. When the join is computed, the results are produced in memory and pipelined to the next operation. As a result, the second join has the additional cost of $30 * 5000 = 150,000$.

Total cost $= 505,000 + 150,000 = 655,000$.

(c) $\delta(R)$ using sort based duplicate removal where $M = 200$ buffer pages are allocated for sorting and $M = 1$ buffer page is allocated for duplicate removal. (Note that you can do duplicate removal combined with sorting as well, this question is simplifying the logic for you).

Answer. Let's sort R first using 200 blocks.

Step 1 will read R once and write R once, and create $5,000/200 = 25$ sorted files/groups. Total cost of this step is $10,000$ pages.

At the next stage, we need to read one page from each sorted file/group and merge. Given we only have 25 groups and 200 blocks, we can complete sorting in this step and pipe the results to the duplicate removal operation, by reading R once, merging and

piping the results to the input buffer of the next operation. Total cost of this step is $5,000$.

Total cost of this query is $15,000$ as duplicate removal is done in memory and does not add an additional cost.

(d) $R \cup S$ using sort based union where $M = 200$.

Answer. Ok, obviously R and S cannot be joined given that they are not compatible. But, let's assume that they are for the sake of this question.

We will then sort R and S first and then do the union.

Let's first use the simple formula for doing so:

To sort R, we complete step 1 and step 2 once, total cost of $20,000$ (as oppposed to the previous question, we will write the sorted R into disk before the next operation).

To sort S, first step of sorting will produce $10,000/200 = 50$ sorted files, so we can completely sort S in the next step (we need to read one block from each file, which means we need 50 blocks and have 200). Total cost is $4 * 10,000 = 40,000$ disk pages.

Now, to do the union, we need to read one page from each of R and S (and we have 200 pages total, so this is not hard), and do the union. To complete the union, we need to read R and S completely once. Total cost of this step is $15,000$, and the total cost of this operation is $20,000 + 40,000 + 15,000 = 75,000$.

Ok, let's get fancy. The last sort merge step of sorting R needs to merge 25 sorted groups and S requires to merge 50 sorted groups. We can actually combine the sort merge phase with union easily. To do both sort/merge of R and S and union together requires us to read one page from each sorted group of R and S, i.e. 75 blocks, which is easily available (we have 200). So, instead of sorting completely and writing the results to disk, we can just compute query in the second stage. To accomplish this we need to do step 1 of sorting for both relation (1 read and 1 write), and read both at the same time for the remainder of the query (1 read). Total cost in this case is $3 * (5,000 + 10,000) = 45,000$.

Both answers are acceptable.

(e) $\sigma_{A=5 \text{ and } B>10 \text{ and } B<100} R$ using sequential scan of R.

Answer. Cost: reading R once, $5,000$ pages.

(f) $\sigma_{A=5 \text{ and } B>10 \text{ and } B<100} R$ using index I1 on $R(B, A)$ of height 2 where each node (except root) contains about 200 entries and there are $2,000$ tuples for $A = 5$ and $50,000$ tuples for $B > 10$ and $B < 100$ and 500 tuples for the whole selection condition.

Answer. We can scan the index for the B condition and then check the A condition, which requires $50,000/200 = 250$ leaf nodes and two internal nodes. 252 nodes total.

Then, we need to read each matching tuple from disk to find other attributes of R which requires an additional 500 disk reads.

Total cost is 752 disk pages.

Question 2. Estimate the size of the following relations in terms of number of tuples and number of disk pages (or memory blocks). Assume each attribute takes 40 bytes and each disk block is 4K bytes (assume 4,000 bytes for simplicity). For example, given a relation

with 5 attributes, each tuple is about 200 bytes and a disk page takes about 4000/200=20 tuples.

Suppose $R(A, B, C, D)$ has 10,000 tuples, and $VALUES(R.A) = 10,000$, $VALUES(R.B) = 2,000$, $VALUES(R.C) = 5,000$ and $VALUES(R.D) = 200$. Suppose $S(E, F, G)$ has 1,000,000 tuples, $VALUES(S.E) = 10,000$, $VALUES(S.F) = 1,000$ and $VALUES(S, G) = 5,000$.

(a) $C_1 : R.A = 5$

Answer. $Selectivity(C_1) = 1/10,000$.

$Tuples(C_1) = 1$, $B(C_1) = 1$.

(b) $C_2 : R.B = 10$ and $20 \leq R.A$ and $R.A \leq 100$

Answer. $4000/160 = 25$ tuples of R per block.

$Selectivity(C_2) = 1/200 * 1/3 = 1/600$

$Tuples(C_2) = 10,000/600 = 16$, $B(C_2) = 1$.

(c) $C_3 := 20 \leq R.A$ and $R.C = 100$ and $50 \leq R.D$

Answer. $Selectivity(C_3) = 1/3 * 1/5000 * 1/3 = 1/45,000$

$Tuples(C_3) = 10,000/45,000 = 1$ and $B(C_3) = 1$.

(0 or 1 tuples)

(d) $C_4 : S.F = 10$

Answer.

S has $4,000/120 = 33$ tuples per block.

$Selectivity(C_4) = 1/1,000$

$Tuples(C_4) = 1,000,000/1,000 = 1,000$, and $B(C_4) = 1,000/33 = 30$ blocks.

(e) $C_5 : S.E = 1$ or $S.G = 2$

Answer.

$Selectivity(S.E = 1) = 1/10,000$

$Selectivity(S.G = 2) = 1/5,000$

$Selectivity(C_5) = 1 - ((1 - 1/10,000) * (1 - 1/5,000)) = 0.0003$

$Tuples(C_5) = 1,000,000 * 0.0003 = 300$

$B(C_5) = 300/33 = 9$ blocks.

(f) $\pi_{B,F}(R \bowtie_{A=E} S)$

Answer. Join size = Size of cartesian product/ max{selectivity(R.A=c), selectivity(S.E=c)}

Join size $= 10,000 * 1,000,000/10,000 = 1,000,000$.

After projection, we have two attributes. So, we can fit $4,000/80 = 50$ tuples in a block. Hence, the result fits in $1,000,000/50 = 20,000$ blocks.

Question 3. For each query above in Question 2, discuss the best index that would impact the query. Write one sentence to explain why this is a good index to create.

Answer. (a) Index on R.A, (b) Index on $R.B, R.A$ is best, (c) Index on $R.C, R.A, R, D$ would be good (we could only scan for R.C and R.A conditions, but can find all the tuples satisfying the condition. However, $R.C$ is already very selective, so we can also just create an index on $R.C$ to be able to quickly find tuples for $R.C$ (note that an index on just $R.C$ would be smaller and faster to search). (d) Index on $S.F$, (e) No good single index really, the best we can do is two indices $S.E$ and $S.G$ both. But, if only one index can be generated, then we can use $S.E, S.G$ where we have to scan the whole index. As the conditions are both selective, this might still be useful. (d) The best B-tree index happens to be for $S.E$ as for each tuple of R, we can find all the matching tuples. However, this index is unlikely to be used as there 10,000 distinct values each for both $R.A$ and $S.E$ so every tuple in S will join with a tuple in R. A much better choice would be to sort/cluster S on disk by $S.E$. Then, the sort-merge join is possible by simply sorting R and would most likely to be used.

Also, assuming that each query is executed with the same frequency, what is the most important single index to create?

Answer. R takes about 400 pages and S about 30,000 pages. So, a query on R even with a most selective index reduces cost from 400 pages to 2-3 pages (height of the tree plus 1 page look up).

Queries on S are more crucial to optimize, but query in part (d) is not that selective (1,000 random access pages may be quite costly), similarly paart (e) is slightly more selective, 300 random access pages but only after scanning two indices. If we had a single index on $S.E, S.G$, the cost of finding the matching tuples is also going to be considerable. So, even if two indices was allowed, we get a speed up of an order of 100.

(f) This depends on the memory available. Now, suppose we have $M = 1,000$ i.e. 4MB available for this query which is reasonable for a high load server. Then, a block-nested loop join would cost: 310,000 blocks. If S was sorted, then we can sort R and do a sort-merge join in $60,000$ blocks, a speed up of a factor of 5 on the costliest query.

If $M = 10,001$, then there is no speed up.

Given this, indices on R provide a speed up of an order of magnitude 100. So, that would be the most preferable. especially $R.A$ which is the most costly one. Further, an index on $R.A, R.B$ or $R.A, R.C$ can be used for two different queries even though it is not optimal for all, can still be useful.

(Note that there are other considerations, but we do not have a good example for these in the given queries. For example, if queries also required sorting by $S.E$, then definitely the sorting of S on $S.E$ will be useful there also. Another consideration is how much the user is willing to wait. For example, (f) is a very costly query. If (f) takes just too long for a user's patience, then optimizing that would be a high priority. The queries on R probably are quite fast anyways given R is small. So far, we have been trying to find biggest reduction factor but the aim could be to bring the processing times to a certain acceptable limit for all queries which would change the results.)