COP 6726: DATABASE SYSTEM IMPLEMENTATION
Assignment 1: Heap 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 DBFile class. The job of the DBFile class within the database system is simply to store and retrieve records from the disk. We have also written unit test cases using google test unit testing library for every function which we have implemented in DBFile Class. We have also recorded the execution time of each query and also recorded the time taken to scan all the records.

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

Private Data Structures used in DBFile Class:-

1. **File* myFile:** DBFile Class is built on top of File Class.
2. **Record* currentRecord:** This variable points to the current record of the DBFile.
3. **Page* myPage:** DBFile Class is built on top of Page Class.
4. **bool pageWritten:** This variable indicates whether the current page has been written to the DBFile or not.
5. **Off_t currentPageIndex:** This variable specifies the current page index.

Public Functions used in DBFile Class:-

1. **void Add (Record &addMe);**

   This function is used to add records to the file. In the case of the unordered heap file that has been implemented in this assignment, this function simply adds the new record to the end of the file.

2. **int Create (char *name, fType myType, void *startup);**

   This function is used to create the file.

3. **int Open (char *name);**

   This function is used to open a file that has already been created and also being closed.

4. **int Close();**

   This function simply closes the file.

5. **void Load (Schema &mySchema, char *loadMe);**

   This function bulk loads the DBFile instance from a text file, appending new data to it using the SuckNextRecord function from Record.h.

6. **int GetNext (Record &fetchMe);**

   This function simply gets the next record from the file and returns it to the user.

7. **int GetNext (Record &fetchMe, CNF &applyMe, Record &literal);**

   This function accepts a selection predicate and returns the next record in the file which is accepted by the selection predicate.

8. **void MoveFirst ()**

   This function forces some pointer to correspond to the first record in the file.

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 gTest

To run the unit tests, type
cd bin
./gTestProj1

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

VI. RESULTS

We ran our program on 2 sets of data which was generated by tpch data generator. One set of data is 1G and other is 10M.

A. 1G Data

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

Query 1

(r_name = ‘EUROPE’)

```
thunder:74% ./test.out

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      catalog
 tpch files dir:        /cise/tmp/dbi_sp11/DATA/1G/
 heap files dir:


 select test:
         1. load file
         2. scan
         3. scan & filter
         3

 select table:
         1. nation
         2. region
         3. customer
         4. part
         5. partsupp
         6. orders
         7. lineitem
         2
 Filter with CNF for : region
 Enter CNF predicate (when done press ctrl-D):
        (r_name = 'EUROPE')
r_regionkey: [3], r_name: [EUROPE], r_comment: [ly final courts cajole furiously final excuse]
Selected 1 Records.
Query Time 6.8e-05seconds.
thunder:75%
```

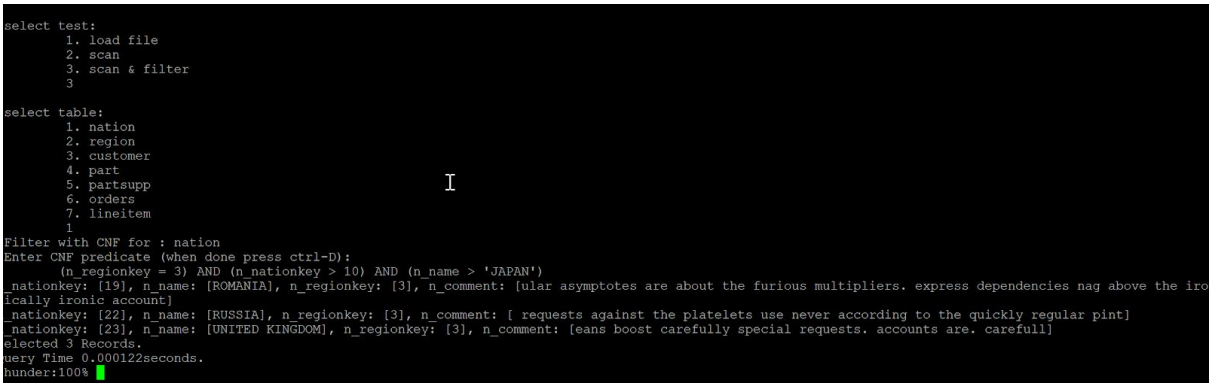
Query 2

(r_name = ‘MIDDLE EAST’) AND (r_regionkey > 1)

```
thunder:76% ./test.out

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      catalog
 tpch files dir:        /cise/tmp/dbi_sp11/DATA/1G/
 heap files dir:


 select test:
         1. load file
         2. scan
         3. scan & filter
         3

 select table:
         1. nation
         2. region
         3. customer
         4. part
         5. partsupp
         6. orders
         7. lineitem
         2
 Filter with CNF for : region
 Enter CNF predicate (when done press ctrl-D):
        (r_name < 'MIDDLE EAST') AND (r_regionkey > 1)
r_regionkey: [2], r_name: [ASIA], r_comment: [ges. thinly even pinto beans ca]
r_regionkey: [3], r_name: [EUROPE], r_comment: [ly final courts cajole furiously final excuse]
Selected 2 Records.
Query Time 0.00011seconds.
thunder:77%
```

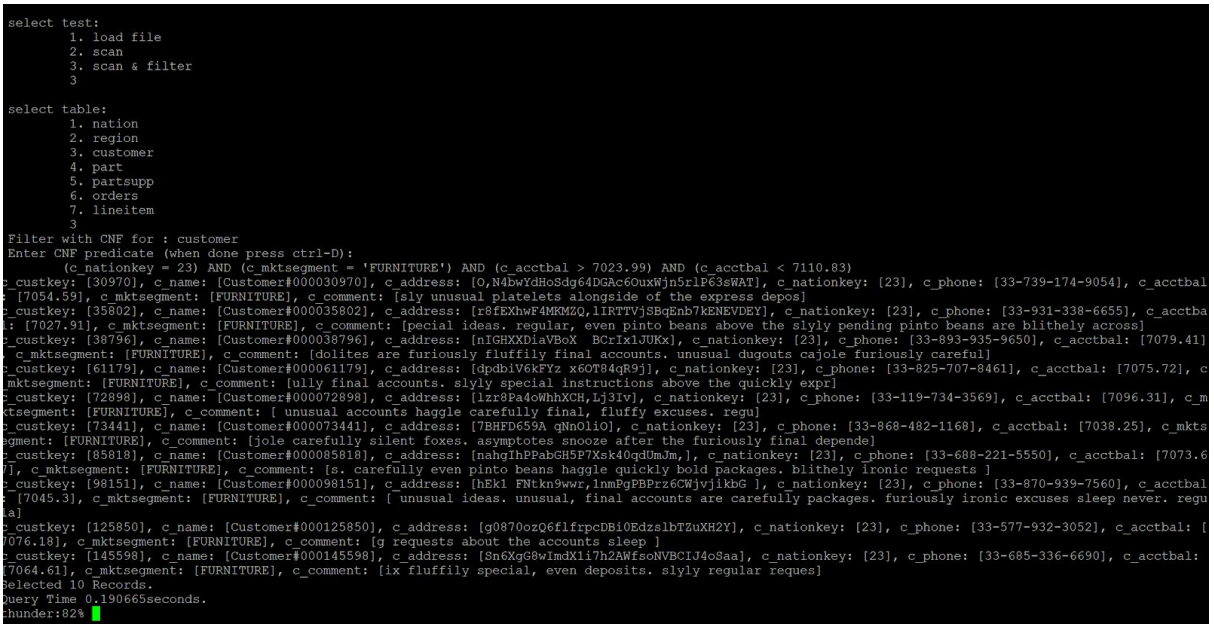
Query 3

(n_regionkey = 3) AND (n_nationkey > 10) AND (n_name > 'JAPAN')



Query 4

(c_nationkey = 23) AND (c_mktsegment = 'FURNITURE') AND (c_acctbal > 7023.99) AND (c_acctbal < 7110.83)

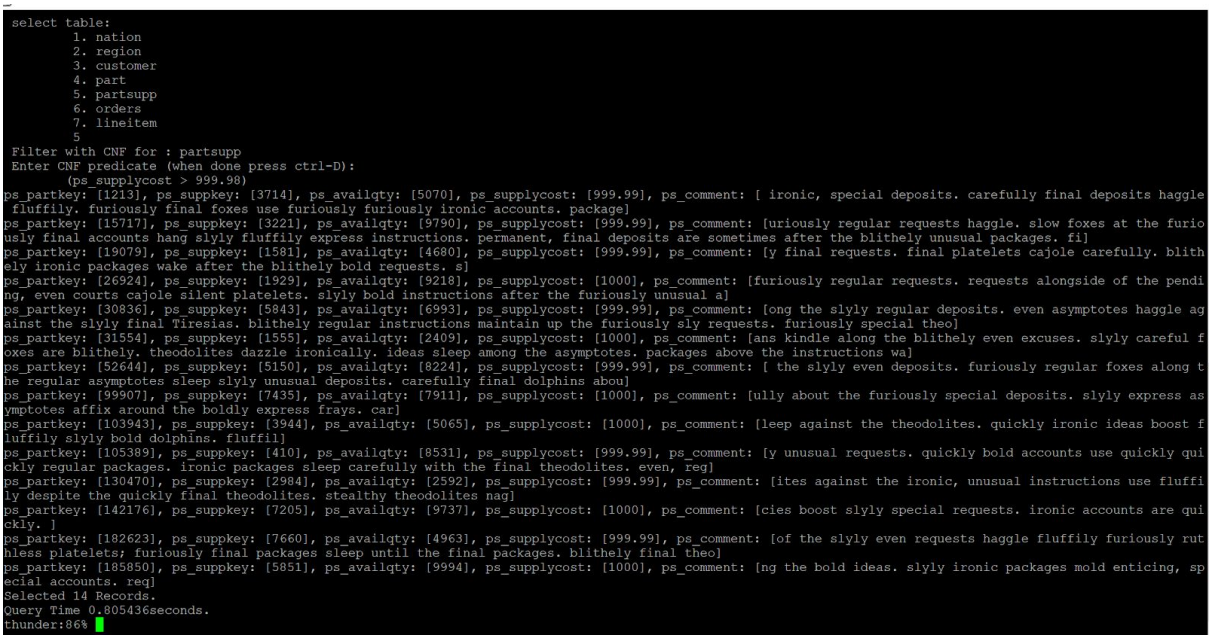


Query 5

(p_brand = 'Brand#13') AND (p_retailprice > 500.00) AND (p_retailprice < 930.00) AND (p_size > 28) AND (p_size < 1000000)

Query 6

(ps_supplycost > 999.98)



Query 7

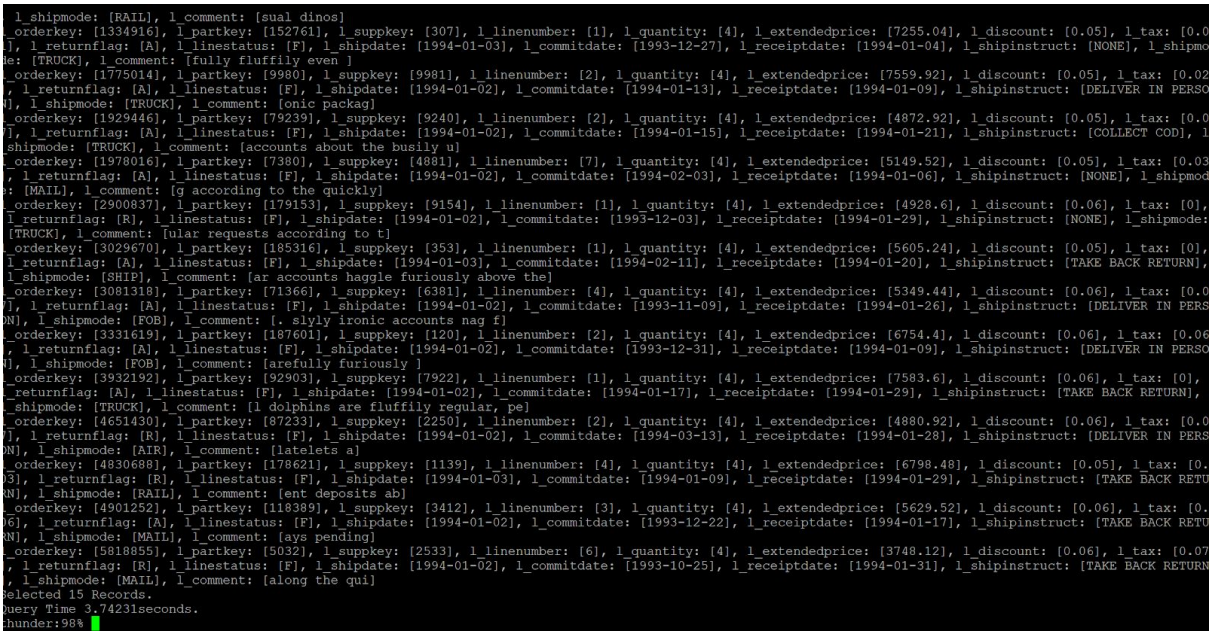
(ps_availqty < 10) AND (ps_supplycost > 100.00) AND (ps_suppkey < 300)

Query 8

(o_orderpriority = '1-URGENT') AND (o_orderstatus = 'O') AND (o_shippriority = 0) AND (o_totalprice > 1015.68) AND (o_totalprice < 1051.89)

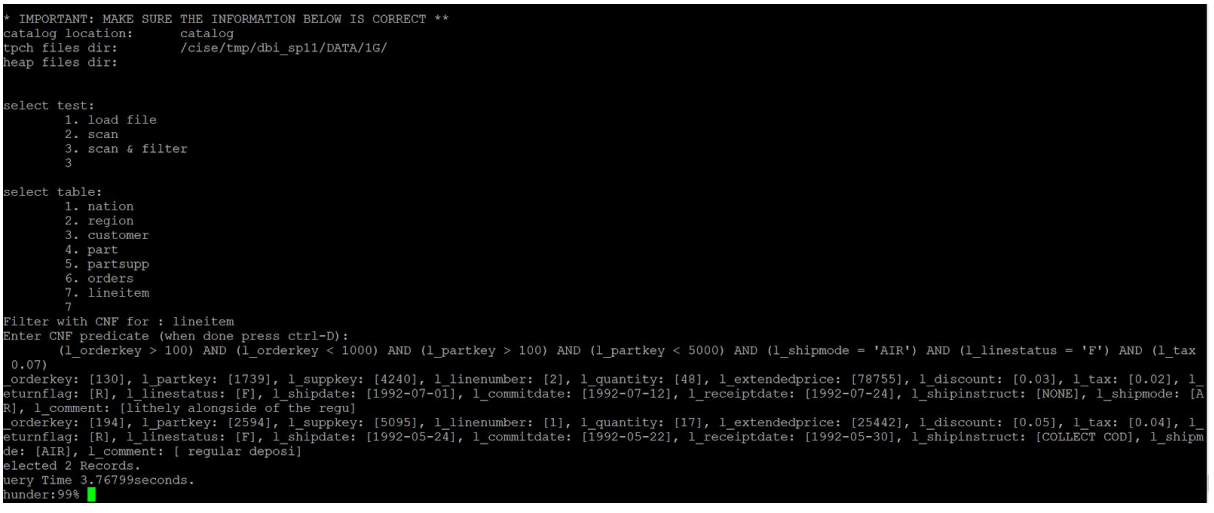
Query 9

(l_shipdate > '1994-01-01') AND (l_shipdate < '1994-01-04') AND (l_discount > 0.04) AND (l_discount < 0.07) AND (l_quantity = 4.00)



Query 10

(l_orderkey > 100) AND (l_orderkey < 1000) AND (l_partkey > 100) AND (l_partkey < 5000) AND (l_shipmode = 'AIR') AND (l_linestatus = 'F') AND (l_tax < 0.07)



We also ran our results on 10M data and verified those results by writing SQL queries in MariaDB.

Google Test Results: