# Project 4 (part 2): Query Compilation and Optimization

## Group Members:


1. Smrati Pandey, UFID: 4323-9459
2. Wins Goyal, UFID: 7357-1559


## Instructions for Execution / Code compilation and to run tests:

### I. To extract folders

a) Extract the contents of the folder, named *SmratiPandey_WinsGoyal_p42.zip*.
b) Open the terminal.
c) Navigate to the extracted folder.

### II. To update "*test.cat*"

a) In *test.cat* file, update: *catalog_path*
b) This should be updated in the first line.

### III. To run the tests

Perform following these commands:

```
$ make clean
$ make a42.out
$ ./a42.out              ## Enter the Query, then push ctrl-D
$ ./a42.out< {fileName}  ## The file feeds the Query to it.
$ ./runTestCases42.sh
```

Last command will generate the Query plan for files *tc1.sql, tc2.sql, tc3.sql, tc4.sql* and *tc5.sql*. These Query plans are saved in *"**output42.txt**"* file afterwards.

## Summarized Documentation of the implemented methods and functions:

We are required to print the *Optimized Query Plan* for a processed *SQL* query, that will also use *Statistics* implemented in the last assignment. The main files where the code is used, written and implemented are *ParseTree.h*, *PrintParseTree.h, PrintParseTree.cc* and *main.cc*

A *QueryPlan* is basically a *Tree* and it's *nodes* represent the *QueryPlanNode*. Also, as mentioned in the assignment, there shall be different types of these nodes, each subclass corresponding to the respective relational operators that were all implemented in *Assignment 3*. These nodes could be *'Select File (SF)', 'Select Pipe (SP)', 'Project (P)', 'Group By (GB)', 'Sum (S)', 'Distinct (D)', 'Join (J)'* and *'Write (W)'*

The following 'Flow Pipeline' shows a brief implementation of the methods corresponding to different *QueryNodes* being called in the *Main* method in main.cc:

1. First, create a 'Select File' node for every relation or table calling the class '*SelectFileNode*'. Segregate Select & Join operations from the list given.
2. Then, create 'Select Pipe' node for every selected table using the class '*SelectPipeNode*'. As we want to minimize the number of generating intermediate tuples, compute the *Optimal Order of Joins*.
3. Create 'Join Node' for what is directed by the computed 'Optimal Order of Joins', using the class '*JoinNode*'.

4. Now, an order of Precedence follows for upcoming relational operations to be performed. That is,
   a) If there is an attribute for 'Group By' → Use class '*GroupByNode*' to create 'Group By' node.
   b) If there is any aggregate present in the attributes → Use class '*SumNode*' to create 'Sum' node. The aggregate function is nested in the *else* condition of 'Group By'.
   c) If Query asked for 'Projection', Use class '*ProjectNode*' to create 'Project' node. Or use '*NewProjNode*'.
   d) If Query asked for 'Distinct' tuples in the result, Use class '*DistinctNode*' to create 'Distinct' node.

## **Screenshots of Running All Test Case Results:** (from *output41.txt*)

I. **TC1:**

output42.txt

```
TC1
enter:
Number of selects: 1
Number of joins: 0

IN ORDER TRAVERSAL

 ***********
SELECT FILE
Input Pipe 0
Output Pipe 1
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String

 ***********
SELECT PIPE
Input Pipe 1
Output Pipe 2
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String
SELECTION CNF :
 ((n.n_name = UNITED STATES))
 ***********
PROJECT
Input Pipe 2
Output Pipe 3
Output Schema:
    Att n.n_nationkey : Int
****************************************
```

II. **TC2:**

output42.txt

```
TC2
enter:
Number of selects: 1
Number of joins: 1

IN ORDER TRAVERSAL

 ***********
SELECT FILE
Input Pipe 0
Output Pipe 1
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String

 ***********
SELECT PIPE
Input Pipe 1
Output Pipe 2
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String
SELECTION CNF :
 ((n.n_nationkey > 5))
 ***********
SELECT FILE
Input Pipe 0
Output Pipe 3
Output Schema:
    Att r.r_regionkey : Int
    Att r.r_name : String
    Att r.r_comment : String

 ***********
 ***********
JOIN
Left Input Pipe 2
Right Input Pipe 3
Output Pipe 4
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
CNF:
(n.n_regionkey = r.r_regionkey)

 ***********
PROJECT
Input Pipe 4
Output Pipe 5
Output Schema:
    Att n.n_name : String
**********
```

III. **TC3:**

output42.txt

```
TC3
enter:
Number of selects: 1
Number of joins: 1

IN ORDER TRAVERSAL

 **********
SELECT FILE
Input Pipe 0
Output Pipe 1
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String

 **********
SELECT PIPE
Input Pipe 1
Output Pipe 2
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String
SELECTION CNF :
 ((n.n_name = UNITED STATES))
 **********
SELECT FILE
Input Pipe 0
Output Pipe 3
Output Schema:
    Att r.r_regionkey : Int
    Att r.r_name : String
    Att r.r_comment : String

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

```

 **********
JOIN
Left Input Pipe 2
Right Input Pipe 3
Output Pipe 4
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
CNF:
(n.n_regionkey = r.r_regionkey)

 **********
SUM
Input Pipe ID : 4
Output Pipe ID : 5
Output Schema:
    Att sum : Double
Function :
(n.n_nationkey)
distinctFunc: 0
**********
```

IV. **TC4:**

output42.txt

```
TC4
enter:
Number of selects: 1
Number of joins: 1
GROUPING ON n.n_regionkey
    Att n.n_regionkey: String
IN ORDER TRAVERSAL

 ***********
SELECT FILE
Input Pipe 0
Output Pipe 1
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String

 ***********
SELECT PIPE
Input Pipe 1
Output Pipe 2
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String
SELECTION CNF :
 ((n.n_name = UNITED STATES))
 ***********
SELECT FILE
Input Pipe 0
Output Pipe 3
Output Schema:
    Att r.r_regionkey : Int
    Att r.r_name : String
    Att r.r_comment : String

```

output42.txt

```

 ***********
JOIN
Left Input Pipe 2
Right Input Pipe 3
Output Pipe 4
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
CNF:
(n.n_regionkey = r.r_regionkey)

 ***********
GROUP BY
Left Input Pipe 4
Output Pipe 5
Output Schema:
    Att sum : Double
    Att n.n_regionkey : Int
OrderMaker :
NumAtts =     1
  0:     2 Int
Function :
(n.n_regionkey)
distinctFunc: 0

 ***********
PROJECT
Input Pipe 5
Output Pipe 6
Output Schema:
    Att sum : Double
**********
```

V. **TC5:**

output42.txt

```
TC5
enter:
Number of selects: 1
Number of joins: 2
GROUPING ON r.r_regionkey
    Att r.r_regionkey: String
IN ORDER TRAVERSAL

***********
SELECT FILE
Input Pipe 0
Output Pipe 1
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String

***********
SELECT PIPE
Input Pipe 1
Output Pipe 2
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
    Att n.n_regionkey : Int
    Att n.n_comment : String
SELECTION CNF :
((n.n_nationkey > 10))
***********
SELECT FILE
Input Pipe 0
Output Pipe 3
Output Schema:
    Att r.r_regionkey : Int
    Att r.r_name : String
    Att r.r_comment : String

```

output42.txt

```
***********
JOIN
Left Input Pipe 2
Right Input Pipe 3
Output Pipe 4
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
CNF:
(n.n_regionkey = r.r_regionkey)

***********
SELECT FILE
Input Pipe 0
Output Pipe 5
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

***********
JOIN
Left Input Pipe 4
Right Input Pipe 5
Output Pipe 6
Output Schema:
    Att n.n_nationkey : Int
    Att n.n_name : String
```

output42.txt

```
        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
CNF:
(n.n_nationkey = c.c_nationkey)

 ***********
GROUP BY
Left Input Pipe 6
Output Pipe 7
Output Schema:
        Att sum : Double
        Att r.r_regionkey : Int
OrderMaker :
NumAtts =     1
  0:     4 Int
Function :
((n.n_nationkey) + (r.r_regionkey))
distinctFunc: 1

 ***********
PROJECT
Input Pipe 7
Output Pipe 8
Output Schema:
        Att sum : Double
****************************************
```

## GTests:

- Perform following commands to execute the GTests:

```
$ make clean
$ make gTest.out
$ ./gTest.out
```

a) *QueryOptimizationTest for Join Node:→* ***JoinNode***
This tests whether the Join Order is generated if 'And' is passed. Boolean expression handles this in the structure. The result is compared with the expected outcome.

b) *QueryOptimizationTest for Project Node:→* ***ProjectNode***
This tests whether the if the Projection is required, the node is generated in Query Plan or not. This is done by passing the 'Comparison Pointed' in the structure. The result is compared with the expected outcome.

Following is the screenshot of these two *Gtest-cases* in execution (terminal):

```
g++ -O2 -Wno-deprecated -g -c gTest.cc
g++ -O2 -Wno-deprecated -o gtest.out Record.o Comparison.o ComparisonEngine.o
.o  gtest.o -l pthread -lgtest
[==========] Running 2 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 2 tests from QueryOptimizationTest
[ RUN      ] QueryOptimizationTest.JoinNode
([       OK ] QueryOptimizationTest.JoinNode (0 ms)
[ RUN      ] QueryOptimizationTest.ProjectNode
 unknown code 97184248[       OK ] QueryOptimizationTest.ProjectNode (0 ms)
[----------] 2 tests from QueryOptimizationTest (0 ms total)

[----------] Global test environment tear-down
[==========] 2 tests from 1 test case ran. (0 ms total)
[  PASSED  ] 2 tests.
```