The assignment is to be turned in before Midnight (by 11:59pm) on February 20. You should turn in the solutions to this assignment as a PDF file through Canvas. The solutions should be produced using editing software programs, such as LaTeX or Word, otherwise they will not be graded. The assignment should be done in groups of two students. Each group must submit only one file that contains the full name, OSU email, and ONID of every member of the group.

1: Query processing (3 points)

Consider the natural join of the relation *R(A,B)* and *S(A,C)* on attribute A. Neither relations have any indexes built on them. Assume that R and S have 80,000 and 20,000 blocks, respectively. The cost of a join is the number of its block I/Os accesses. If the algorithms need to sort the relations, they must use two-pass multi-way merge sort.

(a) Assume that there are 10 blocks available in the main memory. What is the fastest join algorithm for computing the join of R and S? What is the cost of this algorithm?

(b) Assume that there are 350 blocks available in the main memory. What is the fastest join algorithm to compute the join of R and S? What is the cost of this algorithm?

(c) Assume that there are 110,000 blocks available in the main memory. We like to have the output sorted based on the join attribute. What is the fastest join algorithm to compute the join of R and S? What is the cost of this algorithm?

2: Query processing (4 points)

Consider the following relations:

```
Dept (did (integer), dname (string), budget (double), managerid (integer))
Emp (eid (integer), ename (string), age (integer), salary (double))
```

Fields of types *integer*, *double*, and *string* occupy 4, 8, and 40 bytes, respectively. Each block can fit at most one tuple of an input relation. There are at most 22 blocks available to the join algorithm in the main memory. Implement the optimized sort-merge join algorithm for $Dept \bowtie_{Dept.managerid=Emp.eid} Emp$ in C++.

- Each input relation is stored in a separate CSV file, i.e., each tuple is in a separate line and fields of each record are separated by commas.
- The result of the join must be stored in a new CSV file. The files that store relations Dept and Emp are Dept.csv and Emp.csv, respectively.
- Your program must assume that the input files are in the current working directory, i.e., the one from which your program is running.
- The program must store the result in a new CSV file with the name join.csv in the current working directory.
- Your program must run on Linux. Each student has an account on *hadoop-master.eecs.oregonstate.edu* server, which is a Linux machine. You may use this

machine to test your program if you do not have access to any other Linux machine. You can use the following *bash* command to connect to *voltdb1*:

```
> ssh your_onid_username@hadoop-master.eecs.oregonstate.edu
```

Then it asks for your ONID password and probably one another question. You can only access this server on campus.

- You can use following commands to compile and run C++ code:

```
> g++ main.cpp -o main.out
> main.out
```