CSI 410. Database Systems – Spring 2021

Homework Assignment III

The deadline for this assignment is **11:59 PM, April 10, 2021**. *Submissions after this deadline will not be accepted.* Each student is required to enter the UAlbany Blackboard system and then upload a .pdf file (in the form of [first name]_[last name].pdf) containing answers to Problems 1-6.

The total grade for this assignment is 100 points. If you find any error or have questions or suggestions, please contact the instructor (`jhh@cs.albany.edu`).

Problem 1. (20 points) Consider a situation where *hash join* is applied to relations R and S. Assume that (1) relation R contains 10^9 disk blocks, (2) relation S contains 10^8 disk blocks, (3) the size of each disk block is $1KB$, (4) the hash join partitions S into 15 buckets in a manner where each bucket fits into the main memory, and (5) $16GB$ of main memory is used for buffering (one buffer is used for each file being read or written and all of the buffers have the same size). Answer each of the following questions:

(a) (10 points) Calculate the *number of block transfers* and the *number of disk seeks* during the *partitioning phase* of the hash join. Justify your answer. To simplify cost analysis, assume that (i) the last block of each bucket from R and S is completely filled and (ii) when the end of each bucket is saved, the buffer for the bucket is completely filled.

(b) (10 points) Calculate the *number of block transfers* and the *number of disk seeks* during the *build and probe phases* of the hash join. Justify your answer.

Problem 2. (20 points) Consider a situation where *external sort-merge* is applied to relation R in Problem 1. Note that the size of the main memory is $16GB$ and thus the main memory can keep up to 16×10^6 disk blocks. Answer each of the following questions:

(a) (10 points) Explain *how many initial runs* will be created. Also, describe how *many block transfers* and *disk seeks* will be needed to create all of these initial runs.

(b) (10 points) After the initial runs are created as explained above, *external sort-merge* repeatedly merges a number of runs into a bigger run until only one final run remains. Assume that each merge pass uses 8 input buffers and 1 output buffer. Explain *how many merge passes* are needed to obtain one final run. Also, calculate *the total number of block transfers* during these merge passes (assume that the final run is not actually written to disk and consumed in memory, for example, by a selection or a join operator). Justify your answer.

Problem 3. (20 points) Consider a situation where *merge join* is applied to relations R and S described in Problem 1. Answer each of the following questions:

(a) (10 points) Assume relations R and S are already sorted by the join column(s). Explain how many *block transfers* and *disk seeks* this join requires when it uses, for R and S, two buffers of the same size in the $16GB$ of the main memory.

(b) (10 points) Explain whether it is more advantageous to use *hash join* or *merge join* when only S is already sorted by the join column(s) and thus R needs to be sorted (only for merge join, not hash join) as in Problem 2. Justify your answer by considering *the number of block transfers* (and ignoring the number of disk seeks) for each of these join scenarios.

Problem 4. (10 points) Consider relations $R(A, B, C)$, $S(C, D, E)$ and $T(E, F, G)$. Find a relational expression which (1) is equivalent to $\Pi_{A,G}(R \bowtie S \bowtie T)$ and (2) applies projection to each of R, S and T on the smallest number of attributes.

Problem 5. (20 points) Determine whether or not each of the following statements is true for any arbitrary relations $R(A, B)$ and $S(A, B)$. If a statement is true, prove it. Otherwise, give a relevant example.

(a) (10 points) $\Pi_A(R \cap S) = \Pi_A(R) \cap \Pi_A(S)$

(b) (10 points) $\sigma_\theta(R \cup S) = \sigma_\theta(R) \cup \sigma_\theta(S)$

Problem 6. (10 points) Consider four relations $R(\underline{A}, Z)$, $S(\underline{B}, A)$, $T(\underline{C}, B)$ and $U(\underline{D}, C)$ (the primary keys are underlined). Assume that (1) A in S is a foreign key to R, (2) B in T is a foreign key to S, (3) C in U is a foreign key to T, (4) these relations do not contain any `null` value, and (5) the size (i.e., the number of rows) of each relation is as follows:

$$r_R = 4000,\ r_S = 3000,\ r_T = 2000,\ r_U = 1000.$$

Also, assume that all attributes of the relations are of the same length and we use hash join, so the cost of joining $X \in \{R, S, T, U\}$ and $Y \in \{R, S, T, U\}$ can be expressed as:

$$k(r_X \cdot c_X + r_Y \cdot c_Y)$$

where k is a constant, r_X and c_X denote the number of rows and the number of columns of X, respectively, and r_Y and c_Y denote the number of rows and the number of columns of Y (we ignore the cost of producing the output relation).

Under the above assumptions, find the lowest cost plan for computing $R \bowtie S \bowtie T \bowtie U$ using *dynamic programming* and *left-deep* join trees. You need to complete the following table while finding the best plans (e.g., in the form of $((\square \bowtie \square) \bowtie \square) \bowtie \square$ in the last line) and associated costs.

Subquery	Size	Cost	BestPlan
$R \bowtie S$	3000	$14000k \quad [= (4000 \cdot 2 + 3000 \cdot 2)k]$	$R \bowtie S$
$R \bowtie T$			
$R \bowtie U$			
$S \bowtie T$			
$S \bowtie U$			
$T \bowtie U$			
$R \bowtie S \bowtie T$			
$R \bowtie S \bowtie U$			
$R \bowtie T \bowtie U$			
$S \bowtie T \bowtie U$			
$R \bowtie S \bowtie T \bowtie U$			

After solving the above problems, please state the amount of time spent for this assignment. Feel free to add comments or suggestions if any.