Final Exam

DSE201, Winter 2016

Name:

Brief Directions:

- Write clearly!
- Good luck!

1 Algebra and Estimation (Homework, revisited)

In the homework problem, you were asked to consider three relations R(A;B;C), S(A;D), W(B;E) and the query

```
SELECT *
FROM R, S, W
WHERE R.A=S.A AND R.B=W.B AND R.C=1
```

Then you considered an optimizer that produces all join expressions, where the selection $\sigma_{R.C=1}$ is pushed down and applied directly on the table R. Plans cannot have cartesian products or trivial natural joins that are equivalent to cartesian products.

After excluding expressions that can be derived from each other just by using the commutativity of the natural join, we came up with the solutions:

1. $(\sigma_{R.C=1}R \bowtie S) \bowtie$ W
2. $(\sigma_{R.C=1}R \bowtie W) \bowtie$ S

Then you were asked to select the best logical query plan by estimating the size of each intermediate result (i.e., of each subexpression) and selecting the plan that has the smallest sum of intermediate result sizes. The statistics and key information were:

- B is a key of R and W.B is a non-null foreign key that references R.B
- A is a key of S and R.A is a non-null foreign key that references S.A
- T(R) = 10^6
- T(S) = 10^5
- T(W) = 10^3
- V (R.C) = 10^2

The second plan ended up being the best, with total intermediate results being $10^4 + 10$ tuples.

New Problem Consider an alternate optimizer that produces all possible algebra expressions, including ones where the selection $\sigma_{R.C=1}$ is not applied directly on R. This alternate optimizer also chooses as "optimal" the expression with the smallest sum of intermediate result sizes.

Is the optimal expression still the one we had found above (the second plan)? Justify your answer.

2 Algebra and Estimation

Produce an optimal algebraic expression for the following query over tables `R(A, B)` and `S(A, C, D)`, where "optimal" means that it has the smallest total size of intermediate results, among all possible algebraic expressions that are equivalent to this query. Write the sizes of all intermediate results.

```
SELECT A, C, AGG(B) AS N
FROM R, S
WHERE S.A = R.A AND S.D = 1
GROUP BY A, C
```

given the following statistics

$$
\begin{aligned}
&T(R) = 10^9\\
&V(R, A) = 10^6\\
&V(R, B) = 10^9\\
&T(S) = 10^{10}\\
&V(S, A) = 10^7\\
&V(S, C) = 10^2\\
&V(S, D) = 10
\end{aligned}
$$

Assume (the common assumption) that

$$V(R, A) < V(S, A) \Rightarrow \pi_A R \subset \pi_A S$$

3 Semijoin

Given two relations R and S the semijoin $R \ltimes_c S$ has the following output: The attributes of $R \ltimes_c S$ are the same with the attributes of R. If a tuple r appears n times in R and there is at least one tuple s in S, such that r and s satisfy the condition c, then the output includes n copies of r. No other tuples appear in the output.

For brevity, let's also define a natural semijoin $R \ltimes S$ that has the following output: The attributes of $R \ltimes S$ are the same with the attributes of R. If a tuple r appears n times in R and there is at least one tuple s in S, such that r and s have the same values on the common attributes, then the output includes n copies of r. No other tuples appear in the output.

For example, assuming that R is

A	B
1	2
1	2
3	4

and S is

B	C
2	3
2	3
2	3

the semijoin $R \ltimes_{R.B=S.B} S = R \ltimes S$ is

A	B
1	2
1	2

Use in Queries Consider this query. Write an equivalent algebra that uses semijoin.

```
SELECT *
FROM classes c
WHERE c.ID IN (SELECT e.class FROM enrollment e)
```

Algebraic Definition Write an algebraic expression that computes the semijoin $R \ltimes_c S$ using other operators. For example, if the list of attributes of R is $\bar{A}$ then this is one possible "solution"

$$R \rhd\!\!<_c S = \pi_{\bar{A}}(R \bowtie_c S)$$

Except that it is wrong. Provide a correct one.

Equivalences Now, assume relations R, S and T. The notation $c(A)$ refers to a condition that refers to the list of attributes A only. Declare true or false each of the following. If the answer is "no", also provide counterexample.

1. $\sigma_{c(A)}(R \rhd\!\!< S) = (\sigma_{c(A)}R) \rhd\!\!< S$, where R has a list of attributes A and S has no attributes of A.
2. $\delta(R \rhd\!\!< S) = (\delta R) \rhd\!\!< S$
3. $(R \rhd< S) \rhd< T = (R \rhd< T) \rhd< S$, where all of R, S and T have a single common attribute A and no pair of R, S and T has a common attribute other than A.

Recall, δ is the duplicate elimination operator.

4 Column Databases

Consider the table `R(D1,D2,D3,M)`. You already have a PostGres database and you wonder whether it is worthy to buy a column database in order to accomodate queries on `R`. Of course, the answer depends on knowing the queries that will be issued. For each of the following queries, declare whether a column database will be significantly better or whether PostGres is good enough(or even better). Just place a circle around the relevant system. "Significantly" means at least close to a multiple, say 2x. A 10% improvement is not significant.

When you consider column databases, assume they do not have indices. (This is not exactly accurate, as column databases also have indices, but we adopt it for the sake of the exercise.) Furthermore, assume that the table is in the order of terabytes and resides in hard disk. The main memory is only 32GB. Each one of the `D` attributes is an integer and `M` is a float. Assume that $V(\texttt{D1},\texttt{R}) = V(\texttt{D2},\texttt{R}) = V(\texttt{D3},\texttt{R}) = 10^6$. Assume $V(\texttt{[D1,D2,D3]},\texttt{R}) = 10^8$.

1. `SELECT D1, SUM(M) FROM R GROUP BY D1`: Column - Postgres

2. `SELECT D1, D2, D3, SUM(M) FROM R`: Column - Postgres

3. `SELECT D1, D2 SUM(M) FROM R WHERE D3=?`: Column - Postgres

As usual, "?" means that a constant will be given at query time.