DB Implementation Project 5

Contributors :

Megha Nagarmunoli UFID: 6768-3882
Siddharth Jain UFID: 7185-1161

Steps to execute:

1. Unzip the file *SiddharthJain_MeghaNagarmunoli_p5.zip*
2. cd *SiddharthJain_MeghaNagarmunoli_p5/*

Note: *SiddharthJain_MeghaNagarmunoli_p5*
All necessary bin files, meta files, Statistics.txt and source code files are in the same folder.

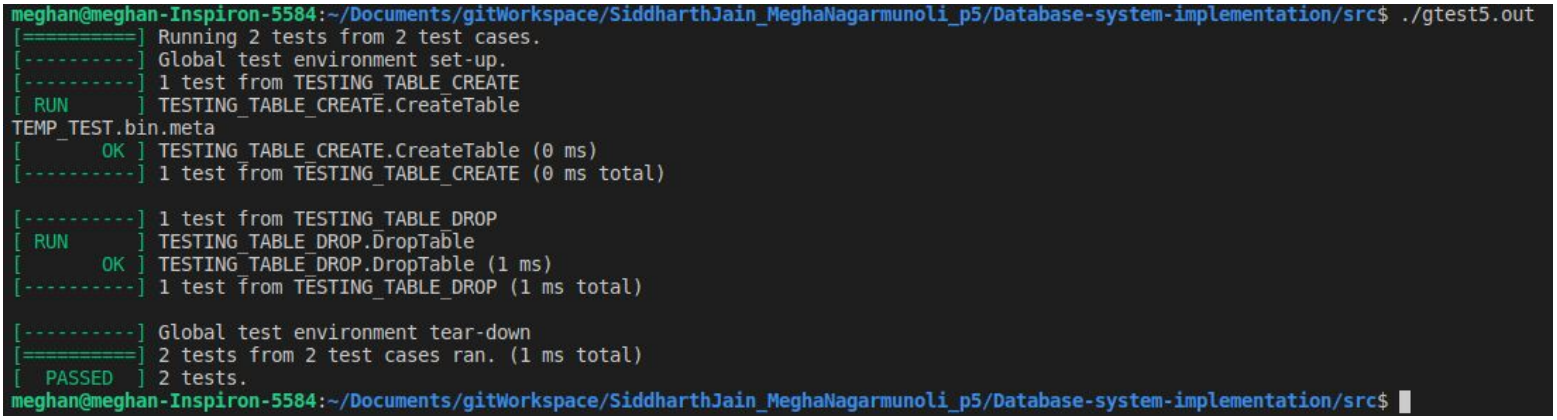
How to compile test.cc

1. make a5.out

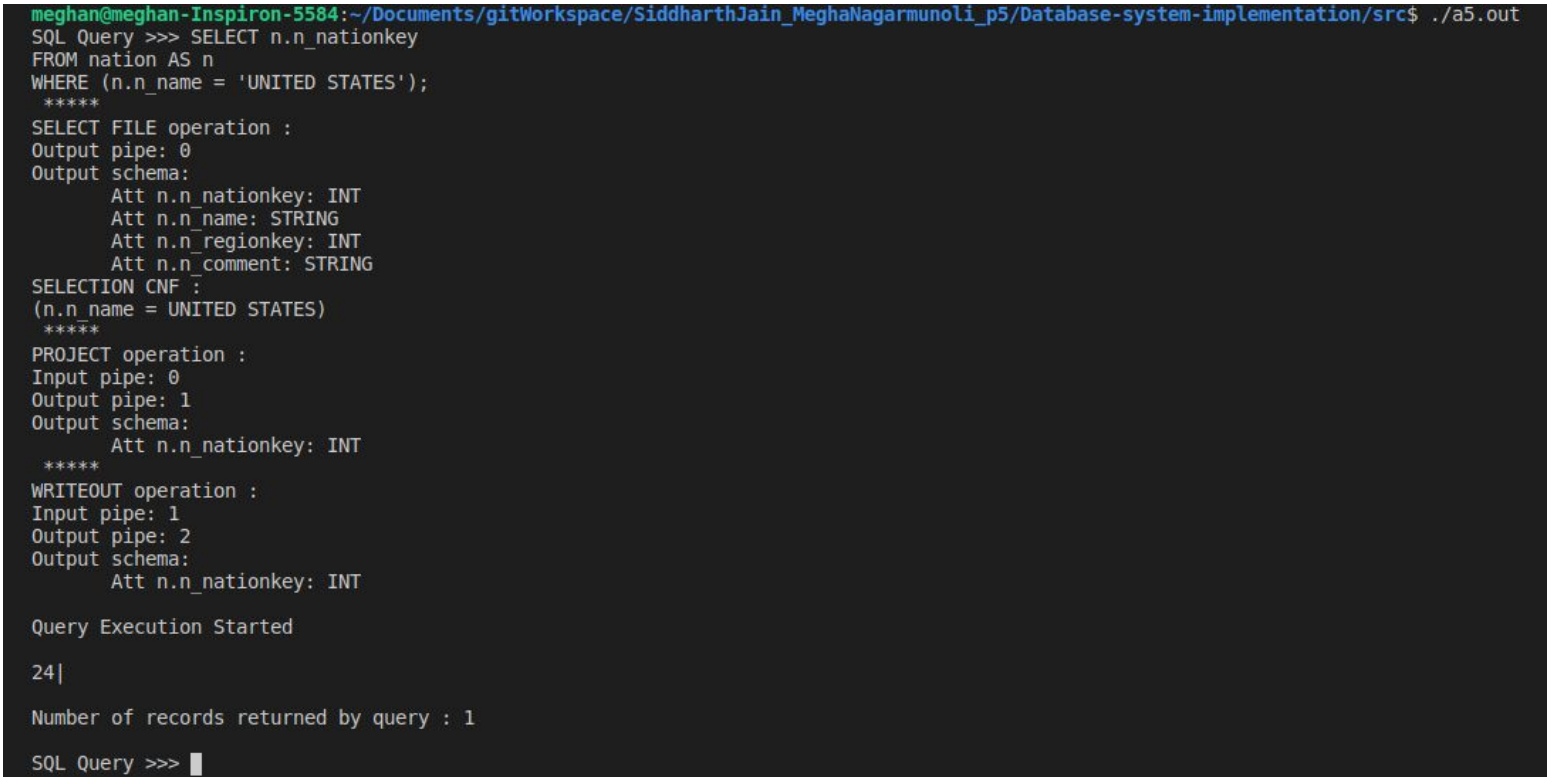
To run the gTests:

1. make gtest5.out
2. ./gtest5.out

Screenshot for GTest:



Screenshot for Query 1:



The other queries can be run. This is just provided as an example.

YouTube Link:

https://youtu.be/bgBZTwCNreI

Code Implementation:

Below are the classes and functions that were implemented to achieve the final goal of the Database Implementation project.

Class SqlQueryInterpreter: This class is used to interpret the DDL statements and execute them accordingly.
Method:
SqlQueryInterpreter::run():
This function runs the yyparse() which is used to interpret the commands as per requirement. The commands to be matched or interpreted are added to Parser.y

Example:
CREATE TABLE Name ‘(‘ NewAtts ‘)’ AS HEAP ‘;’
{
	newAttrs = $5
	newTable = $3

}
This populates the external variables used in the Interpreter.
These values are used to make a decision on what is to be executed by the DDL class.
For queries, it runs the plan, print and execute functions of the QueryPlanDriver.

SqlQueryInterpreter::clear(): This function clears all the variables used by the Interpreter.

class TableDriver:

TableDriver::createTable()
This function enters the schema in the catalog file. It creates and updates the metafile. Once done, it calls create on the DBFile which creates the .bin file.

TableDriver::insertInto()
This function loads the tbl file into the .bin file.

TableDriver::dropTable()
This function deletes the bin file and meta file for the table and removes the table details from the catalog

TableDriver::setOutput()
This function sets the output location to stdout or given filename

In the QueryPlanDriver.cc

void QueryPlanDriver::execute()
This function creates Pipes and calls the execute on the root node of the query plan.

void LeafNode::execute(Pipe** pipes, RelationalOp** relops)
This function opens the file and outputs the records to a pipe.

void ProjectNode::execute(Pipe** pipes, RelationalOp** relops)
This function projects the expected attributes from input pipe to output pipe.

void DedupNode::execute(Pipe** pipes, RelationalOp** relops)
This function puts distinct records to the output pipe.

void SumNode::execute(Pipe** pipes, RelationalOp** relops)
This function sums the values in the input pipe.

void JoinNode::execute(Pipe** pipes, RelationalOp** relops)
This function joins two relations based on the CNF. It takes two pipes (pleft and pright) as inputs, joins them and gives it to the output pipe.

void WriteNode::execute(Pipe** pipes, RelationalOp** relops)
This function writes the results to an output stream.