DSI Assignment 2.1 Report

Shlok Gilda, UFID: 39548312, shlokgilda@ufl.edu
Aditi Telang, UFID: 13277280, atelang1@ufl.edu

Compilation directions:

We have implemented the necessary code. Use the following instructions to compile the code.

Run test (test.cc)

1. make test.out
2. ./test.out

Run gtest (gtest.cc)

1. make gtest.out
2. ./gtest.out

Function description:

We have completed the BigQ class by writing the required functions. The job of the BigQFile within the system is to sort the records based on the input predicate, write to console and the store the sorted file with.bigq extension.

Implementation Details:

We have created the following private fields in the BigQ.h.

Pipe *inputPipe; // Input pipe to get records.
Pipe *outputPipe; // Output pipe to push records.
OrderMaker *sortedOrder; // Sorted order required for sorting.
int *runLength; // Run length.
File *runsFile; // File pointer for the runs file.
vector<int> runPointers; // List of pointers to all the runs.

Method Description:

- **BigQ::BigQ(Pipe &in, Pipe &out, OrderMaker &sortorder, int runlen)**
    - This is the constructor of the BigQ class. A new “Runs” file is created to store the runs.

- **static void* invoke_tpmmsAlgo(void* args)**
    - This function is the start of the worker thread. It performs the sorting. We are typecasting the void arguments to bigQ and invoking the worker function.

- **void BigQ::worker()**
    - Worker function which does the TPMMS Algorithm. The algorithm is written in two phases.
    - The first phase breaks the File into runs and sorts each run.
    - In the second phase, all the runs are merged into a single sorted file using priority queue.

- **void BigQ::sortRun(vector <Record*> &)**
    - This function sorts the vector of records using the input predicate.

- **int BigQ::addRuntoFile(vector <Record*> &)**
    - This function takes input of sorted records into a vector and writes them to File and returns the current size of the file which acts as a pointer to the next run.

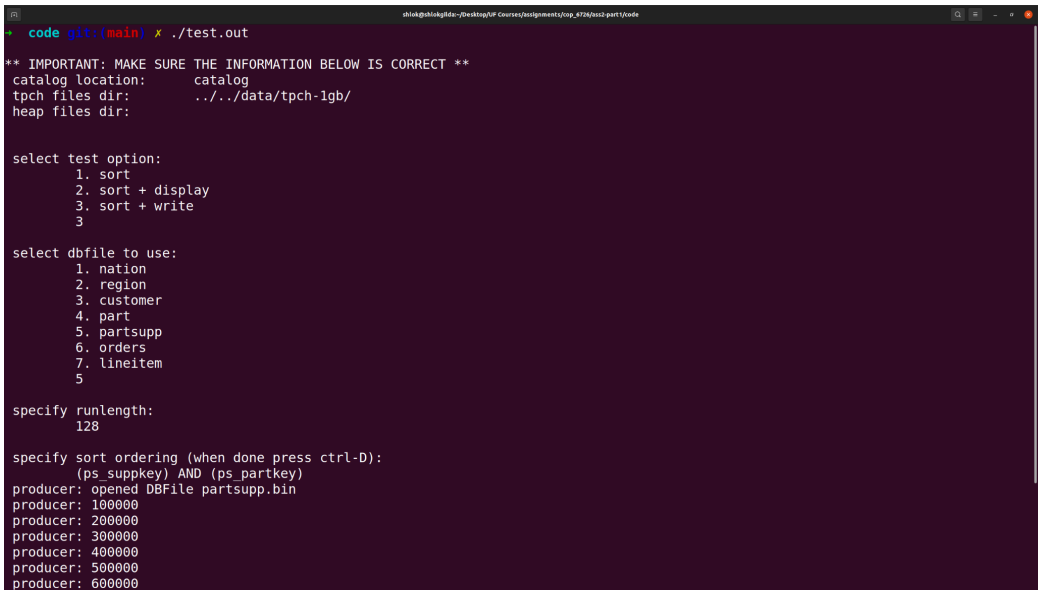
GTest test cases:

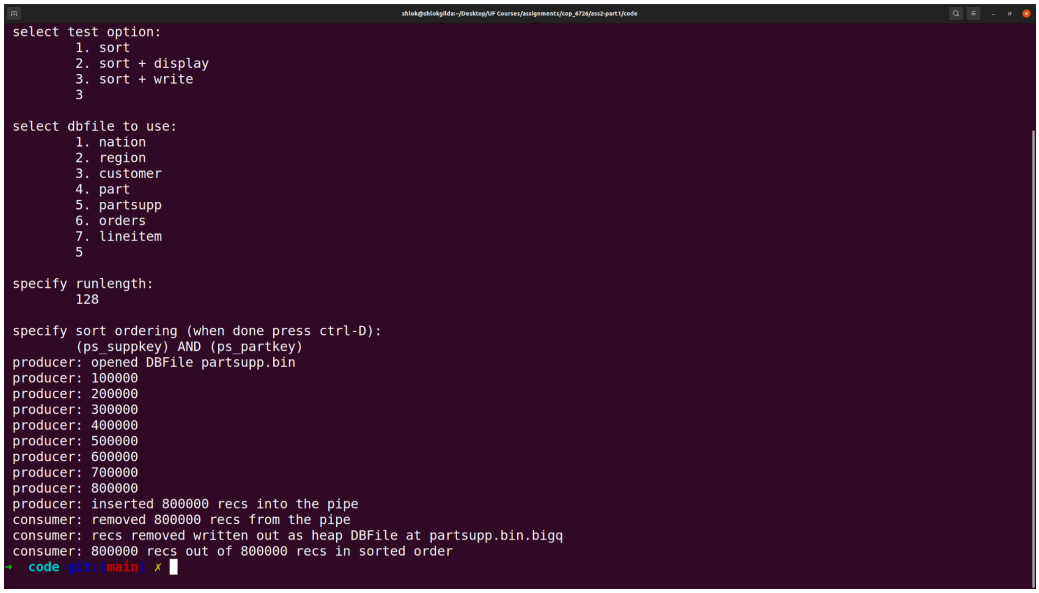
Tests:

1. Check whether the nation DB file is sorted or not.
2. Check whether the customer DB file is sorted or not.
3. Check whether the part DB file is sorted or not.
4. Check whether the lineitem DB file is sorted or not.

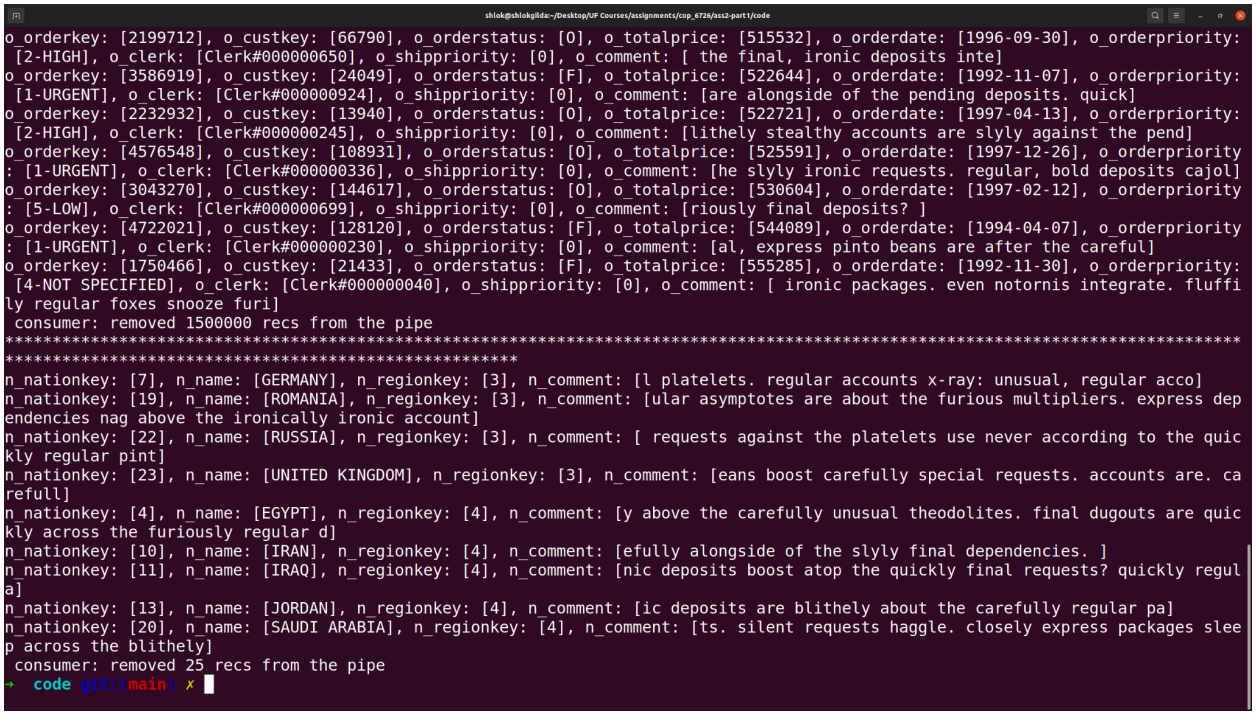
Test case results:

1. The following is a screenshot of the output of “test.out”. The test case that we ran was:
    a. **Test option:** 3. Sort + Write
    b. **Dbfile:** 5. partsupp
    c. **Runlength:** 128
    d. **CNF:** (ps_suppkey) AND (ps_partkey)





2. This is the output of running “./runTestCases.sh”. The “output1.txt” has been attached with the code. The screenshot below shows only a sample of “output1.txt”.



3. The following screenshots show the results of gtest.