COP6726 – Database System Implementation

Project 4 – Part One Report

Team members:

1. Akshay Ganapathy (UFID: 3684-6922)
2. Riyaz Shaik (UFID: 4360-0170)

Steps to execute project code:

The code was executed using **Windows Subsystem for Linux (Ubuntu 18.04)** environment.

1. The project source code is in the folder "a4-1test". Its contents include the following:
    a. Code needed to run test.cc.
    b. All ".bin" files should be included in the "bin" folder.
    c. Google test framework has been added in the directory "googletest". This framework is needed to run gtest unit test cases.
    d. All the gtest unit test cases have been added in "test" folder. StatisticsTest.cc contains the unit test cases for methods in Statistics class, respectively. Make sure to include the respective ".bin" and ".meta files in the directory, "bin".
    e. Makefile to compile test.cc and gtest unit test cases.
2. To run test.cc, from project root execute command "make a4-1.out" and then execute with test case number i.e., "./a4-1.out [0-11]".
3. To run gtest unit test cases, from project root execute command "make gtests" and then execute "./gtests".

The following are the methods with their description:

Method	Description
void Statistics :: AddRel (char *relName, int numTuples);	Adds a new relation, specifying its name and the number of tuples.
void Statistics :: AddAtt (char *relName, char *attrName, int numDistincts);	Adds a new attribute for a relation, specifying its name and the number of distinct values for that attribute.
void Statistics::CopyRel(char *oldName, char *newName);	Produces a copy of the relation (including all its attributes and all of its statistics) and stores it under the new name.
void Statistics::Read(char *fromWhere);	Read data from file and stores in object.
void Statistics::Write(char *fromWhere);	Writes object data to file
void Statistics::Apply(struct AndList *parseTree, char *relNames[], int numToJoin);	The Apply operation uses the statistics stored by the Statistics class to simulate

	a join of all of the relations listed in the relNames parameter.
double Statistics::Estimate(struct AndList *parseTree, char **relNames, int numToJoin);	It computes the number of tuples that would result from a join over the relations in relNames, and returns this to the caller
std::ostream &operator<<(std::ostream &os, const Rel &relation);	Inserts the number of attributes of the relation and the relation size in the output stream
std::istream &operator>>(std::istream &is, Rel &Rel);	Reads the number of attributes in the relation from the input stream
std::ostream &operator<<(std::ostream &os, const Statistics &stat);	Adds the size of the relation present inside Statistics as well as the attribute names and values to the output stream followed by the estimate.
std::istream &operator>>(std::istream &is, Statistics &stat);	Reads all the relations present in the file and stores them in the statistics object.

The following is the format used for Statistics.txt:

1. The number in the first line defines the number of relations in the query.

2. The relation name is defined in the second line.

3. The third line defines the total number of tuples contained in that relation.

(If there are multiple relations being used, the different relations are listed in the same order one after the other:(number of relations, relation name, number of tuples).

4. The next line states the number of attributes in consideration.

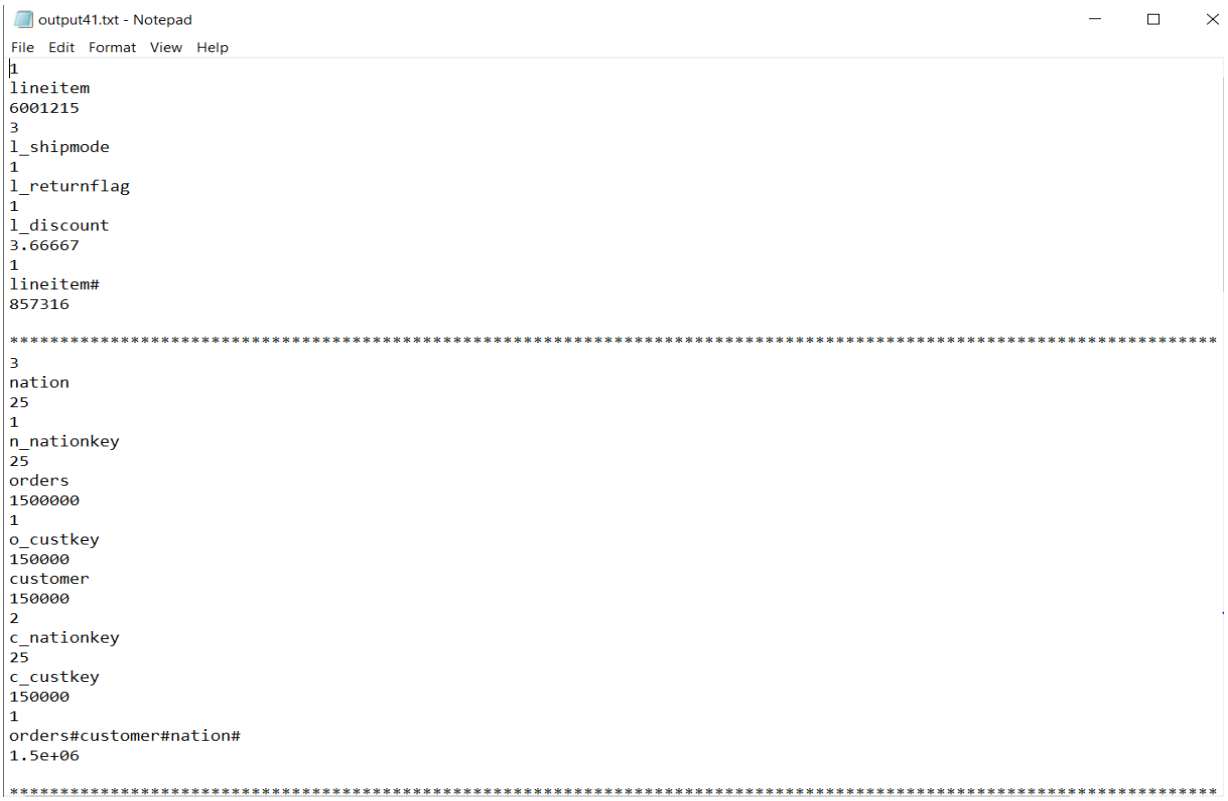
5. We then define the attribute names and their distinct values for each attributes one after the other.

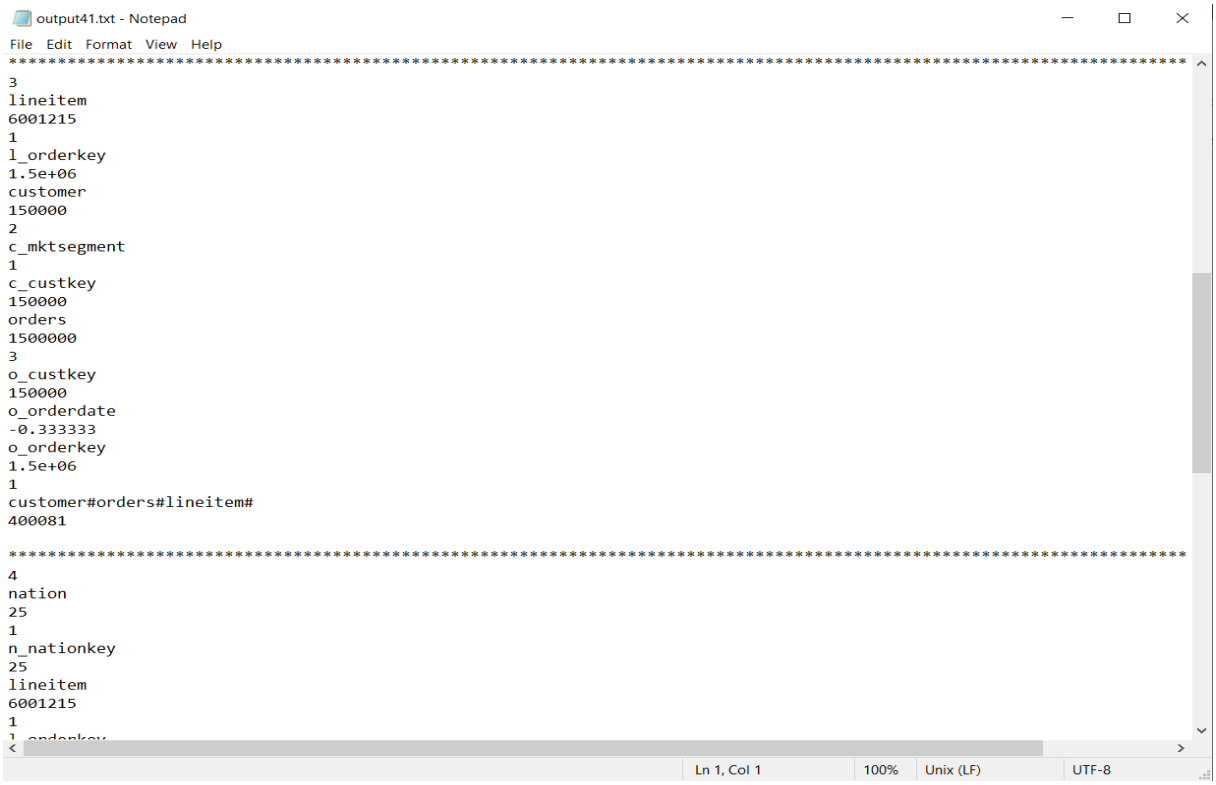
6. The next line contains the names of all the relations involved in the query.

7. The number of tuples in the resulting relation.

The following are the screenshots of the output41.txt:

The following results are obtained by using 1 GB TPCH data generated from tpch-dbgen.





output41.txt - Notepad

File Edit Format View Help

```
400081

********************************************************************************************************************************
4
nation
25
1
n_nationkey
25
lineitem
6001215
1
l_orderkey
1.5e+06
customer
150000
2
c_nationkey
25
c_custkey
150000
orders
1500000
3
o_custkey
150000
o_orderdate
-0.333333
o_orderkey
1.5e+06
1
customer#orders#lineitem#nation#
2.0004e+06

********************************************************************************************************************************
2
lineitem
```

output41.txt - Notepad

File Edit Format View Help

```
orders
1500000
3
o_custkey
150000
o_orderdate
-0.333333
o_orderkey
1.5e+06
1
customer#orders#lineitem#nation#
2.0004e+06

********************************************************************************************************************************
2
lineitem
6001215
3
l_shipmode
2
l_partkey
200000
l_shipinstruct
1
part
200000
2
p_container
2
p_partkey
200000
1
part#lineitem#
21432.9

********************************************************************************************************************************
```

The following are the results of gtest unit test cases:

Result: All the unit test cases passed

StatisticsTest.cc contains the unit test cases for methods in Statistics class. To run gtest unit test cases, from project root execute command “make gtests” and then execute “./gtests”. Make sure to include the respective “.bin” and “.meta” files in the directory, “bin”.

```
akshayg1996@LAPTOP-ACV24OQ1:/mnt/d/Gitlab/dbi/Project 4.1$ ./gtests
[==========] Running 12 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 12 tests from STATISTICSTEST
[ RUN      ] STATISTICSTEST.READ_WRITE_TEST
Read and Write Test Started!
Read and Write Test Ended
[       OK ] STATISTICSTEST.READ_WRITE_TEST (7 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_1
Correctness Test 1 Started!
Correctness Test 1 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_1 (8 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_2
Correctness Test 2 Started!
Correctness Test 2 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_2 (6 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_3
Correctness Test 3 Started!
Correctness Test 3 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_3 (11 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_4
Correctness Test 4 Started!
n.n_regionkey not found!!
Correctness Test 4 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_4 (4 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_5
Correctness Test 5 Started!
Correctness Test 5 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_5 (7 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_6
```

```
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_6
Correctness Test 6 Started!
Correctness Test 6 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_6 (6 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_7
Correctness Test 7 Started!
Correctness Test 7 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_7 (3 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_8
Correctness Test 8 Started!
Correctness Test 8 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_8 (5 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_9
Correctness Test 9 Started!
Correctness Test 9 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_9 (9 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_10
Correctness Test 10 Started!
Correctness Test 10 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_10 (10 ms)
[ RUN      ] STATISTICSTEST.CORRECTNESS_TEST_11
Correctness Test 11 Started!
Correctness Test 11 Ended!
[       OK ] STATISTICSTEST.CORRECTNESS_TEST_11 (5 ms)
[----------] 12 tests from STATISTICSTEST (81 ms total)

[----------] Global test environment tear-down
[==========] 12 tests from 1 test suite ran. (82 ms total)
[  PASSED  ] 12 tests.
akshayg1996@LAPTOP-ACV24OQ1:/mnt/d/Gitlab/dbi/Project 4.1$
```