CS411: Database Systems **Spring 2018**

Assignment 5

Due Tuesday, May 1 at 11:59pm

General Instructions

- Feel free to talk to other members of the class in doing the homework. You should, however, write down your solutions yourself. *List the names of everyone you worked with at the top of your submission.*
- Keep your solutions brief and clear.
- Please use Piazza if you have questions about the homework but do not post answers. Feel free to use private posts or come to the office hours.

Homework Submission

- We DO NOT accept late homework submissions.
- We will be using Compass for collecting the homework assignments. Please submit your answers via Compass. Hard copies are not accepted.
- Contact the TAs if you are having technical difficulties in submitting the assignment; attempt to submit well in advance of the due date/time.
- The homework must be submitted in **pdf** format. Scanned handwritten and/or hand-drawn pictures in your documents won't be accepted.
- Please do not zip the answer document (PDF) so that the graders can read it directly on Compass. You need to submit one answer document, named as **hw5_netid.pdf**.
- Please see the assignments page for more details. In particular, we will be announcing clarifications, if any, on this page.

1 Query Execution (25 pts)

Consider the following relations with no indexes on them:

- Relation R has 5,000 tuples, 100 tuples per block
- Relation S has 2,000 tuples, unknown number of tuples per block

The number of blocks in memory is 10. Say, S is as large as possible (within the limits that main memory can afford) for the two pass sort-merge join (slide 57). That is, if S was larger, the two-pass sort-merge join would not work.
Answer the following questions:

A How many tuples per block does S have? (Do not forget to show your calculations.)

{**A**nswer: R has 5,000/100 = 20 blocks (i.e. B(R) = 50)
B(R) + B(S) = M(M-1), where M = 10 and B(R) = 50
Hence, B(S) = 40, and S has 2000 tuples / 40 blocks = 50 tuples per block.}

B Using your answer from A, what is the cost of joining R and S using the two pass sort-merge join algorithm (slide 57)? {**A**nswer: 3 (B(R) + B(S)) = 3 * 90 = 270}

C Using your answer from A, what is the cost of joining R and S using the optimized block nested-loop join algorithm?

{**A**nswer: Since B(S) <B(R),
Cost = B(S) + B(R) × B(S)/(M-2) = 40 + 40*50/8 = 290 }

D What is the cost of joining R and S using a two pass hash-based join?

{**A**nswer: 3*(B(R) + B(S)) = 3*90 = 270}

E Based on questions 2, 3, and 4, explain which variant of the algorithm you would choose in terms of I/O cost. If multiple algorithms have the same I/O cost, explain other considerations that may influence your choice.

{**A**nswer: Merge join and hash join are equivalently efficient in terms of I/O cost. However, sort-merge join will provide ordered output. Other reasonable explanation is also fine.}

2 Query Optimization (35 pts)

Consider the relations A(x,y,z), B(w,x), and C(u,v,w), with the following properties:

A(x,y)	B(y,z)	C(z,x,u)
T(A) = 2500	T(B) = 1000	T(C) = 6000
V(A, x) = 30	V(B, y) = 250	V(C, z) = 20
V(A, y) = 500	V(B, z) = 100	V(C, x) = 60
		V(C, u) = 40

where, T(R) = number of tuples in relation R and V(R, a) = number of distinct values of attribute a in relation R. Estimate the sizes (measured in number of tuples) of the result of the following expressions:

1. A $\times$ C

   {**A**nswer: T(A) x T(C) = 2500 $\times$ 6000 = 15,000,000}

2. A $\bowtie$ B

   {**A**nswer: T(A) x T(B) = 2500 $\times$ 1000 / max(V(A,y), V(B,y)) = 2500000/500 = 5000}

3. `SELECT u FROM C WHERE u=20`

   {**A**nswer: T(C)/V(C,u) = 6000/40 = 150 }

4. $\sigma_{\text{x = 10 and y=30}}$ (B $\bowtie$ C)

   {**A**nswer: (T(B $\bowtie$ C)) / V(B $\bowtie$ C, x) V (B $\bowtie$ C, y) = 6000 $\times$ 1000/(100 $\times$ 250 $\times$ 60) = 4}

3 Dynamic Programming (40 pts)

Consider the following relations, where T(R) = number of tuples in relation R and V(R, a) = number of distinct values of attribute a in relation R.

A(x,y)	B(y,z)	C(z,x,u)	D(u,v)
T(A) = 2500	T(B) = 1000	T(C) = 6000	T(D) = 2000
V(A, x) = 30	V(B, y) = 250	V(C, z) = 20	V(D, u) = 100
V(A, y) = 200	V(B, z) = 100	V(C, x) = 60	V(D, v) = 40
		V(C, u) = 40	

We want to join all these relations as efficiently as possible. Determine the most efficient way to do the join. Clearly state any assumptions you have made. Show your work by completing the following table (each step in the dynamic programming algorithm should be one row):

Subset	Size	Lowest Cost	Lowest cost plan
...	...	...	...

Subset	Size	Lowest Cost	Lowest cost plan
AB	10,000		BA
BC	60,000		BC
AC	250,000		AC
CD	120,000		DC
ABC	10,000	10,000	C(BA)
BCD	1,200,000	60,000	D(BC)
ACD	5,000,000	120,000	A(DC)
ABCD	200,000	20,000	D(C(BA))