Database System Implementation – Project 5
(Putting it All Together)

Team Members:
Ananda Bhaasita Desiraju
(UFID: 4081-1191)
Nidhi Shashi Sharma
(UFID: 6843-1215)

Instructions to compile and run the code:

- To compile the code: make
- To run all the tests in main.cc: ./test.out followed by each sql query
- To create a new table: ./test.out followed by the (example) query:
  CREATE TABLE table_name (att1 INTEGER, att2 DOUBLE, att3 STRING) AS HEAP;
- To insert date into a table: ./test.out followed by the (example) query:
  INSERT 'file_name' INTO table_name;
- To drop a table: ./test.out followed by the (example) query:
  DROP TABLE table_name;
- To compile the tests in GTest.cc: make gtest.out
- To run all the GTests in GTest.cc: ./gtest.out

Goal

The goal of this project is to put together all of the pieces from the previous projects and have a completely working system that can be used to run any SQL queries along with creating new tables, inserting data into the tables and deleting the tables as well.

Parser

Parser takes a simple SQL statement, processes it, and puts the result into a set of data structures. The linked lists and flags that are constructed by the parser are declared in the file parser.y are:

- struct FuncOperator *finalFunction;
- struct TableList *tables;

- struct AndList *boolean;
- struct NameList *groupingAtts;
- struct NameList *attsToSelect; int distinctAtts;
- int distinctFunc;
- int distinctAtts;
- struct InOutPipe *io;
- string createTableName;
- string createTableType;
- vector<string> atts;
- vector<string> attTypes;
- int queryType;

These data structures hold the result of the parsing. Extern declaration is used in order to access these data structures externally from the parser.

QueryOutputGenerator

The QueryOutputGenerator class was written in the previous part for query compilation and query optimization. The following functions were added for the final integration of the project.

Functions

void RunInOrder():

This function is used to run the RUN function in accordance with the binary tree order.

void Run();

Based on the type of the node, that is, SELECTPIPE, SELECTFILE, PROJECT, JOIN, DUPLICATEREMOVAL, SUM, GROUPBY or WRITEOUT, the query plan is run accordingly

void WaitUntilDone();

This function is used to stall the execution of the worker thread until the execution of the run function completes.

QueryPlan

QueryPlan class is used to generate all the possible query plans. The following new functions are added in query plan to help integrate the final project with the part 4.2 of the project

Functions

int clear_pipe (Pipe &in_pipe, Schema *schema, bool print):

This function is used to empty the contents of the input pipe.

int clear_pipe (Pipe &in_pipe, Schema *schema, Function &func, bool print):

This is the overloaded method of the above function

string getTableName(string tblAttr):

This function is used to fetch the name of the table from the given attribute.

Main

In the main.cc file we have implemented functions that call the required functions for the execution of a query, or creation/deletion of a table or insertion of data in the table.

Functions:

void ReadFileLocations():

This function is used to read the path for the catalog directory, the tpch directory and the bin directory from the test.cat file

void GetLoadPath(char *loadPath, char *table, char *prefix, char *extension):

This function is used to fetch the path from which the data of the given table is to be loaded.

void RemoveTableFromCatalog(string table):

This function is used to remove the table mentioned in the DROP TABLE query and all its contents as well.

void createTable():

This function is supposed to create a new table having the name mentioned in the CREATE TABLE query.

main():

In this function we execute the appropriate action based on the type of query mentioned. The types of queries executed are the SELECT query, CREATE TABLE query, INSERT INTO query, DROP TABLE query, SET Query, etc.

GTESTS

- **Test Case 1**
  CreateTest: Test to check if the table is properly created.

- **Test Case 2**
  InsertTest: Test to verify if values were properly inserted in the table.

SCREENSHOTS

GTests