Database Implementation System
Project: 4.2

Group Members: Sathya Sai Ram Kumar (34891238) & Mohit Kalra (13906151)

Instruction to Compile and Run:

To run a4-2.out:

> make clean

> make a4-2.out

> ./a4-2.out

To run runTestCases42.sh:

Please use the below commands to run the bash file. This will generate an output42.txt file.

> make clean

> make a4-2.out

> ./runTestCases42.sh

To run GTest:

> make clean

> make gtest

>./gtest < tc6.sql

Project Flow & Structure

QueryTreeNode.h

This file is used to define the basic structure of the execution tree node. The following are the nodes defined.

Base Node: The class **RelOpNode** is defined as a base node from which all other nodes inherit a common structure to implement. Following are the children nodes of this base node:

Join Node : The class JoinOpNode represent the join node.

Project Node : The class ProjectOpNode represent the join node.

SelectFile Node: The class SelectFileOpNode represent the join node.

SelectPipe Node: The class SelectPipeOpNode represent the join node.

Distinct Node: The class DistinctOpNode represent the join node.

GroupBy Node: The class GroupByOpNode represent the join node.

WriteOut Node: The class WriteOutOpNode represent the join node.

QueryPlanner.h

This file is used to define the query planner class which takes in the SQL statement and parses it for an execution tree for the different RelOp they can be used to process the statement.

```
class QueryPlanner{
        public:
                // attributes
                vector<char *> tableNames;
                vector<char *> joinOrder;
                vector<char *> buffer;
                AliaseMap aliaseMap;
                SchemaMap map;
                Statistics s;
                RelOpNode *root;
                // functions
                QueryPlanner();
                void PopulateSchemaMap ();
                void PopulateStatistics ();
                void PopulateAliasMapAndCopyStatistics () ;
                void CopyNameList(NameList *nameList, vector<string> &names) ;
                void Compile();
                void Optimise();
                void BuildExecutionTree();
                void Print();
};
```

Function Descriptions

1. **QueryPlanner();**

   The constructor function helps boot up the class object and populates the necessary variables necessary for query parsing and optimization plan. The constructor first calls the yyparse () function which populates the extern data structure variables used throughout the program. Post the parsing, it populates the string to schema map and attribute statistics for the tables present in the catalog.

2. **void PopulateSchemaMap ();**

   This function is called when the constructor is invoked by creation of a new QueryPlanner Object. This function populates the map *SchemaMap* variable of the class. Each of the string names of schemas are mapped to their corresponding Schema Objects.

3. **void PopulateStatistics ();**

2. The *Statistics* 's' object is populated by this method. As the previous method, this method is invoked by the QueryPlanner constructor. The AddRel method is invoked to populate the statistics for each of the attributes pertaining to all the relations of the catalog.

1. **void PopulateAliasMap ();**

   Alias map holds the map from the actual table to the alias name being used in the query. After the query is parsed, all the aliases are stored in this table for uniformity in table names.

2. **void CopyNameList(NameList *nameList, vector<string> &names);**

   This is a utility function utilized for converting a NameList data structure to a vector of strings. The vector of strings is passed by reference and each name is pushed until the nameList is completely read.

3. **void Compile();**

   This function marks the starting point for the entire logic. This function calls the Optimise and BuildExecutionTree which comprises of the main logic of the entire assignment.

4. **void Optimise();**

   Optimise function uses the estimate function from assignment 4.1 to permute the best ordering of joins given more than 1 join. Join conditions are first extracted from AndList of **WHERE** condition. Permutations of these joins are then run over a for loop for computing the minimum join cost, the join ordering of which is stored in a variable to be used further during node creation.

5. **void BuildExecutionTree();**

   This function constructs the query plan tree from the information processed until now. Query nodes correspond to 7 relational operator implementations from assignment 3. Starting from the root node, each of different types of nodes like SelectFile, SelectPipe etc. are constructed and then appended to the corresponding parent pointer. Unique Pipe IDs are assigned, and updated schema is passed down to the child node by the parent node.

6. **void Print();**

   Print function prints the tree branch with the minimum cost. Structures like CNF, Schema are printed along other necessary information.

Results

Screenshots from output42.txt

Query 1

```
TC1
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 1
 Output Schema:
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
 Select CNF:
 ( n.n_name = UNITED STATES)
-----------------------------------
-----------------------------------
 PROJECT OPERATION
 Input Pipe ID : 1
 Output Pipe ID 2
 Number Attrs Input : 4
 Number Attrs Output : 1
 Attrs To Keep : [0]
 Output Schema:
        Att n.n_nationkey : Int
-----------------------------------
**********************************************************************************************************************************************
```

Query 2

```
TC2
6: syntax error at SELECT
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 1
 Output Schema:
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
 Select CNF:
 ( n.n_nationkey > 5)
-----------------------------------
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 3
 Output Schema:
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Select CNF:
-----------------------------------
-----------------------------------
 JOIN OPERATION
 Left Input Pipe ID : 1
 Right Input Pipe ID : 3
 Output Pipe ID : 2
 Output Schema :
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Join CNF :
 ( n.n_regionkey = r.r_regionkey)
-----------------------------------
-----------------------------------
 PROJECT OPERATION
 Input Pipe ID : 2
 Output Pipe ID 4
 Number Attrs Input : 7
 Number Attrs Output : 1
 Attrs To Keep : [1]
 Output Schema:
        Att n.n_name : String
-----------------------------------
************************************************************************************************************************************************************
```

Query 3

```
TC3
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 1
 Output Schema:
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
 Select CNF:
 ( n.n_name = UNITED STATES)
-----------------------------------
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 3
 Output Schema:
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Select CNF:
-----------------------------------
-----------------------------------
 JOIN OPERATION
 Left Input Pipe ID : 1
 Right Input Pipe ID : 3
 Output Pipe ID : 2
 Output Schema :
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Join CNF :
 ( n.n_regionkey = r.r_regionkey)
-----------------------------------
-----------------------------------
 SUM OPERATION
 Input Pipe ID : 2
 Output Pipe ID : 4
 Function :
 (n.n_nationkey)
 Output Schema:
        Att sum : Int
-----------------------------------
**********************************************************************************************************************************
```

Query 4

```
TC4
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 1
 Output Schema:
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
 Select CNF:
 ( n.n_name = UNITED STATES)
-----------------------------------
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 3
 Output Schema:
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Select CNF:
-----------------------------------
-----------------------------------
 JOIN OPERATION
 Left Input Pipe ID : 1
 Right Input Pipe ID : 3
 Output Pipe ID : 2
 Output Schema :
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Join CNF :
 ( n.n_regionkey = r.r_regionkey)
-----------------------------------
-----------------------------------
 GROUP OPERATION
 Input Pipe ID : 2
 Output Pipe ID : 4
 Output Schema :
        Att sum : Int
        Att n.n_regionkey : Int
 Function :
 (n.n_regionkey)
 Grouping On :
        NumAtts =  1
        n.n_regionkey : Int
-----------------------------------
***********************************************************************************************************************************
```

Query 5

```
TC5
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 1
 Output Schema:
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
 Select CNF:
 ( n.n_nationkey > 10)
-----------------------------------
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 3
 Output Schema:
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Select CNF:
-----------------------------------
-----------------------------------
 JOIN OPERATION
 Left Input Pipe ID : 1
 Right Input Pipe ID : 3
 Output Pipe ID : 2
 Output Schema :
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
 Join CNF :
 ( n.n_regionkey = r.r_regionkey)
-----------------------------------
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 4
 Output Schema:
        Att c.c_custkey : Int
        Att c.c_name : String
        Att c.c_address : String
        Att c.c_nationkey : Int
        Att c.c_phone : String
        Att c.c_acctbal : Double
        Att c.c_mktsegment : String
        Att c.c_comment : String
 Select CNF:
-----------------------------------
```

```
-----------------------------------
 SELECT FILE OPERATION
 Select File Operation
 Output Pipe ID 4
 Output Schema:
        Att c.c_custkey : Int
        Att c.c_name : String
        Att c.c_address : String
        Att c.c_nationkey : Int
        Att c.c_phone : String
        Att c.c_acctbal : Double
        Att c.c_mktsegment : String
        Att c.c_comment : String
 Select CNF:
-----------------------------------
-----------------------------------
 JOIN OPERATION
 Left Input Pipe ID : 2
 Right Input Pipe ID : 4
 Output Pipe ID : 5
 Output Schema :
        Att n.n_nationkey : Int
        Att n.n_name : String
        Att n.n_regionkey : Int
        Att n.n_comment : String
        Att r.r_regionkey : Int
        Att r.r_name : String
        Att r.r_comment : String
        Att c.c_custkey : Int
        Att c.c_name : String
        Att c.c_address : String
        Att c.c_nationkey : Int
        Att c.c_phone : String
        Att c.c_acctbal : Double
        Att c.c_mktsegment : String
        Att c.c_comment : String
 Join CNF :
 ( n.n_nationkey = c.c_nationkey)
-----------------------------------
-----------------------------------
 GROUP OPERATION
 Input Pipe ID : 5
 Output Pipe ID : 6
 Output Schema :
        Att sum : Int
        Att r.r_regionkey : Int
 Function :
 ( (n.n_nationkey + r.r_regionkey))
 Distinct Function :  True
 Grouping On :
        NumAtts =  1
        r.r_regionkey : Int
-----------------------------------
***************************************************************************************************************************************************
```

Gtest Results

```
(base) mk@mk:~/Documents/uf_docs/sem_2/Database_Implementation/workspace/DBI4.2$ ./gtest < tc6.sql
[==========] Running 2 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 2 tests from QueryTesting
[ RUN      ] QueryTesting.CheckQueryAlias
[       OK ] QueryTesting.CheckQueryAlias (0 ms)
[ RUN      ] QueryTesting.TestJoinOptimization
[       OK ] QueryTesting.TestJoinOptimization (1 ms)
[----------] 2 tests from QueryTesting (1 ms total)

[----------] Global test environment tear-down
[==========] 2 tests from 1 test suite ran. (1 ms total)                [  PASSED  ] 2 tests.
(base) mk@mk:~/Documents/uf_docs/sem_2/Database_Implementation/workspace/DBI4.2$
```