Project Report – Statistical Estimation

Arpan Banerjee

UFID: 9359-9083

[arpanbanerjee@ufl.edu](mailto:arpanbanerjee@ufl.edu)

Krutantak Patil

UFID: 5615-6343

Krutantakb.patil@ufl.edu

**Steps to compile and execute:**

1. **make ./a2test.out**
   1. Builds all the essential files to generate the required bin files.
2. **./a2test.out**
   1. Generates required bin files and stores in the directory specified in test.cat.
3. **make ./a4-1.out**
   1. Builds all the essential files required to execute the porgram.
4. **./a4-1.out [0 - 11]**
   1. Executes the program and runs the query number specified as an argument.
   2. Ensure that a2test.out is executed and all the essential bin files a generate beforehand.
5. **./runTestCases.sh**
   1. Executes the program and runs all the queries mentioned in the runTesCases.sh script.
   2. Output is written to output41.txt file.
6. **make ./gtest.out**
   1. Builds all the essential files required to run the google gtest file
7. **./gtest.out**
   1. Executes the google test cases written in gtest.cc file.
   2. Ensure that a2test.out is executed and all the essential bin files a generate beforehand.

**Description of the functions implemented for Statistical Estimation:**

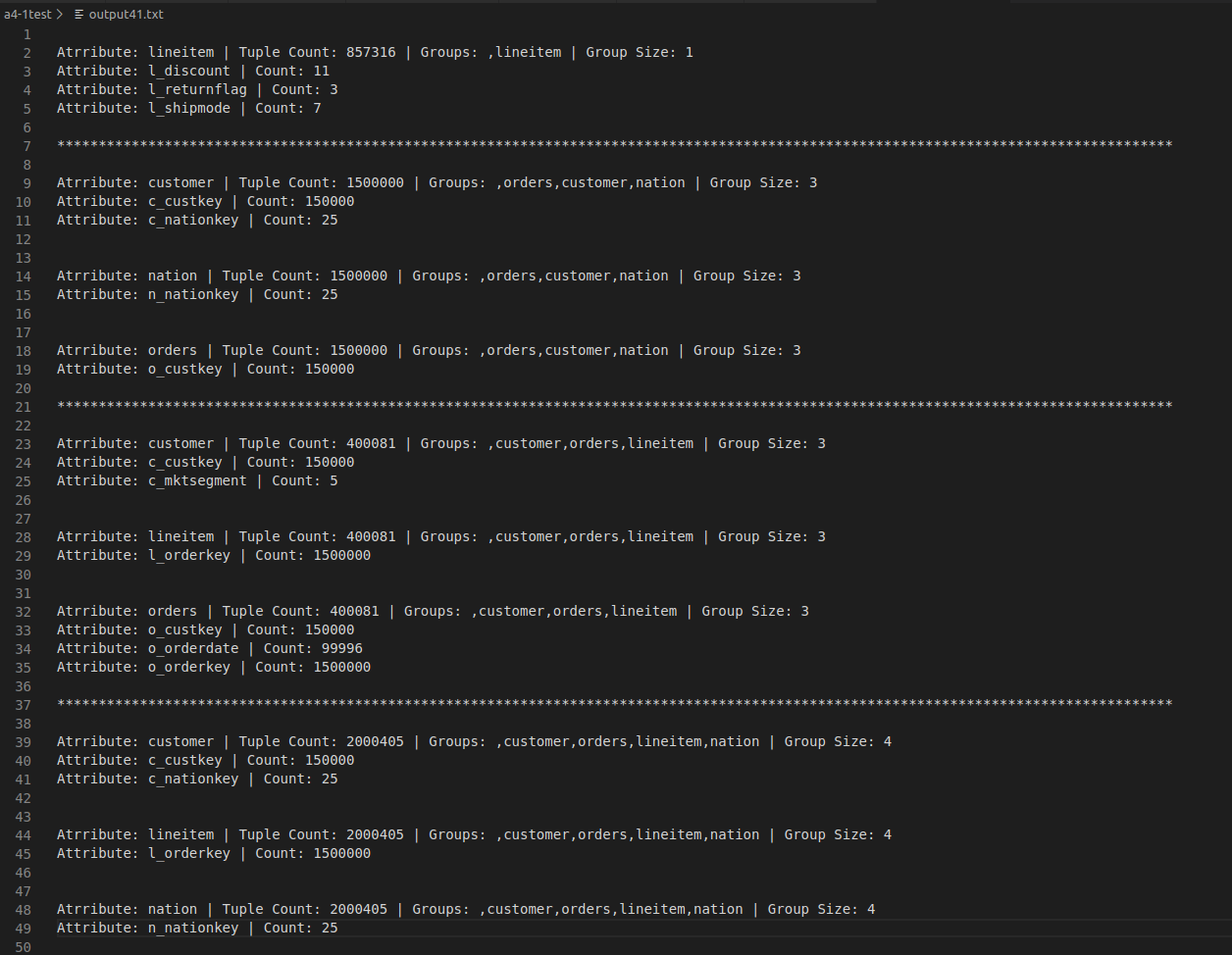
(functions are declared in Statistics.h and implemented in Statistics.cc files)

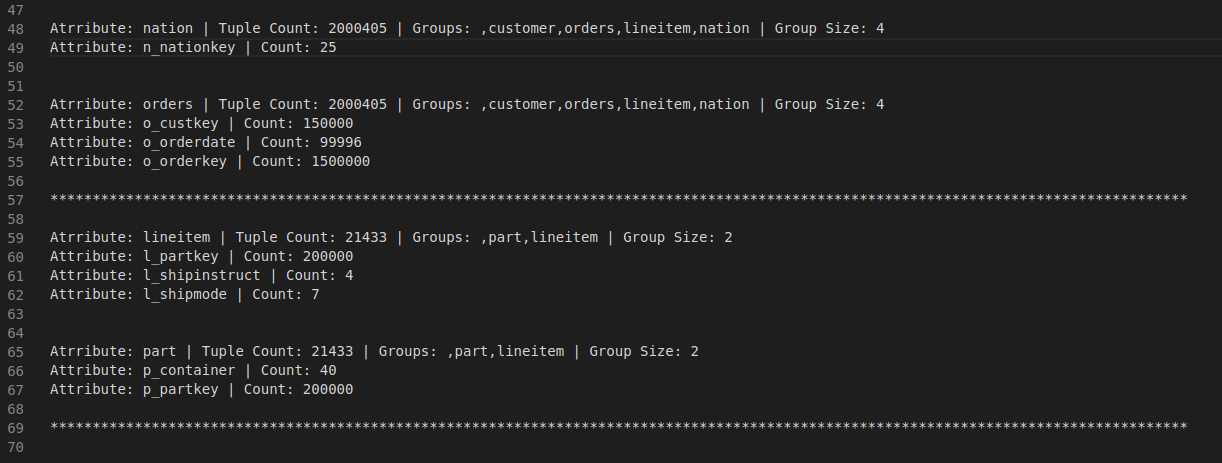
1. **Statistics::Statistics()**
   1. Default constructor of Statistics.cc.
2. **Statistics::Statistics(Statistics &copyMe)**
   1. Parametrized constructor of Statistics.cc
   2. Copies RelationStatistics map from the give object copyMe to processing object RelationalStatistics map.
   3. Iterators through each element, instantiates RelationStatistics object and then writes it to processing objects RelationStatistics map.
3. **Statistics::~Statistics()**
   1. Default destructor of Statistics.cc.
   2. Iterators through the processing objects RelationStatistics map to free memory space.
4. **void Statistics::AddRel(char \*relName, int numTuples)**
   1. Adds specified relation statistics of the relation entry, if found any, in objects RelationStatistics Map and updates the tuple count against it.
   2. If existing relation entry is not found, it creates a new RelationStatistics object s and adds itself to the map.
5. **void Statistics::AddAtt(char \*relName, char \*attName, int numDistincts)**
   1. Retrieves a relation from the objects RelationStatistics map with specified relName and adds mentioned attribute name and distinct count to the relation table.
6. **void Statistics::Apply(struct AndList \*parseTree, char \*relNames[], int numToJoin)**
   1. Estimates values for the given query for each of its relation and updates it RelationStatistics map.
7. **void Statistics::CopyRel(char \*oldName, char \*newName)**
   1. Copies content of old relation into the new relation.
   2. First, it check whether the specified old relation and new relation are not same.
   3. Then, if they are different, new relation and old relation are retrieved using the specified names
   4. Finally it updates/adds it to the RelationStatistic Map.
8. **double Statistics::Estimate(struct AndList \*parseTree, char \*\*relNames, int numToJoin)**
   1. Initially the give relation attributes are checked using GivesError method and program is terminated with an error message if they are invalid.
   2. Selectivity is evaluated using Assess method for each of the relation nodes in the parsetree.
   3. In the end all the selectivity values are multiplied with the estimated realtion records.
9. **void Statistics::Read(char \*fromWhere)**
   1. Opens the fromWhere file and then reads the given specified file and adds it to the objects RelationStatistics map after encountering “BEGIN” until “END”.
   2. It maps the data attributes to relName, rowCount, groupNameArr and groupNo and addes it to respective relations.
   3. This continues till the we reach end of file or the file is read completely.
10. **void Statistics::Write(char \*fromWhere)**
    1. Writes the output to fromWhere file.
    2. Iteratively it reads the object’s RelationStatistics map and each relation data is written to the file as per following format –
       1. Attribute: <name>, Tuple Count: <count>, Groups: <relation names who are part of the query predicate>, Group Size: <count of distinct group length>.
       2. Next lines are the displayed as – Attribute: <name of relation attributes used in query predicate>, Count: <distinct number of values in the respective relation attribute>
    3. Between each relation entry, it appends a blank line (“\n”) for better readability.

***Description of the functions implemented for Statistical Estimation:***

1. **execute command ./runTestCases.sh for 1GB data.**

(output41.txt snapshot)



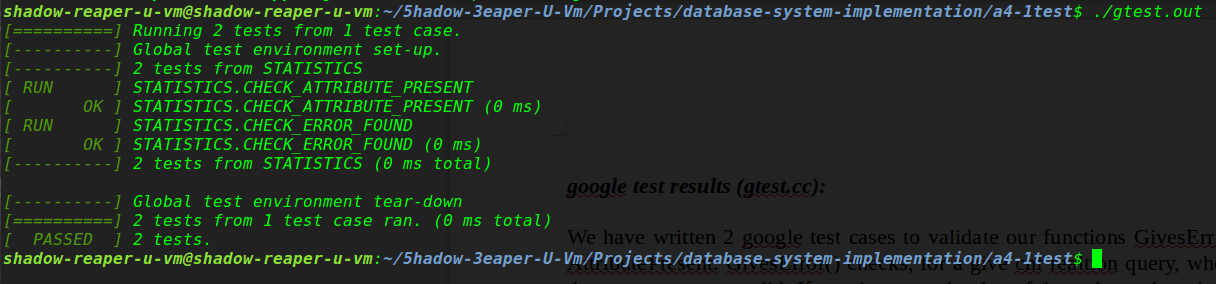


***google test results (gtest.cc):***

We have written 2 google test cases to validate our functions GivesError() and AttributePresent.

1. **CHECK\_ATTRIBUTE\_PRESENT**
   1. In this test case we are checking for give “lineitem” relation cnf attributes values using AttributePresent() method.
   2. AttributePresent() method will return true if value of attribute in the given query is present in the specified relation table.
   3. Otherwise, it will return false.
   4. AttributePresent() method will check all the attributes in specified query predicate.
2. **CHECK\_ERROR\_FOUND**
   1. In this test case we are checking for give “lineitem” relation cnf attributes values as well as operands are valid using GivesError() method.
   2. GivesError() method will validate predicate attributes using AttributePresent() method along with code of each operand specifed in the predicate.return true if value of attribute in the given query is present in the specified relation table.
   3. If any of the operand code does not match or the attribute is not in the specified relation, it will return false. Otherwise, it will return true.

**gtest.cc output snapshot –**



***Bugs encountered:***

1. Correction in q3 query – “s\_nationey” to “s\_nationkey”.
2. Correction in q4 query – invalid/incorrect relation parameters are properly updated.
3. Correction in q5 query – for relation “orders” “o\_orderdate” attribute added with numTuples = 99996.
4. Correction in q7 query – for relation “lineitem” “i\_receiptdate” attribute added with numTuples = 198455.
5. Correction in q10 query – “yyparse()” line is added at the correct places. Also, for relation “orders” “o\_orderdate” attribute added with numTuples = 99996
6. Correction in q11 query – “p\_conatiner” to “p\_container”.