**Homework 9 for CS542 - Fall 2023**

**Assigned:** Wednesday, November 1st, 2023  
**Due:** Wednesday, November 8th 2023 at 10:00am ET  **NO LATE SUBMISSIONS ACCEPTED!  
Submission:**  CANVAS.  
**Maximum:** 100pts   
**Note:** This homework is to be completed by each student individually. No help besides the textbook should be needed. ***Copying any answers or part of answers from other sources (including the internet and your colleagues) will earn you a grade of zero.***

**Problem 1 (Evaluation of Relational Operators) [56 Points]**

Given the following relational operators and some properties about their input relations:

1. Duplicate elimination operator over unsorted relation R
2. Grouping operator (group by column X) over a sorted relation R on column X
3. Grouping operator (group by column X) over unsorted relation R
4. Sorting operator (sort by column X) over unsorted relation R
5. Sorting operator (sort by column X), and assume the operator can use a B-tree index that exists on R.X to read the tuples.
6. Join of two relations R and S
7. Bag Union of relations R and S
8. **[2 Points each Item]** For each of the items above, report whether the operator is “Blocking” or “Non-Blocking” and describe why.
9. **[6 Points each Item]** Assume relation R is 1,000 blocks and relation S is 150 blocks, and the available memory buffers are 200. Moreover, for Point (e), the R.X index size is 70 blocks. For each of the items above, discuss:
   1. Whether the operator can be done in one pass or not.
   2. If it can be done in one pass, what are the size constrains?
   3. If it cannot be done in one pass, then how many passes are needed? Describe the algorithm that uses that number of passes you suggest? What will be the I/O cost?

**Problem 2 (Query Processing Strategies & Their Costs) [44 Points (11 each)]**

****

**Consider the condition join R1⋈R1.a=R2.b R2, given the following information**

**about the relations to be joined. The cost-metric is the number of IOs. The cost of writing**

**the result would be the same independent of the particular join method used, hence we henceforth**

**can ignore it.  
Given**

* R1 has 10,000 tuple, 10 tuples per block
* R2 has 2,000 tuple, 10 tuples per block
* The available memory buffers are 52

1. Assume we use a block-oriented nested loop join.
   1. Which relation you suggest to be the outer relation?
   2. What is the cost of the join if we use the outer relation as you suggested?
   3. What is the cost of the join if we use the other relation (not what you suggestion) as the outer one?
2. Assume we use a sort-merge join, and we use the “Efficient Sort-Merge” algorithm covered in class where we merge the sorting and joining together
   1. What is the cost of the join algorithm?
   2. What is the minimum number of buffers needed for the cost to remain unchanged, i.e., Can we use less than 52 buffers and still have the same cost that you calculated in 2.a?
3. Assume we use a hash-join, and we will do a simple hash-join.
   1. What is the cost of the join algorithm?
   2. What is the minimum number of buffers needed for the cost of the hash join to remain unchanged, i.e., Can we use less than 52 buffers and still have the same cost that you calculated in 3.a?
4. Assume we use an index-join with R2 as the outer relation, and we have an index on R1.a. Assume that the index fits in memory. Moreover, on average we get 5 R1 tuples matching every R2 tuple.
   1. What is the cost of the join algorithm?

Here you might need to make assumptions i.e. one or both tables are clustered (or not), or index is clustered or not, index in memory or not, etc.