You should turn in the solutions to the written part of this assignment as a PDF file through Canvas before Midnight (by 11:59pm) on February 3rd. The solutions should be produced using editing software programs, such as LaTeX or Word, otherwise they will not be graded. You should turn in the source code to the programming question (question 4) separately through Canvas before Midnight (by 11:59pm) on February 8th. Thus, each group will have two distinct submissions in Canvas for this assignment. The assignment should be done in groups of two students. Each submission must contain the full name, OSU email, and ONID of every member of the group.

1: Constraint Inference (0.5 point)

(a) Given that X, Y, W, Z are attributes in a relation, using Armstrong's axioms, prove that if we have $X \rightarrow Y$ and $YW \rightarrow Z$, then $XW \rightarrow Z$. (0.25 point)

(b) Given that X, Y, Z are attributes in a relation, using Armstrong's axioms, prove that if we have $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$. (0.25 point)

2: Schema Decomposition (2 points)

Consider the relation schema R with attributes A, B, C, and D and the following functional dependencies: AB→C, AC→B, B→D, BC→A.

(a) List all keys for R. (0.5 point)

(b) Is R in BCNF? If it is not, decompose it into a collection of BCNF relations. (0.5 point)

(c) Is R in 3NF? If it is not, convert it into a collection of 3NF relations. (0.5 point)

(d) Prove that, if relation R has only one simple key, it is in BCNF if and only if it is in 3NF. (0.5 point)

3: Information preservation (1.5 points)

(a) Suppose you are given a relation R(A,B,C,D) with functional dependencies B→C and D→A. State whether the decomposition of R to S_1(B,C) and S_2(A,D) is lossless or dependency preserving and briefly explain why or why not. (0.5 point)

(b) Prove that the 3NF synthesis algorithm produces a lossless-join decomposition of the relation containing all the original attributes. (1 point)

4: Sorting on external storage (5 points)

(a) Consider a file with records of following structure:

```
Emp (eid (integer), ename (string), age (integer), salary (double))
```

Fields of types *integer*, *double*, and *string* occupy 4, 8, and 40 bytes, respectively. Assume that each (I/O) block can fit at most one record (tuple) of the input file. Implement the two-pass multi-way sorting for the file ***Emp.csv*** in C/C++ using the skeleton code posted with this assignment. The sorting should be based on the attribute ***eid***. There are at most 22 blocks available to the sort algorithm in the main memory, i.e., the **size of the buffer is 22**.

- The input relation is stored in a CSV file, i.e., each tuple is in a separate line and fields of each record are separated by commas.
- The result of the sort must be stored in a new CSV file. The file that stores the relation *Emp* are Emp.csv.
- Your program must assume that the input file is 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 **EmpSorted.csv** in the current working directory.
- Your program must run on Linux. Each student has an account on *hadoop-master.engr.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 it:

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

  Then it asks for your ONID password and probably one another question. You can access this server on campus. If you want to access from outside of campus, you'll need to use VPN to access the campus network.
- You must name the file that contains the source code of the main() function **main4.cpp**.
- You may use following commands to compile and run C++ code:

  ```
  > g++ -std=c++11 main4.cpp -o main4.out
  > main4.out
  ```