COP 6726: DATABASE SYSTEM IMPLEMENTATION
Assignment 2.1: Sorted File Implementation

Group Info:

	Name	UFID
1.	**Manishkumar Chopra**	**1796-7121**
2.	**Neel Manish Rami**	**7712-3151**

I. INTRODUCTION

In this assignment, we have implemented the BigQ class. The BigQ class encapsulates the process of taking a stream of inserts, breaking the stream of inserts into runs, and then using an in-memory priority queue to organize the head of each run and give the records to the caller in sorted order. We have also written unit test cases using google test unit testing library for every function which we have implemented in DBFile class.

II. ASSUMPTIONS

A. gcc (or clang) must be installed together with “bison” and “flex” on the machine on which the project will be tested.
B. Google Test Unit Testing Library should also be installed and compiled on the machine on which the project will be tested.
C. Google Test Unit Testing Library is not installed on CISE Thunder Server so the gTest command won’t run on thunder.

III. FILE STRUCTURE

A. bin/ : Contains object(.o) files.
B. src/: Contains .cc and .h files.
C. table_files/: Contains .tbl files which were generated using tpch data generator.
D. gtest/: Google Test Unit testing library.
E. catalog: Catalog file for the schema of .tbl files
F. MakeFile: Make File.

IV. CODE EXPLANATION

Data Structures used in BigQ Class:-

1. **struct TPMMS_DS:** This structure is used during the algorithm.
2. **struct RecordSorter:** This struct is used by the vector of Records to sort records.
3. **struct RecordComparator:** This struct is used by a priority queue.
4. **struct Record1:** It's a wrapper around Record Class which has an instance of Record Class and an integer which stores the run number to which the record belongs.

Public Functions used in BigQ Class:-

1. **void* TPMMS_Algo(void* args)**

This method implements the TPMMS Algorithm/

V. COMMANDS TO RUN THE PROGRAM

To compile the given test driver, type
make test.out

To run the test driver, type
./test.out
and then follow the on-screen instructions. For scan and filter operation, write any CNF which has been given in the result section below. After writing the CNF, press ENTER and then press Ctrl+D.

The thunder server at cise.ufl.edu doesn't have Google Test installed so you might have to download it.

To compile Google Test, type
make gTest2

To run the unit tests, type
cd bin
./gTestProj2

The unit tests should run and you should be able to see the test results.

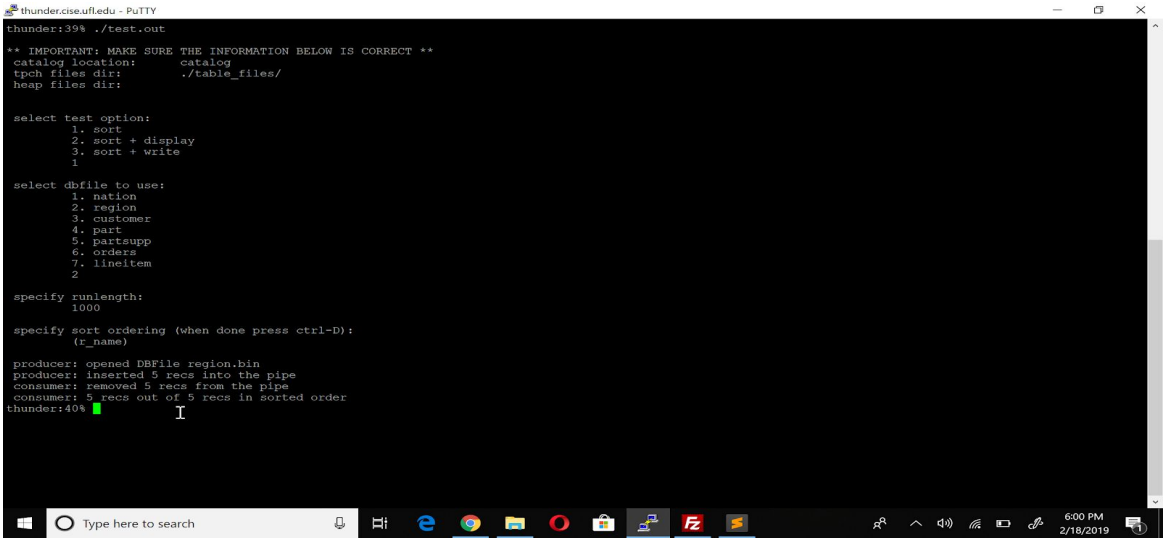
VI. RESULTS

We ran our program on 10M dataset.

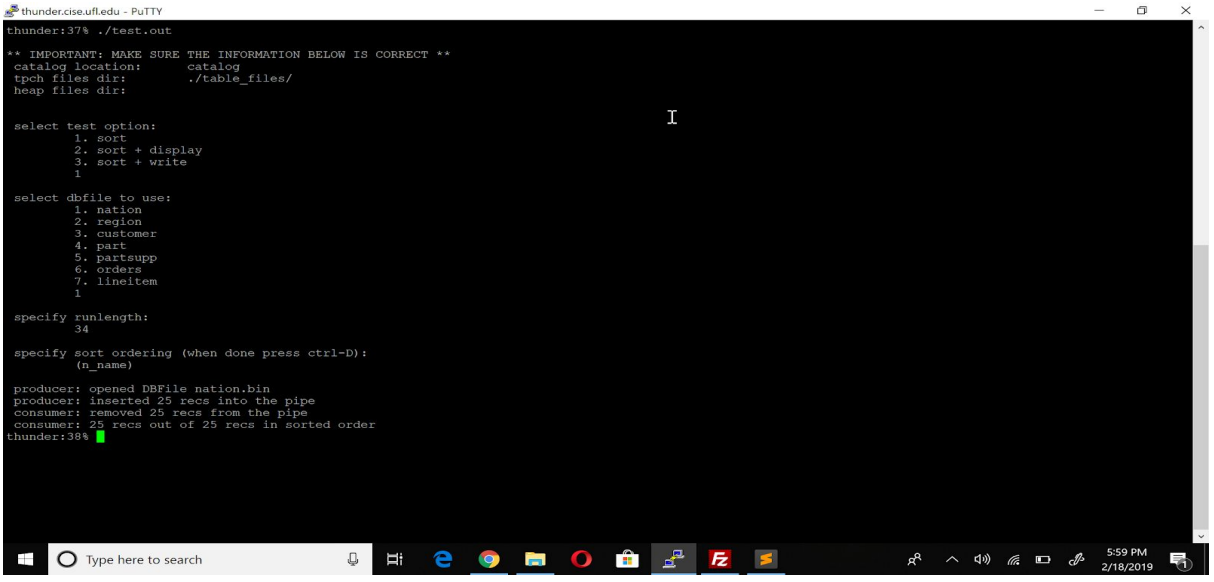
A. 10M Data

We verified our results with the results which have been provided by the professor in the assignment folder.

Query 1 (Sort): (r_name)

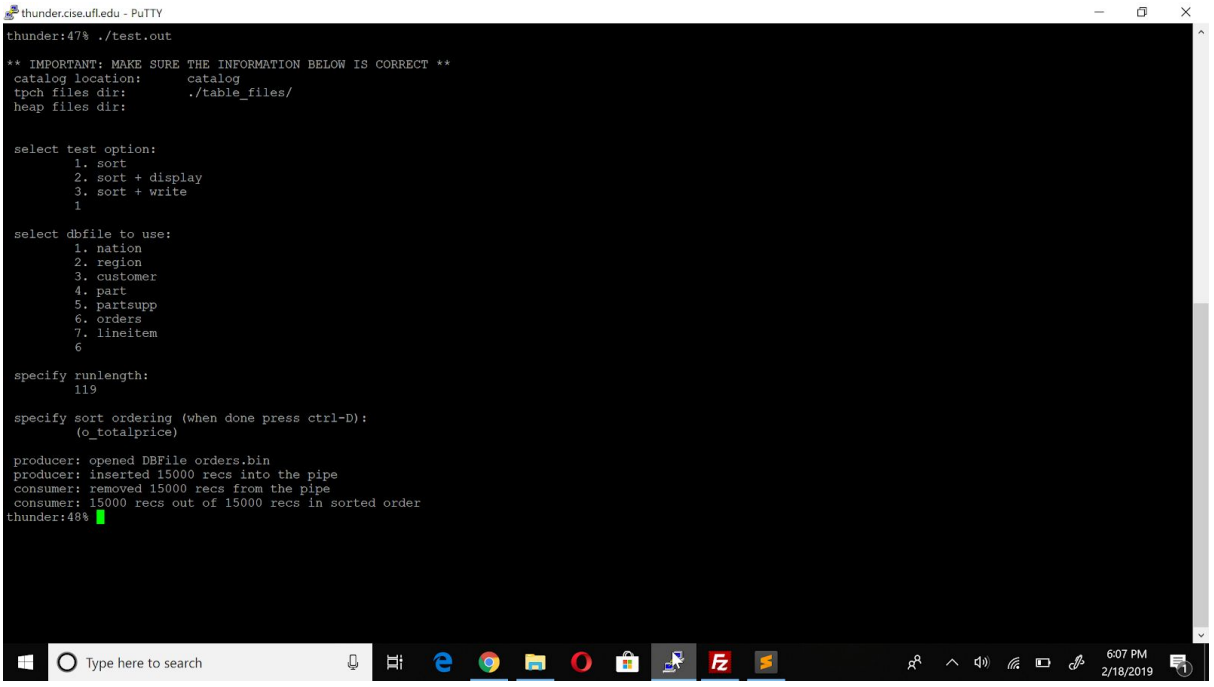


Query 2 (Sort): (n_name)



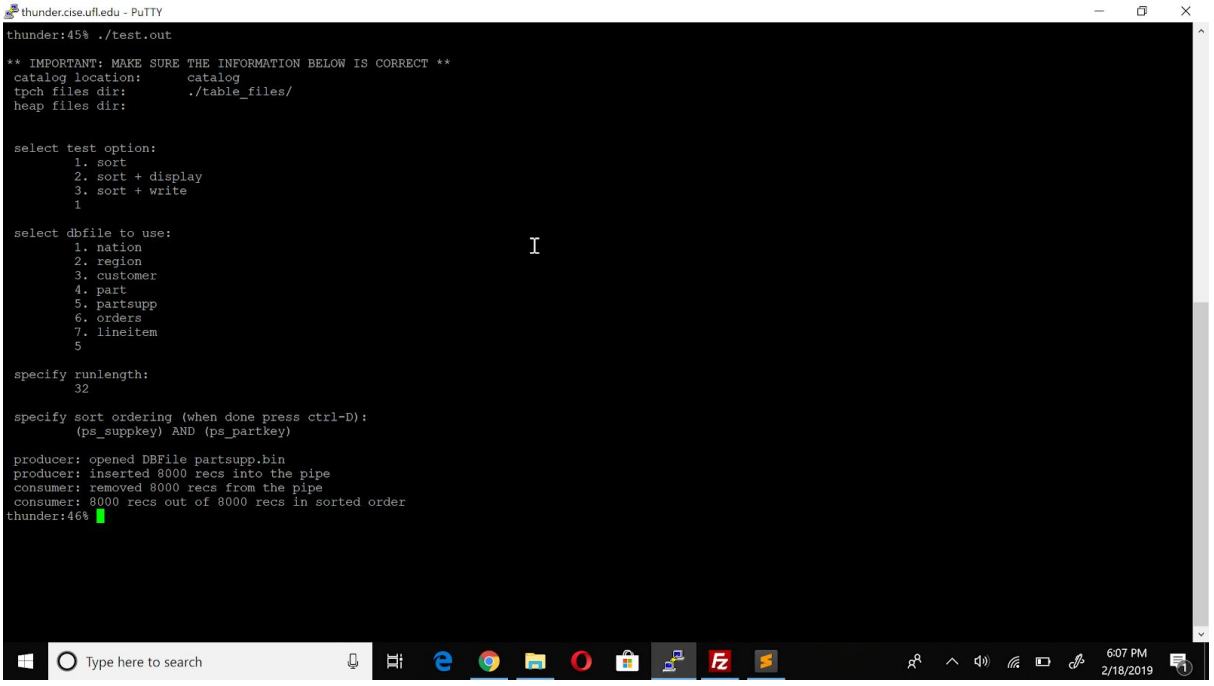
Query 3 (Sort):

(o_totalprice)



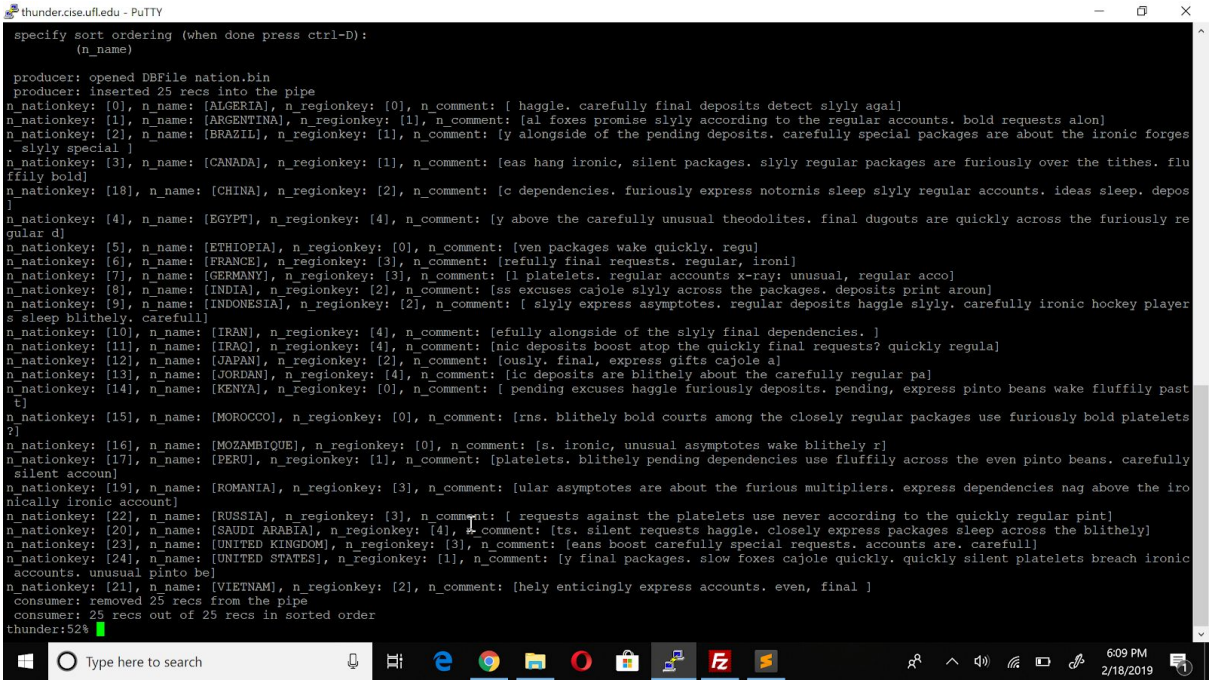
Query 4: (Sort)

(ps_suppkey) AND (ps_partkey)



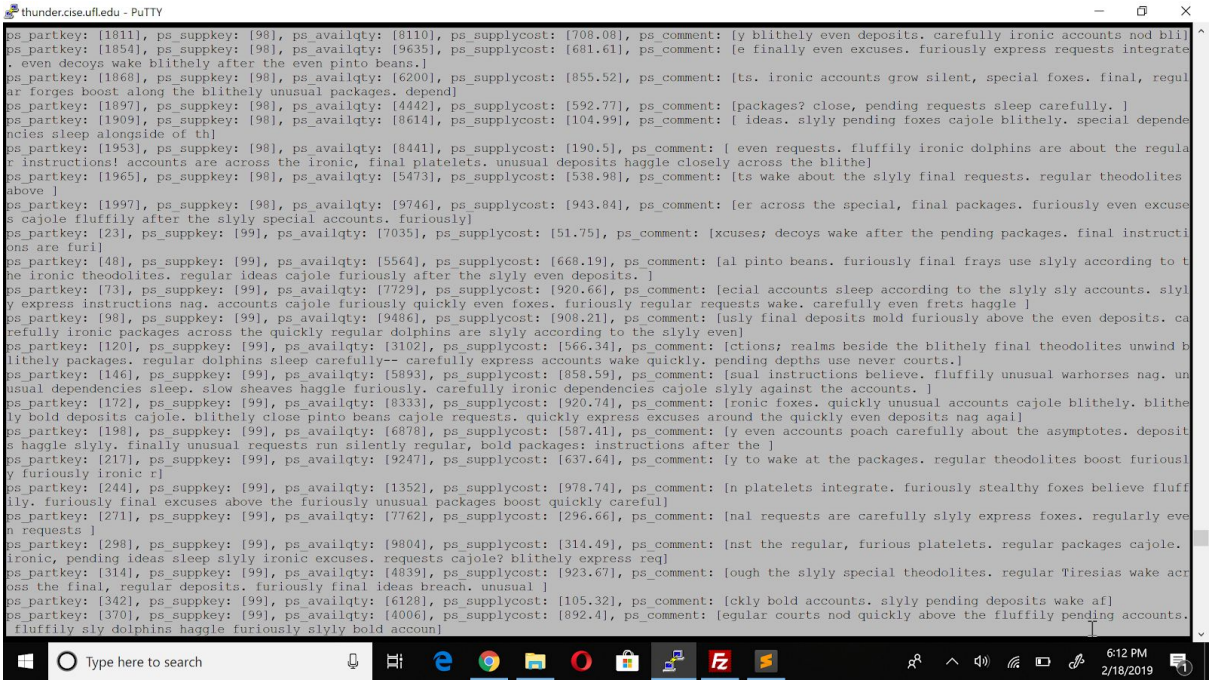
Query 5 (Sort + Display):

(n_name)



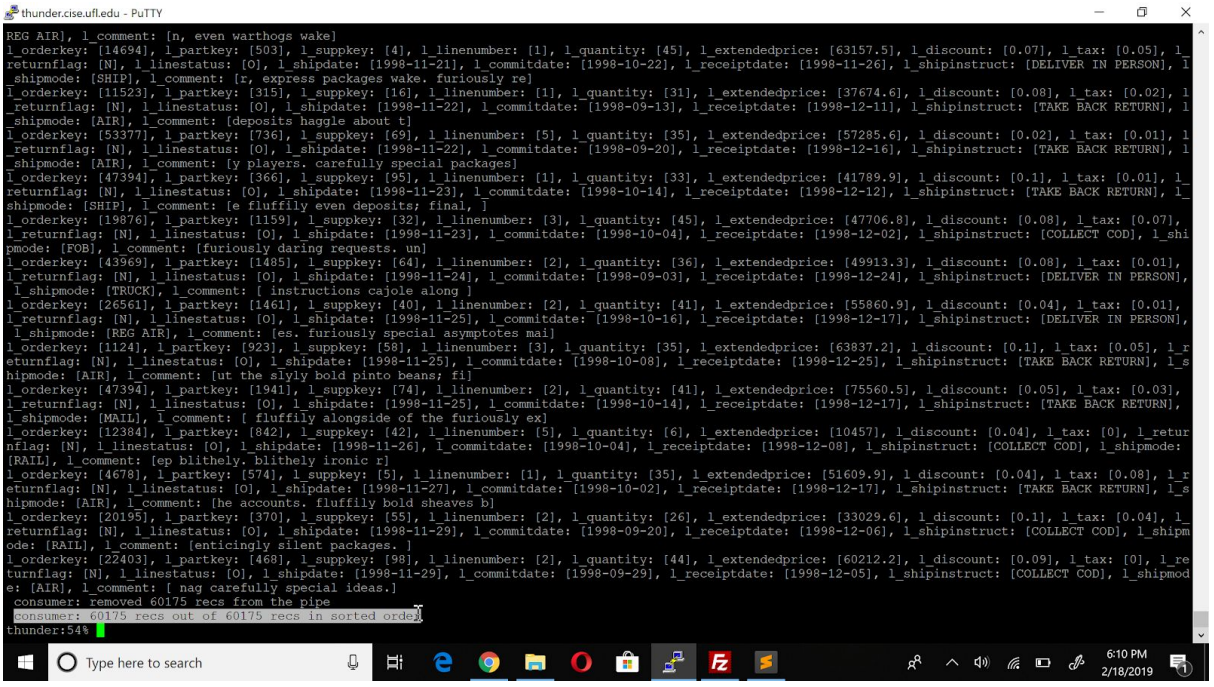
Query 6 (Sort + Display):

(ps_suppkey) AND (ps_partkey)



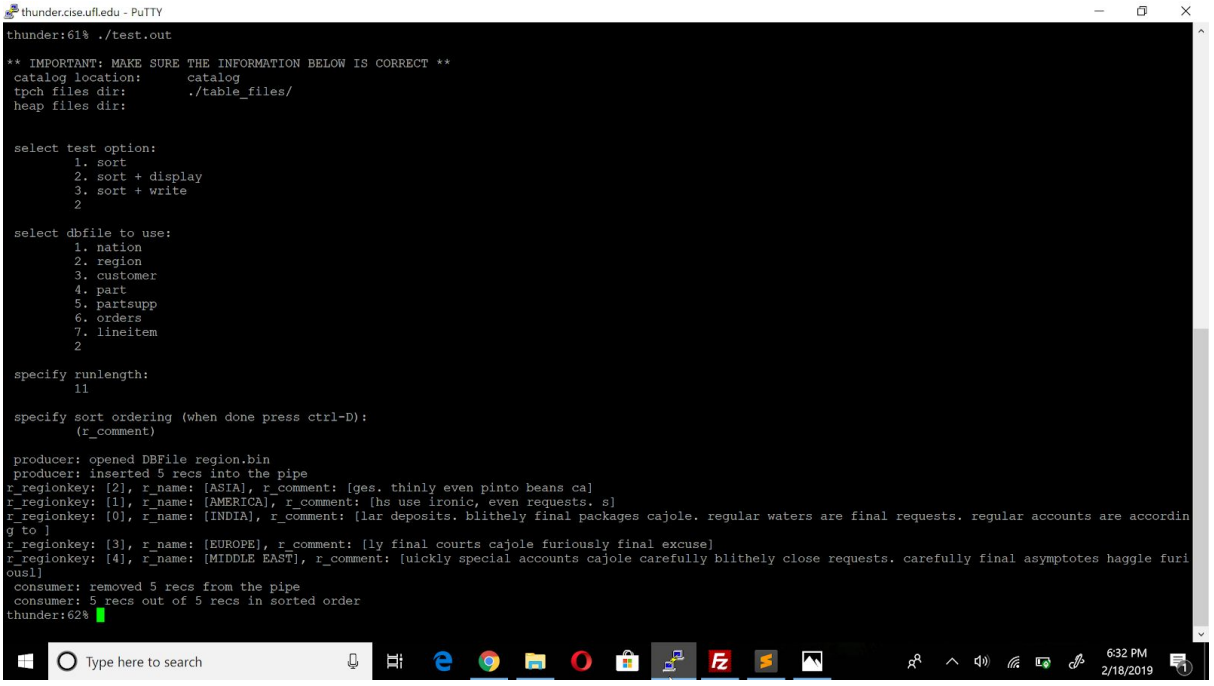
Query 7 (Sort + Display):

(l_shipdate) AND (l_extendedprice) AND (l_quantity)



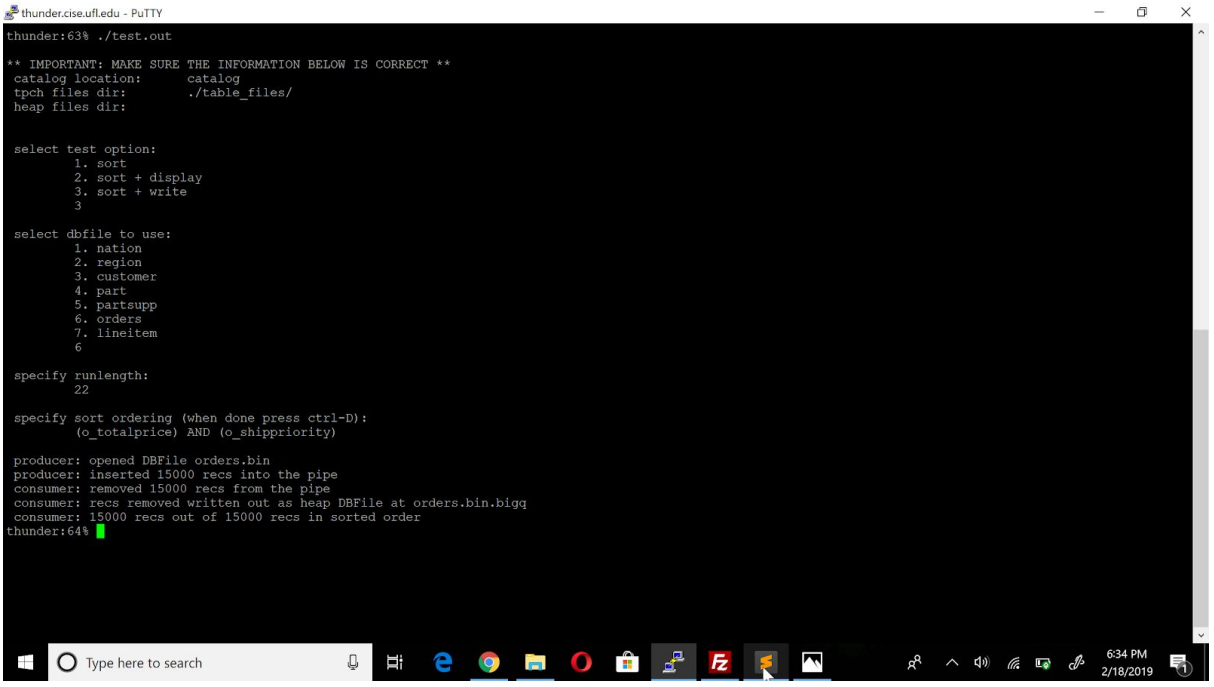
Query 8 (Sort + Display):

(r_comment)



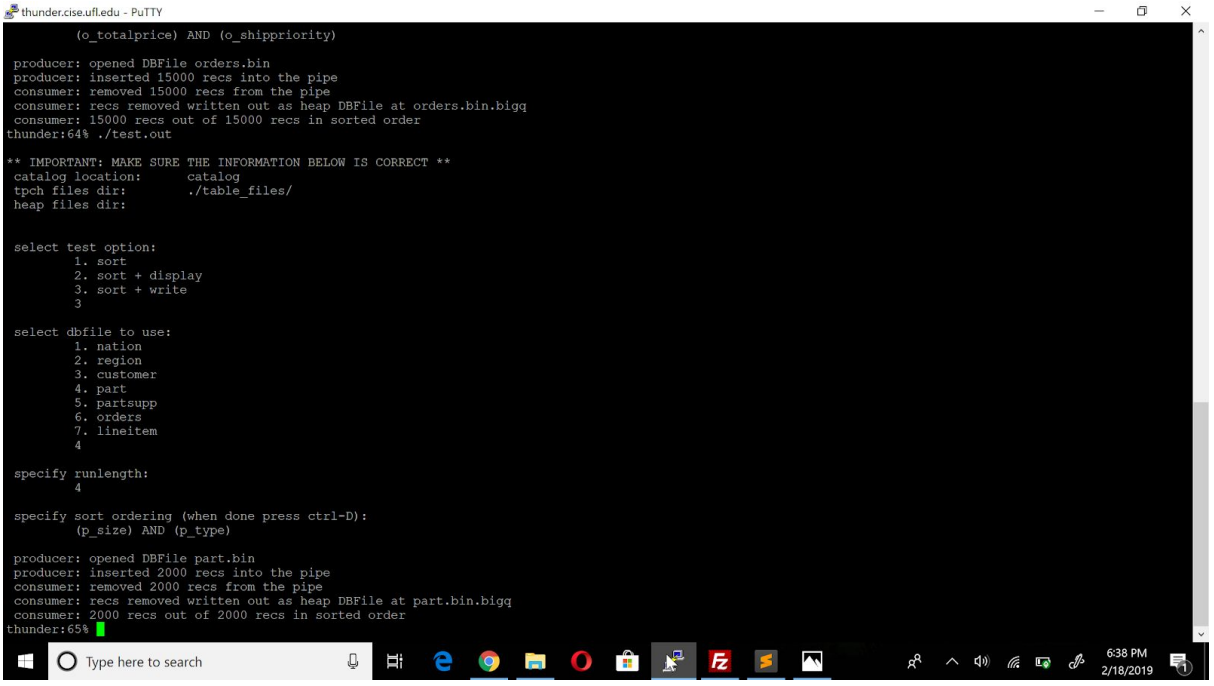
Query 9 (Sort + Write):

(o_totalprice) AND (o_shippriority)



Query 10 (Sort + Write):

(p_size) AND (p_type)



Google Test Results: