DATABASE SYSTEMS (CSE 441)

Assignment 2

Deadline: Friday, 12th Feb 9pm

Instructions:

1. You are not allowed to use any external library/jar files in this assignment.
2. **Plagiarism will not be tolerated.**
3. Be careful about your submissions. You must strictly follow the upload format.
4. Languages allowed to code are C/C++/Java.

Two-phase Merge Sort Algorithm:

You have to implement two-phase merge sort algorithm to sort large number of records with the specifications mentioned below :

1. **Metadata information:**
   - **a.** This information will be provided in a file as an input to your algorithm and will contain the information about the columns and their data type.
   - **b.** The number of columns can vary from 1 to 20.
   - **c.** The data type for the column can be String, Integer or Date.
     - i. **Date**: It will be represent by date. e.g. col1,date or mycol,date etc.
     - ii. **String**:It will be represented by char(<length of string>) i.e. every string will be of same size defined by <length of string> e.g. cols,char(10) or column,char(11) etc.
     - iii. **Integer**: e.g. It will be represented by int. col1,int or mycol,int etc.
2. **Input Data**:
   - **a.** Each record will start from newline.
   - **b.** Column values will be separated by comma.
   - **c.** String will comprise of alphanumeric characters only i.e. A-Za-z0-9
   - **d.** Date will be in the format DD/MM/YYYY format.
3. **Output Data:**
   - **a.** Output file will contain the columns that you have specified as an argument to your algorithm. So, the number of columns can differ in output file as compared to input file.
   - **b.** Each record will be separated by new line and column values by comma (similar to input data format).
4. **Main memory:**
   - **a.** The main memory that is allowed to use will be specified by command line argument.

   - **b.** You will need to check if the two phase merge sort is feasible or not in the provided memory as argument. If not, proper **error message** needs to be generated.
   - **c.** For C/C++, use the dynamic memory allocation malloc and request 80% of the main memory specified in MB from the underlying OS and use the same memory for sorting.
   - **d.** For Java, use the argument -Xmx to restrict the heap size equal to 10 times memory specified in MB.

5. **Sorting columns:**
   - **a.** If the sorting needs to be done using multiple columns then if the values of the first column is same, you need to sort on the basis of the second and so on.
   - **b.** If the value of all the columns mentioned for sorting is same, for such records the order in the output should be maintained as the order in the input file.

Different file formats:

1. **metainfo.txt:**

   <column name 1>,<datatype of the column>
   <column name 2>,<datatype of the column>
   <column name 3>,<datatype of the column>
   ......
   <column name n>,<datatype of the column>

   Example:

   ```
   col1,date
   col2,char(11)
   col3,int
   col4,date
   ```

2. **Input File:**

   <value1 of col1>,<value1 of column>,<value1 of col3>,<value1 of col2>
   <value1 of col1>,<value1 of column>,<value1 of col3>,<value1 of col2>
   <value1 of col1>,<value1 of column>,<value1 of col3>,<value1 of col2>

   Example:

   ```
   10/02/2015,SDERqwespoq,84382,10/03/2015
   11/02/2015,SDERqwespoq,84,10/04/2016
   10/02/2016,SDERqddspoq,84821,11/03/2015
   ```

Command Line Flags:

1. **--meta_file** <Metadata file path> i.e metadata file containing the metadata information.
2. **--input_file** <Input file path> i.e input file containing the raw records.
3. **--output_file** <output file path> i.e. Ouput file containing sorted records
4. **--output_column** <list of columns that will be present in output file separated by comma>
5. **--sort_column** <list of columns that will be used to sort the records>
6. **--mm** <size for main memory in MB> Main memory size (in MB)
7. **--order** <asc/desc> i.e. asc means to sort in ascending order and desc means to sort in descending order

Example:

Let us consider input file containing the columns col1, col2,col3, col4

```
./mysort --meta_file metafile.txt --input_file input.txt --output_file output.txt --output_column
col1,col2 --sort_column col1,col2 --mm 1040 --order asc
```

File input.txt to be sorted in ascending order with 1040 MB space based on column col0 and col2, if any column have same value for col0 then on the basis of col1, available. The column that will be present in the output file will be col1,col2,col3,column4 (Similar to ORDER BY clause in SQL).

```
./mysort --meta_file metafile.txt --input_file input.txt --output_file output.txt --output_column
col3,col2,column4 --sort_column col1,col2 --mm 50 --order asc
```

```
./mysort --meta_file metafile.txt --input_file /home/user/mydir/db/input.txt --output_file
/home/user/mydir/db/output.txt --output_column col1,col2,col3,column4 --sort_column
col2,col1 --mm 50 --order desc
```

Graph Generation:

1. You can use any language or tool to plot the graph. (Matlab, Ms Excel).
2. You need to provide the matrix and the image file of the graph in a pdf document named as “analysis.pdf”.

Generate two graphs:

1. **FileSize(X-axis) v/s Time(Y-axis)**

Considering the main memory size fixed as 100 MB, run your code for file size 5MB, 50MB, 500MB, 1GB, 2GB and note the time taken for each run. Plot the graph.

2. Memory Size (X-axis) v/s Time (Y-axis)

Considering the input file fixed as 512MB, run your code with main memory as 10 MB, 25 MB, 100 MB, 250 MB and note the time taken for each run.Plot the Graph.

Submission format:

Create a folder with name rollno_Assignment2 and put the following into it:

a) “**code**” folder : Your source code in the folder named as code. (It should not contain the code used for graph generation)

b) “**analysis.pdf”** A pdf file with the name analysis.pdf containing following information

i) Configuration of the System

ii) Both matrix used to plot the graph (in tabular format)

iii) Both the image files of the graph

c) “**rollno.sh**” A bash file with the name rollno.sh that should take the required arguments as mentioned in the command line flag and should be able to compile your code and run your code. In case we find your bash file missing, your assignment will not be evaluated.

Compress the folder and upload the **rollno_Assignment2.tar.gz**

```
rollno_Assignment2
        |--------code
                |---- all the source code files
        |--------analysis.pdf
        |--------rollno.sh
```

Note: Do Not upload any extra files like input/output or the metadata files.