COP-6726 DATABASE SYSTEM IMPLEMENTATION

Project-3.0 Report

Group Member Names:

K Jaswanth Reddy - (UFID: 22719671)
M Dheeraj Kumar - (UFID: 82690922)

Running the Code:

1. Source code is in the folder “a3test”. It contains all the tbl and respective bin files for all tables.
2. Unit Testing is done and are presented in gtest.cc
3. To run main.cc, from project root execute command “make main” and then execute “./main”
4. Google test framework has been added in the directory “googletest”. This framework is needed to run gtest unit test cases.
5. All the gtest unit test cases have been added in the “test” folder. To run test.cc, from project root execute command “make test.out” and then execute “./test.out [1-5]”

Steps to run Tests(test.cc):

- Execute the “**make test.out**” command from the project root. Then execute “**./test.out**”.

Steps to run gTests(gTest.cc):

- Execute the “**make gTest**” command from the project root. Then execute “**./gTest.out**”.

<u>The following are the methods and their descriptions:</u>

<u>RelationalOpThreadMemberHolder:</u>

To avoid the problem of creating duplicates for functions with similar parameters we create multiple instances for RelationalOpThreadMemberHolder all the threads implemented in the code are single parameters. This keeps the multiple arguments needed for passing in hold until used.

<u>SelectPipe:</u>

This function reads data sequentially from the input pipe and adds the record to the pipe when it finds a match to the CNFs in the database.

1. void SelectPipe::Run(Pipe &inputPipe, Pipe &outputPipe, CNF &selOp, Record &literal):
    a. This function is used to assign the class level variables and starts SelectPipe class worker thread.
2. void *SelectPipe::thread_work(void *args):
    a. This function is used to perform the scan of dbfile provided to it and for every tuple accepted by the CNF, it inserts the tuple into the output pipe.

<u>SelectFile:</u>

Instead of utilizing a pipe to read the data, this operation assigns the duty of identifying the matching CNF record to the DBfile. The records are obtained sequentially or utilizing binary search depending on whether DBfile is implemented as a heap or sorted file.

1. void SelectFile::Run(DBFile &inFile, Pipe &outputPipe, CNF &selOp, Record &literal):
    a. This function is used to assign the class level variables and starts SelectFile class worker thread.
2. void *SelectFile::thread_work(void *args);
    a. This function is used to insert the records that are matched according to the CNF expression from a dbfile into the output pipe.

<u>Project:</u>

This operation projects the data read from the inputPipe and pushes it into the outputPipe.

1. void Project::Run(Pipe &inputPipe, Pipe &outputPipe, int *keepMe, int numAttsInput, int numAttsOutput):
    a. This function is used to assign the class level variables and starts Project class worker thread.
2. void *Project::thread_work(void *args):
    a. This function is used to run the SQL and perform the projection operation and inserts only the attributes provided by the input from the tuples into the output pipe.

Join:

This operation uses the GetSortOrders function to validate the correctness of a CNF when it is given. SortMergeJoin or NestedLoopJoin are used for yes or no, depending on whether CNF is permitted or not. SortMergeJoin creates two BigQ instances with sorted input data from two inputPipes, and then compares the first record of both instances. The smaller head is removed and the outputPipe is added. This is done in a recursive manner until both BigQs are empty. In NestedLoopJoin, however, all of the records are first stored in the rightPipe, and then each record is compared to the record in the leftPipe. The records that match are then pushed to the outputPipe.

1. void Join::Run(Pipe &inputPipeL, Pipe &inputPipeR, Pipe &outputPipe, CNF &selOp, Record &literal):
    a. This function is used to assign the class level variables and starts Join class worker thread.
2. void *Join::thread_work(void *args):
    a. This function takes two input pipes and an output pipe along with the CNF and performs the join from the two pipes according to the CNF.

GroupBy:

This function adds all of the records together while removing duplicates from the pipe. The attribute is merged with the current record when the final sum is determined.

1. void GroupBy::Run(Pipe &inputPipe, Pipe &outputPipe, OrderMaker &groupAtts, Function &computeMe):
    a. This function is used to assign the class level variables and starts GroupBy class worker thread.
2. void *GroupBy::thread_work(void *args):
    a. It groups the data and then sends one sum for each group to the output pipe. The first attribute of every tuple sent to the output pipe is the sum, followed by the values for each of the grouping attributes as the remaining attributes.

WriteOut:

This procedure allows you to edit records by storing their attribute values in a pointer and then updating them with the Write method.

1. void WriteOut::Run(Pipe &inputPipe, FILE *outFile, Schema &mySchema):
    a. This function is used to assign the class level variables and starts WriteOut class worker thread.
2. void *WriteOut::thread_work(void *args):
    a. This function uses the schema for writing the text version of the output records to the respective file.
3. void WriteOut::WaitUntilDone()
    a. This function makes the class worker thread wait until all other threads are completed.

Sum:

Using the CalculateSum template function, this function calculates the sum of an int or a double. This CalculateSum function takes a record from the pipe and adds it to the next record in the pipeline. Depending on the function type called by sum, the final sum is subsequently saved in an attribute of type Int or Double.

1. void Sum::Run(Pipe &inputPipe, Pipe &outputPipe, Function &computeMe):
    a. This function is used to assign the class level variables and starts the Sum class worker thread.
2. void *Sum::thread_work(void *args):
    a. Sum executes the SUM SQL aggregate function to the input stream and returns a single tuple with the sum in the output pipe.

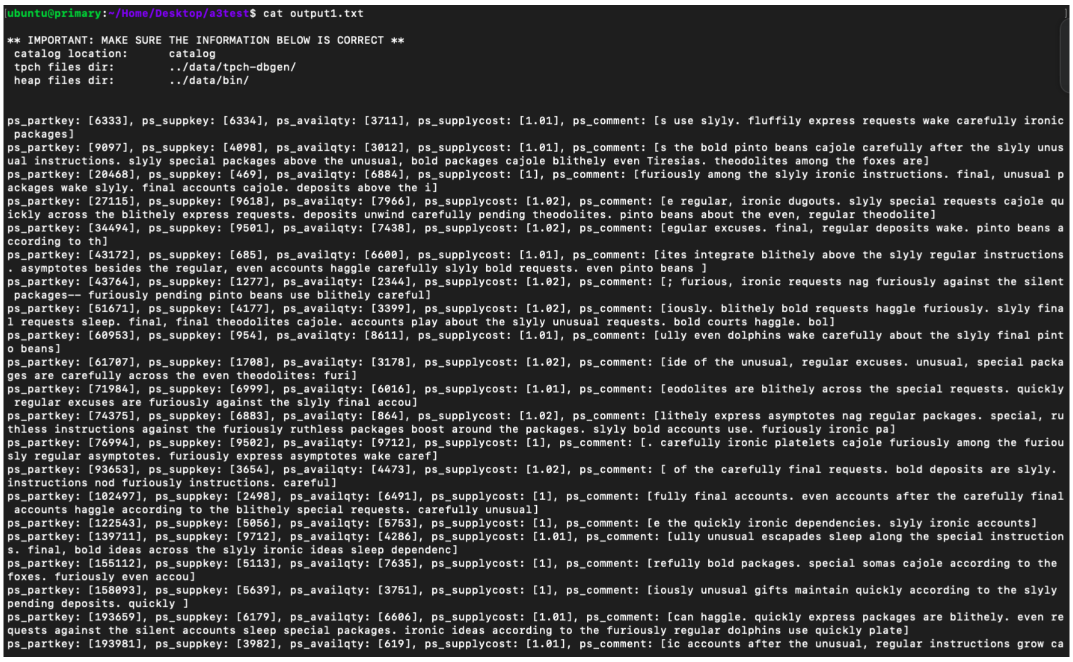
DuplicateRemoval:

When the records are sorted, all of a record's duplicates display next to each other. When matching CNF records are extracted, they are pushed into a BigQ, and each record is matched with adjacent records using this method. If a match is found, the duplicates are deleted and the adjacent is eliminated.

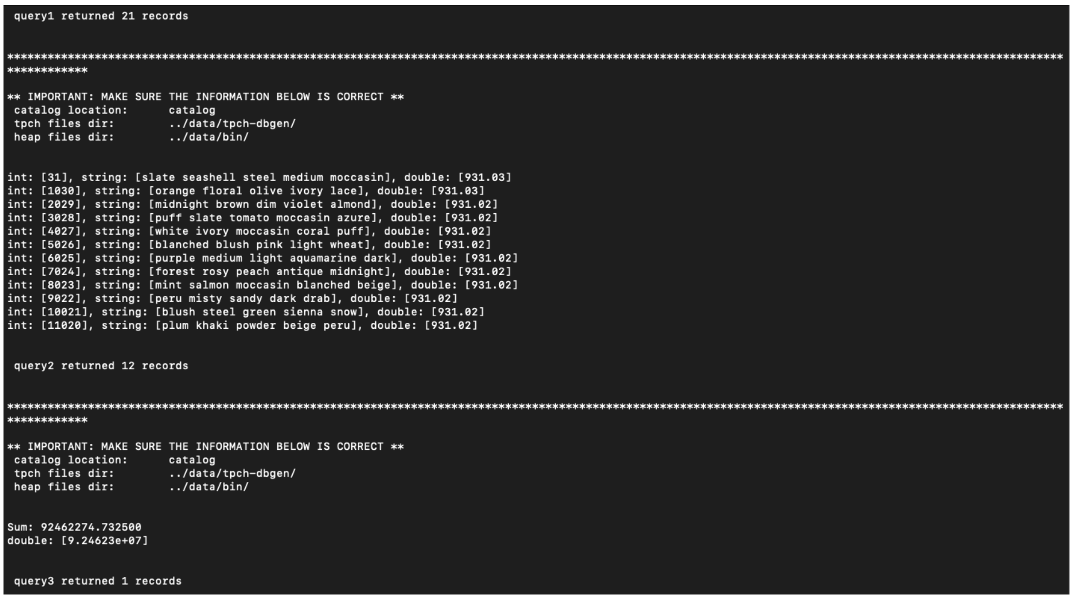
1. void DuplicateRemoval::Run(Pipe &inputPipe, Pipe &outputPipe, Schema &mySchema):
    a. This function is used to assign the class level variables and starts the DuplicateRemoval class worker thread.
2. void *DuplicateRemoval::thread_work(void *args):
    a. This function takes Input pipe, output pipe and schema coming from them and performs the removal of the duplicates in them.

Output:

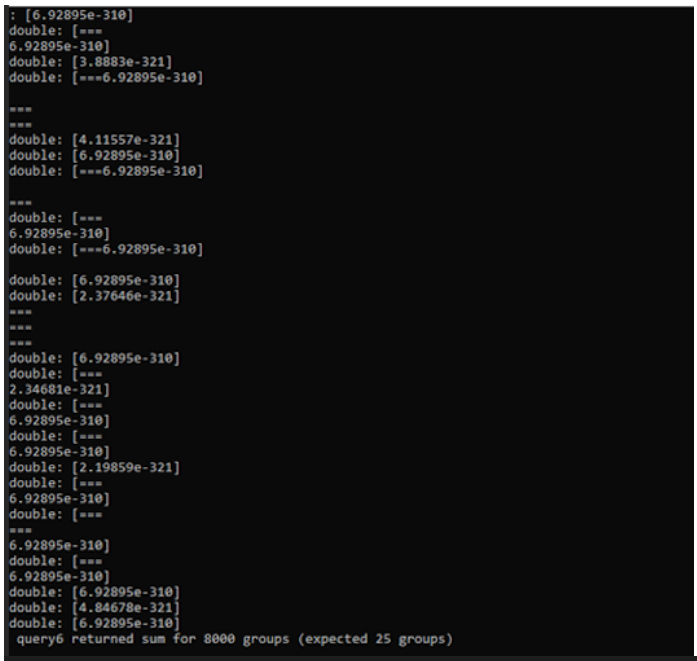
The following are the screenshots of results obtained by **using 1 GB TPCH data** generated from tpch-dbgen after executing all the test cases using the shell script “runTestCases.sh”



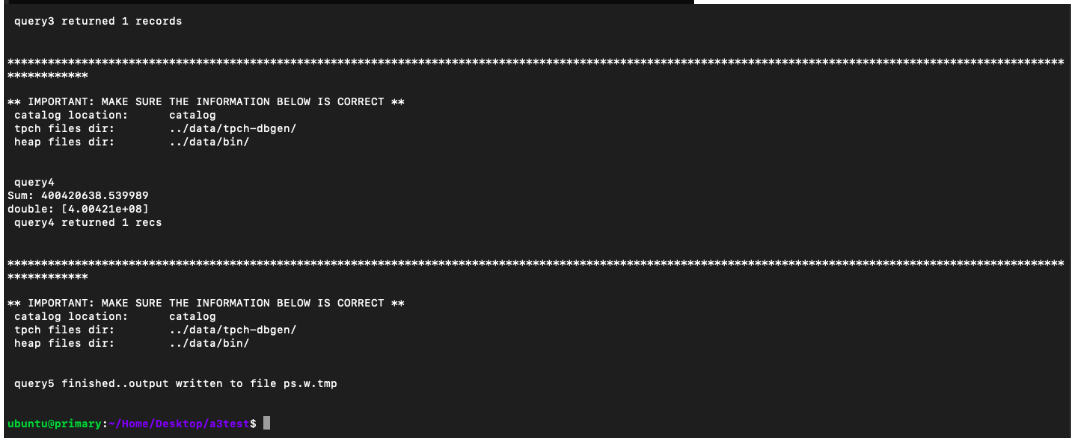
```
ubuntu@primary:~/Home/Desktop/a3test$ cat output1.txt

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      catalog
 tpch files dir:        ../data/tpch-dbgen/
 heap files dir:        ../data/bin/


ps_partkey: [6333], ps_suppkey: [6334], ps_availqty: [3711], ps_supplycost: [1.01], ps_comment: [s use slyly. fluffily express requests wake carefully ironic
 packages]
ps_partkey: [9097], ps_suppkey: [4098], ps_availqty: [3012], ps_supplycost: [1.01], ps_comment: [s the bold pinto beans cajole carefully after the slyly unus
ual instructions. slyly special packages above the unusual, bold packages cajole blithely even Tiresias. theodolites among the foxes are]
ps_partkey: [20468], ps_suppkey: [469], ps_availqty: [6884], ps_supplycost: [1], ps_comment: [furiously among the slyly ironic instructions. final, unusual p
ackages wake slyly. final accounts cajole. deposits above the i]
ps_partkey: [27115], ps_suppkey: [9618], ps_availqty: [7966], ps_supplycost: [1.02], ps_comment: [e regular, ironic dugouts. slyly special requests cajole qu
ickly across the blithely express requests. deposits unwind carefully pending theodolites. pinto beans about the even, regular theodolite]
ps_partkey: [34494], ps_suppkey: [9501], ps_availqty: [7438], ps_supplycost: [1.02], ps_comment: [egular excuses. final, regular deposits wake. pinto beans a
ccording to th]
ps_partkey: [43172], ps_suppkey: [685], ps_availqty: [6600], ps_supplycost: [1.01], ps_comment: [ites integrate blithely above the slyly regular instructions
. asymptotes besides the regular, even accounts haggle carefully slyly bold requests. even pinto beans ]
ps_partkey: [43764], ps_suppkey: [1277], ps_availqty: [2344], ps_supplycost: [1.02], ps_comment: [; furious, ironic requests nag furiously against the silent
 packages-- furiously pending pinto beans use blithely careful]
ps_partkey: [51671], ps_suppkey: [4177], ps_availqty: [3399], ps_supplycost: [1.02], ps_comment: [iously. blithely bold requests haggle furiously. slyly fina
l requests sleep. final, final theodolites cajole. accounts play about the slyly unusual requests. bold courts haggle. bol]
ps_partkey: [60953], ps_suppkey: [954], ps_availqty: [8611], ps_supplycost: [1.01], ps_comment: [ully even dolphins wake carefully about the slyly final pint
o beans]
ps_partkey: [61707], ps_suppkey: [1708], ps_availqty: [3178], ps_supplycost: [1.02], ps_comment: [ide of the unusual, regular excuses. unusual, special packa
ges are carefully across the even theodolites: furi]
ps_partkey: [71984], ps_suppkey: [6999], ps_availqty: [6016], ps_supplycost: [1.01], ps_comment: [eodolites are blithely across the special requests. quickly
 regular excuses are furiously against the slyly final accou]
ps_partkey: [74375], ps_suppkey: [6883], ps_availqty: [864], ps_supplycost: [1.02], ps_comment: [lithely express asymptotes nag regular packages. special, ru
thless instructions against the furiously ruthless packages boost around the packages. slyly bold accounts use. furiously ironic pa]
ps_partkey: [76994], ps_suppkey: [9502], ps_availqty: [9712], ps_supplycost: [1], ps_comment: [. carefully ironic platelets cajole furiously among the furiou
sly regular asymptotes. furiously express asymptotes wake caref]
ps_partkey: [93653], ps_suppkey: [3654], ps_availqty: [4473], ps_supplycost: [1.02], ps_comment: [ of the carefully final requests. bold deposits are slyly.
instructions nod furiously instructions. careful]
ps_partkey: [102497], ps_suppkey: [2498], ps_availqty: [6491], ps_supplycost: [1], ps_comment: [fully final accounts. even accounts after the carefully final
 accounts haggle according to the blithely special requests. carefully unusual]
ps_partkey: [122543], ps_suppkey: [5056], ps_availqty: [5753], ps_supplycost: [1], ps_comment: [e the quickly ironic dependencies. slyly ironic accounts]
ps_partkey: [139711], ps_suppkey: [9712], ps_availqty: [4286], ps_supplycost: [1.01], ps_comment: [ully unusual escapades sleep along the special instruction
s. final, bold ideas across the slyly ironic ideas sleep dependenc]
ps_partkey: [155112], ps_suppkey: [5113], ps_availqty: [7635], ps_supplycost: [1], ps_comment: [refully bold packages. special somas cajole according to the
foxes. furiously even accou]
ps_partkey: [158093], ps_suppkey: [5639], ps_availqty: [3751], ps_supplycost: [1], ps_comment: [iously unusual gifts maintain quickly according to the slyly
pending deposits. quickly ]
ps_partkey: [193659], ps_suppkey: [6179], ps_availqty: [6606], ps_supplycost: [1.01], ps_comment: [can haggle. quickly express packages are blithely. even re
quests against the silent accounts sleep special packages. ironic ideas according to the furiously regular dolphins use quickly plate]
ps_partkey: [193981], ps_suppkey: [3982], ps_availqty: [619], ps_supplycost: [1.01], ps_comment: [ic accounts after the unusual, regular instructions grow ca
```



```
 query1 returned 21 records


**************************************************************************************************************************************************************
************

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      catalog
 tpch files dir:        ../data/tpch-dbgen/
 heap files dir:        ../data/bin/


int: [31], string: [slate seashell steel medium moccasin], double: [931.03]
int: [1030], string: [orange floral olive ivory lace], double: [931.03]
int: [2029], string: [midnight brown dim violet almond], double: [931.02]
int: [3028], string: [puff slate tomato moccasin azure], double: [931.02]
int: [4027], string: [white ivory moccasin coral puff], double: [931.02]
int: [5026], string: [blanched blush pink light wheat], double: [931.02]
int: [6025], string: [purple medium light aquamarine dark], double: [931.02]
int: [7024], string: [forest rosy peach antique midnight], double: [931.02]
int: [8023], string: [mint salmon moccasin blanched beige], double: [931.02]
int: [9022], string: [peru misty sandy dark drab], double: [931.02]
int: [10021], string: [blush steel green sienna snow], double: [931.02]
int: [11020], string: [plum khaki powder beige peru], double: [931.02]


 query2 returned 12 records


**************************************************************************************************************************************************************
************

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      catalog
 tpch files dir:        ../data/tpch-dbgen/
 heap files dir:        ../data/bin/


Sum: 92462274.732500
double: [9.24623e+07]


 query3 returned 1 records
```



```
: [6.92895e-310]
double: [===
6.92895e-310]
double: [3.8883e-321]
double: [===6.92895e-310]

===
===
double: [4.11557e-321]
double: [6.92895e-310]
double: [===6.92895e-310]

===
double: [===
6.92895e-310]
double: [===6.92895e-310]

double: [6.92895e-310]
double: [2.37646e-321]
===
===
===
double: [6.92895e-310]
double: [===
2.34681e-321]
double: [===
6.92895e-310]
double: [===
6.92895e-310]
double: [2.19859e-321]
double: [===
6.92895e-310]
double: [===
===
6.92895e-310]
double: [===
6.92895e-310]
double: [6.92895e-310]
double: [4.84678e-321]
double: [6.92895e-310]
 query6 returned sum for 8000 groups (expected 25 groups)
```

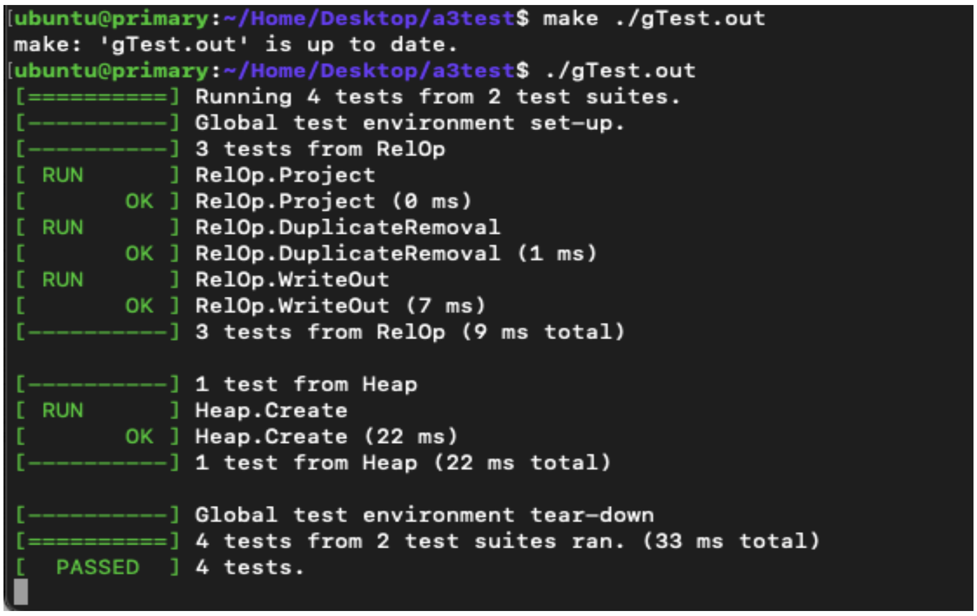


```
 query3 returned 1 records


**************************************************************************************************************************************************************
************

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      catalog
 tpch files dir:        ../data/tpch-dbgen/
 heap files dir:        ../data/bin/


 query4
Sum: 400420638.539989
double: [4.00421e+08]
 query4 returned 1 recs


**************************************************************************************************************************************************************
************

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      catalog
 tpch files dir:        ../data/tpch-dbgen/
 heap files dir:        ../data/bin/


 query5 finished..output written to file ps.w.tmp


ubuntu@primary:~/Home/Desktop/a3test$ 
```

GTEST RESULTS:

Execute the "**make gTests**" command from the project root. Then execute "**./gtest.out**".

Result: All the test cases are Passed.



```
[ubuntu@primary:~/Home/Desktop/a3test$ make ./gTest.out
make: 'gTest.out' is up to date.
[ubuntu@primary:~/Home/Desktop/a3test$ ./gTest.out
[==========] Running 4 tests from 2 test suites.
[----------] Global test environment set-up.
[----------] 3 tests from RelOp
[ RUN      ] RelOp.Project
[       OK ] RelOp.Project (0 ms)
[ RUN      ] RelOp.DuplicateRemoval
[       OK ] RelOp.DuplicateRemoval (1 ms)
[ RUN      ] RelOp.WriteOut
[       OK ] RelOp.WriteOut (7 ms)
[----------] 3 tests from RelOp (9 ms total)

[----------] 1 test from Heap
[ RUN      ] Heap.Create
[       OK ] Heap.Create (22 ms)
[----------] 1 test from Heap (22 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 2 test suites ran. (33 ms total)
[  PASSED  ] 4 tests.
```

Conclusion: We have successfully implemented a set of Relational Operators in the Relational Database.