DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE AND ENGINEERING UNIVERSITY OF FLORIDA

SPRING 2022

Database System Implementation (COP 6726)

PROJECT – II Part I

IMPLEMENTING SORTED FILE

Venkata Vynatheya J - UFID: 7836-3701

Chinmai Sai Eshwar Reddy Kasi - UFID: 9742-3328

ABSTRACT:

DBFile is used to store and retrieve records from disk. In this assignment we are building a functionality on top of DBFile, inorder to implement a sorted and a heap file. To do this, we will use BigQ class which will be used by sorted file to do it's sorting. BigQ is a disk-based priority queue which runs on TPMMS algorithm. BigQ implements these functionalities using threading. It calls several methods on the Pipe and organizes the chunks of data.

Changes to Source Files:

We tweaked the Comparison.cc file a little bit as the comparison between the records was not working as expected. To make it work we commented the line 146 in Comparison.cc

How to run the Project:

The makefile generated in this project is specific to linux system. It might not work as expected in other operating systems. Steps for compiling and running the project are as below. Please make sure you are in the P1 directory of Project 2 before running any of the below commands

1. make clean
2. make
3. ./test.out

In the test.cc file, you will find the path for tpch and db data. Please use the same path or change the path according to your structure.

To execute the testcases please use the following commands

./runTestCases.sh

For compiling and running gtest, refer to the end of this report.

Classes and Functions Definitions:

1. **TPMMS Class:**

   TPMMS class contains all the variables with information needed to implement the TPMMS algorithm. It contains the details regarding collection of pages, first page, current page and the run length. During the execution of the TPMMS algorithm phases, this class will hold the records to be retrieved, sorted and to be put into the output pipe. We use the below methods to implement this function

   a. **TPMMS::TPMMS(File* f, int start, int l):**

      Here we build the constructor to the file and give the file to be sorted, the start index from the batch of records and the number of records from the start index.

   b. **int TPMMS::firstRecordUpdate()**

      This page returns the topmost page from the stack and also gives us the option to update it. It'll return null if no more records are left.

2. **CompareBuffers Class:**

   This class is a buffer comparator. It will compare the records in two different buffers using the OrderMaker that has been provided.

   a. **CompareBuffers::CompareBuffers(OrderMaker* orderMaker):**

      This is a constructor for compareBuffers which takes orderMaker generated for the given CNF, as a parameter

   b. **bool CompareBuffers::operator () (TPMMS* l, TPMMS* r):**

      This is a default comparison operator that will compare two buffers and sort the records.

3. **Class BigQ:**

   This class serves as the foundation for sorting records.

   a. **void* tpmmsMainProcess():**

      This is the main function which implements the TPMMS algorithm. Just like the algorithm the function has 2 phases. In phase one all the records are gathered from input pipe into pages until the run length is reached. And after the records are stored in the buffers, it will then compare, sort and merge them and push them to the priority queue to be written back in the output file.

   b. **void* tpmmsPipeline():**

      This function is used to create the buffer/pipeline to hold the records which are later used for comparing and sorting.

4. **CompareRecords Class:**

   This class is a record comparator. It will compare the records in a buffer using the OrderMaker that has been provided.

   a. **CompareRecords::CompareRecords(OrderMaker* orderMaker):**

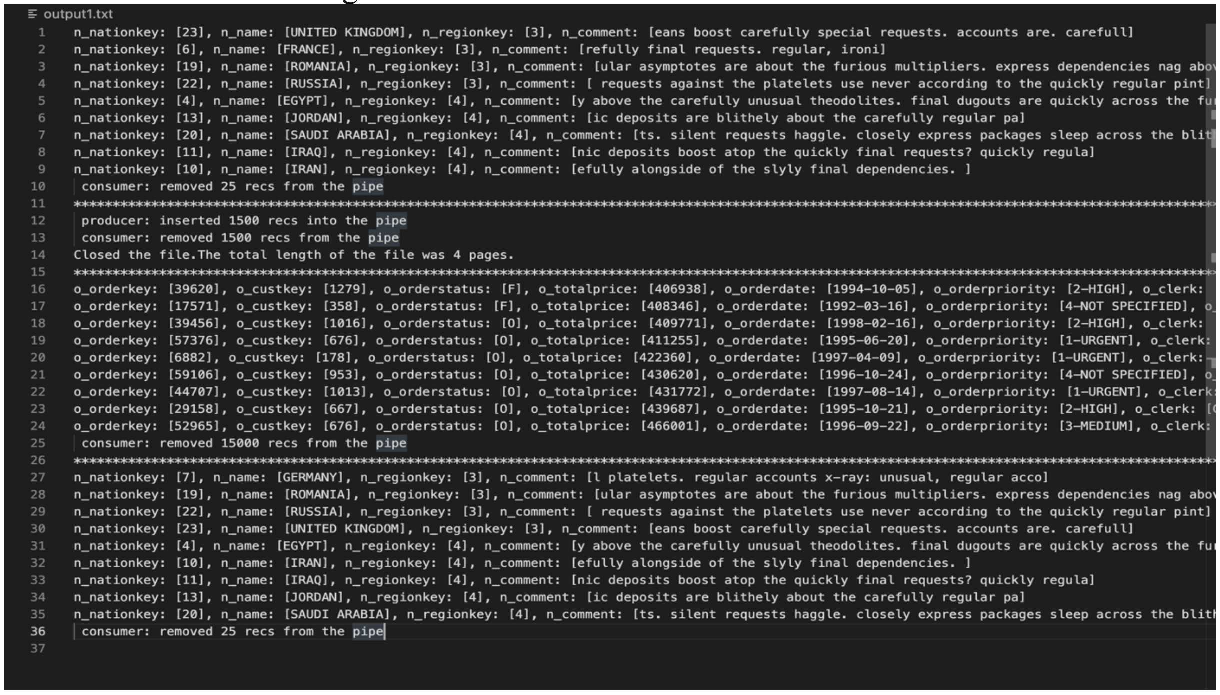
      This is a constructor for compareBuffers which takes orderMaker generated for the given CNF, as a parameter

   b. **bool CompareRecords::operator () (TPMMS* l, TPMMS* r)**

      This is a default comparison operator that will compare two records using the OrderMaker and sort them.

TEST RESULTS:

This is the result of running the ./runTestCases.sh file.



output1.txt

```
n_nationkey: [23], n_name: [UNITED KINGDOM], n_regionkey: [3], n_comment: [eans boost carefully special requests. accounts are. carefull]
n_nationkey: [6], n_name: [FRANCE], n_regionkey: [3], n_comment: [refully final requests. regular, ironi]
n_nationkey: [19], n_name: [ROMANIA], n_regionkey: [3], n_comment: [ular asymptotes are about the furious multipliers. express dependencies nag abo
n_nationkey: [22], n_name: [RUSSIA], n_regionkey: [3], n_comment: [ requests against the platelets use never according to the quickly regular pint]
n_nationkey: [4], n_name: [EGYPT], n_regionkey: [4], n_comment: [y above the carefully unusual theodolites. final dugouts are quickly across the fu
n_nationkey: [13], n_name: [JORDAN], n_regionkey: [4], n_comment: [ic deposits are blithely about the carefully regular pa]
n_nationkey: [20], n_name: [SAUDI ARABIA], n_regionkey: [4], n_comment: [ts. silent requests haggle. closely express packages sleep across the blit
n_nationkey: [11], n_name: [IRAQ], n_regionkey: [4], n_comment: [nic deposits boost atop the quickly final requests? quickly regula]
n_nationkey: [10], n_name: [IRAN], n_regionkey: [4], n_comment: [efully alongside of the slyly final dependencies. ]
 consumer: removed 25 recs from the pipe
****************************************************************************************************************************************************
 producer: inserted 1500 recs into the pipe
 consumer: removed 1500 recs from the pipe
Closed the file.The total length of the file was 4 pages.
****************************************************************************************************************************************************
o_orderkey: [39620], o_custkey: [1279], o_orderstatus: [F], o_totalprice: [406938], o_orderdate: [1994-10-05], o_orderpriority: [2-HIGH], o_clerk:
o_orderkey: [17571], o_custkey: [358], o_orderstatus: [F], o_totalprice: [408346], o_orderdate: [1992-03-16], o_orderpriority: [4-NOT SPECIFIED], o_
o_orderkey: [39456], o_custkey: [1016], o_orderstatus: [O], o_totalprice: [409771], o_orderdate: [1998-02-16], o_orderpriority: [2-HIGH], o_clerk:
o_orderkey: [57376], o_custkey: [676], o_orderstatus: [O], o_totalprice: [411255], o_orderdate: [1995-06-20], o_orderpriority: [1-URGENT], o_clerk:
o_orderkey: [6882], o_custkey: [178], o_orderstatus: [O], o_totalprice: [422360], o_orderdate: [1997-04-09], o_orderpriority: [1-URGENT], o_clerk:
o_orderkey: [59106], o_custkey: [953], o_orderstatus: [O], o_totalprice: [430620], o_orderdate: [1996-10-24], o_orderpriority: [4-NOT SPECIFIED], o_
o_orderkey: [44707], o_custkey: [1013], o_orderstatus: [O], o_totalprice: [431772], o_orderdate: [1997-08-14], o_orderpriority: [1-URGENT], o_clerk
o_orderkey: [29158], o_custkey: [667], o_orderstatus: [O], o_totalprice: [439687], o_orderdate: [1995-10-21], o_orderpriority: [2-HIGH], o_clerk: [
o_orderkey: [52965], o_custkey: [676], o_orderstatus: [O], o_totalprice: [466001], o_orderdate: [1996-09-22], o_orderpriority: [3-MEDIUM], o_clerk:
 consumer: removed 15000 recs from the pipe
****************************************************************************************************************************************************
n_nationkey: [7], n_name: [GERMANY], n_regionkey: [3], n_comment: [l platelets. regular accounts x-ray: unusual, regular acco]
n_nationkey: [19], n_name: [ROMANIA], n_regionkey: [3], n_comment: [ular asymptotes are about the furious multipliers. express dependencies nag abo
n_nationkey: [22], n_name: [RUSSIA], n_regionkey: [3], n_comment: [ requests against the platelets use never according to the quickly regular pint]
n_nationkey: [23], n_name: [UNITED KINGDOM], n_regionkey: [3], n_comment: [eans boost carefully special requests. accounts are. carefull]
n_nationkey: [4], n_name: [EGYPT], n_regionkey: [4], n_comment: [y above the carefully unusual theodolites. final dugouts are quickly across the fu
n_nationkey: [10], n_name: [IRAN], n_regionkey: [4], n_comment: [efully alongside of the slyly final dependencies. ]
n_nationkey: [11], n_name: [IRAQ], n_regionkey: [4], n_comment: [nic deposits boost atop the quickly final requests? quickly regula]
n_nationkey: [13], n_name: [JORDAN], n_regionkey: [4], n_comment: [ic deposits are blithely about the carefully regular pa]
n_nationkey: [20], n_name: [SAUDI ARABIA], n_regionkey: [4], n_comment: [ts. silent requests haggle. closely express packages sleep across the blit
 consumer: removed 25 recs from the pipe
```

GTest Results:

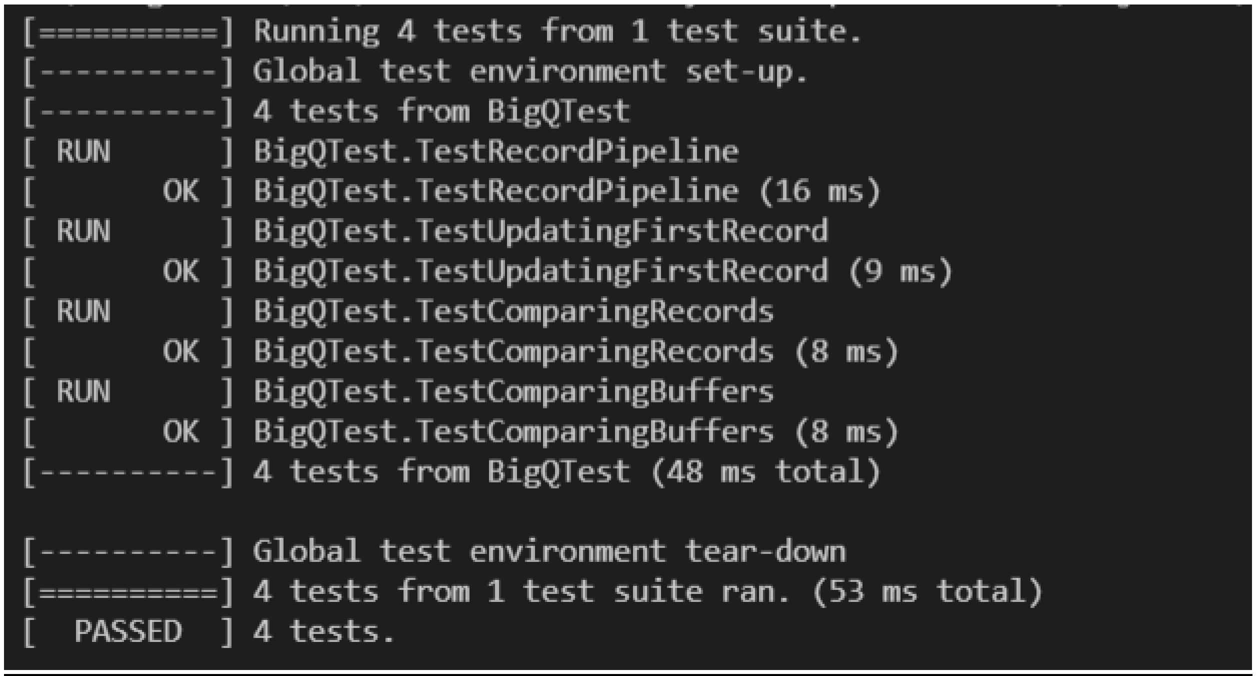
We also ran Google Unit Tests for our code. The tests include

- TestRecordPipeline – This test tests if records are properly being inserted into the pipeline and they are fetched as expected.
- TestUpdatingFirstRecord– This test the first record in the buffer can be updated.
- TestComparingRecords– This tests if the comparison between the records is working as expected.
- TestComparingBuffers– This tests if the comparison between records in different buffers is working as expected.

To run the gtests just type run the following commands

1. make gTests.o
2. make gTests
3. ./gTests.out

The Results of the tests are as follows:



```
[==========] Running 4 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 4 tests from BigQTest
[ RUN      ] BigQTest.TestRecordPipeline
[       OK ] BigQTest.TestRecordPipeline (16 ms)
[ RUN      ] BigQTest.TestUpdatingFirstRecord
[       OK ] BigQTest.TestUpdatingFirstRecord (9 ms)
[ RUN      ] BigQTest.TestComparingRecords
[       OK ] BigQTest.TestComparingRecords (8 ms)
[ RUN      ] BigQTest.TestComparingBuffers
[       OK ] BigQTest.TestComparingBuffers (8 ms)
[----------] 4 tests from BigQTest (48 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 1 test suite ran. (53 ms total)
[  PASSED  ] 4 tests.
```