Data Base System Implementation
COP-6726

Submitted by:
Naman Arora
UFID: 3979-0439
Nikunj Sarda
UFID: 9360-6581

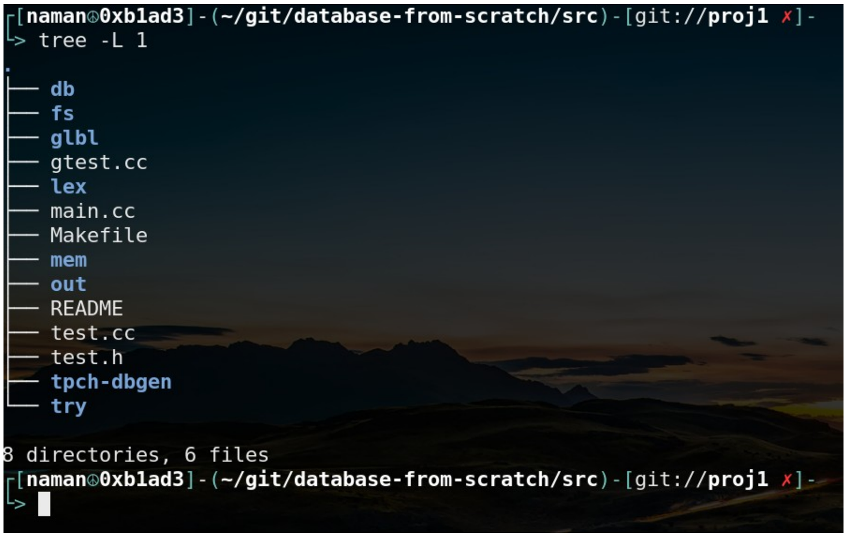
Introduction:

This document is intended to present the outcomes and the process of the completed work for the Project 1. The *heap file* implementation has been successfully materialized and tested. The repository for all the work is present here.

Source File Hierarchy:

The Source code for this project, including the boilerplate code, was rearranged from the ground up into logical sub directories, explained as follows:

All the subdirectories contain respective Makefiles that, when called recursively, compile the contained source code into separate object files. These object files are then linked by the root Makefile to produce the output binary of the program as required.



```
┌[naman☮0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> tree -L 1
.
├── db
├── fs
├── glbl
├── gtest.cc
├── lex
├── main.cc
├── Makefile
├── mem
├── out
├── README
├── test.cc
├── test.h
├── tpch-dbgen
└── try

8 directories, 6 files
┌[naman☮0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> █
```

The db/ sub-dir:

This subdirectory contains the schema.cc, schema.h, catalog file and its Makefile.

The fs/ sub-dir:

This subdirectory contains the filesystem related source files, *viz*. file.cc, record.cc and dbfile.cc with their respective header files and the Makefile.

The mem/ sub-dir:

This subdirectory comprises of the two_way_list.cc and its header files with the required Makefile. (Fig. 4)

The glbl/ sub-dir:

This subdirectory contains the defs.h header along with the timer class source code in timer.cc and timer.h.

The lex/ sub-dir:

The subdirectory houses all the flex/yacc related source code, *viz.* comparison.cc, comparison_engine.cc, lexer.l, parser.y and parse_tree.h along with respective headers and Makefile.

The out/ and try/ sub-dir:

out/ directory will house the <relation_name>.bin files after the DBFile class bulk loads the *.tbl files. While the try/ subdirectory can contain any trial code or files that wont be comitted to the git repository. The contents of both these sub-directories are ignored by the git.

The tpch-dbgen subdir:

This is a git submodule with origin at https://github.com/electrum/tpch-dbgen.git.

The src/ root level directory:

The src/ directory acts as the root level directory as all the paths in source code are relative to this directory.

All the subdirectories have soft links to some of the other subdirectories. This facilitates the linking process when an object file from subdirectory-1 requires to be linked with object file of subdirectory-2.

The compilation process:

The compilation process is accomplished by a recursive call to Makefiles in each subdirectory by the Makefile in the src/ directory.

The provided NamanArora_NikunjSarda_p1.zip has one sub-directory, src/.

```
$> unzip NamanArora_NikunjSarda_p1.zip
$> cd proj1/src
```

Now to build the test.bin:

```
$> make
$> ./test.bin #run
```

To build the main.bin:

```
$> make main_all
$> ./main.bin  #run, make sure tpch-dbgen/*.tbl files exist
```

To make the gtest (assuming the gtest framework is installed and available in default linker library path)

```
$> make gtest_all
$> ./gtest.bin  #run, make sure tpch-dbgen/*.tbl files exist
```

To generate tpch-dbgen files (tpch-dbgen/ sub-dir is always compiled with all of the above targets)

```
$> cd tpch-dbgen/
$> ./dbgen <options> -s <size>
```

All the available relation binary files are available in out/

To clean the repository, excluding the tpch-dbgen/*.tbl and out/*.bin files,

```
$> make clean
```

To clean the repository, everything inclusive

```
$> make distclean
```

Changelog:

→ Added DBFile class in fs/dbfile.{cc,h}.

→ Added timer class in glbl/timer.{cc,h}.

→ Added Page::get_curr_size() function definition in fs/file.{cc,h}.

→ Changed the directory structure, see section 2.

→ Fixed memory leak in class Schema with char *name in struct myAtts and char *fileName in db/schema.{cc,h}.

→ Fixed memory leaks in test.h within relation *rel and Schema * memory addresses.

→ Added Record::get_size() function in fs/record.{cc,h}.

→ test2() and test3() in test.cc now call DBFile::GetNext with Record *temp as the placeholder and store the value as opposed to previous version where either Record **temp was required or a copy from DBFile::head to temp.

→ Added 8th relation as option in main function of test.cc, reconfigured all the options and relations in compliance with changes in test.h.

→ Changed the return type of File::Open to integer in order to indicate error, if any, with the open syscall. Zero is failure and 1 is success.

→ Changed the File::Close return from curLength in pages to error indication when write or close syscalls fail. Zero indicates error and one success.

→ Fixed indentation all across.

The DBFile Class:

The dbfile class exists in fs/dbfile.{cc,h} files and has the following public functions:

→ DBFile::DBFile()

The constructor that instantiates the private variables to their required states.

→ DBFile::~DBFile()

The destructor that frees up memory from all the allocated variables.

→ DBFile::Create(const char *fname, fType type, void *startup)

This function creates the File class instance and return 0 on error and 1 on success.

→ DBFile::Open(const char *fname)

This function opens an already created File at the specified location and fetches the zeroth page. Returns 0 on error and 1 on success.

→ DBFile::MoveFirst()

This function loads up the zeroth page for reading and writes back to disk if already writing.

→ DBFile::Add(Record *placeholder)

This function fetches the latest page for writing, writes page to disk if full and adds the record to class Page instance.

→ DBFile::Load(const char *fname)

This function bulk loads a .tbl file from disk and makes corresponding out/*.bin file.

→ DBFile::GetNext(Record *placeholder)

This function check if the page has been fully read, if yes fetches the next one, then reads the next record. If EOF returns 0 else 1.

→ DBFile::GetNext(Record *placeholder, CNF *exp, Record *literal)

This function calls the ComparisonEngine Object and GetNext(above) function to check if a record applies to the given CNF. Keeps checking records until one satisfies, returns 1. On Error from (above) GetNext, returns 0.

→ DBFile::Close()

This closes the File class instance by writing back the Page if anything is left to write. Returns 0 on error and 1 on success.

The class DBFile has the following private functions:

→ DBFile::writeback()

This writes the page back using the File::AddPage public function.

→ DBFile::fetch(off_t pg_num)

This fetches the specified page from the disk using File::GetPage.

→ DBFile::set_dirty()

This sets the dirty bit to indicate latest changes to page have not been written back to disk.

→ DBFile::unset_dirty()

This unsets the dirty bit indicating disk is in sync with the page.

→ DBFile::chk_dirty()

This fetches the value of dirty bit, to check if the page needs a writeback before context switch.

Some cautions while compiling/running code:

→ Only the Makefile in the src/ must be called at all times.

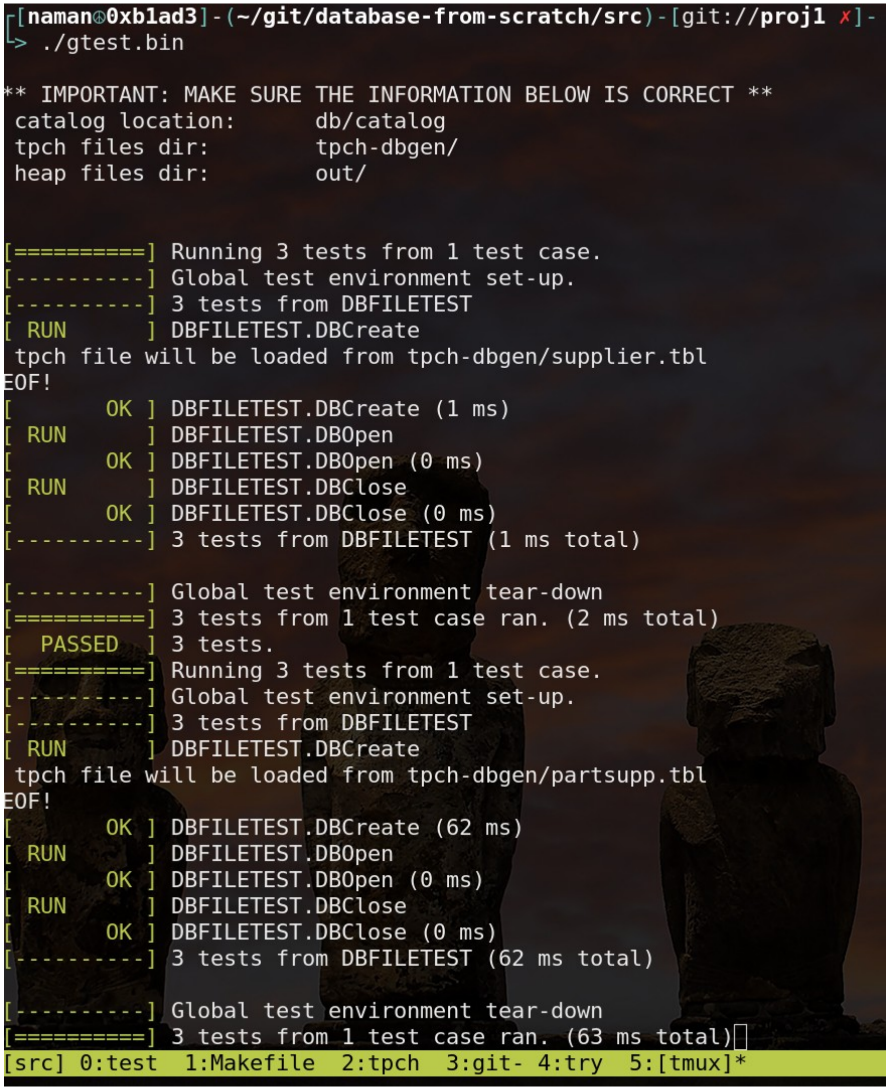
→ All the directories have have been hard coded in the code. Any replacement might lead to faliure in compilation or execution.

→ While CNF is being input in the case of test3() of test.cc, all the string values MUST be in capital letters and l_quantity for the Query 11 must be put as a Double.

→ While executing main.bin, presence of tpch-dbgen/*.tbl files is must else it will cause segfault.

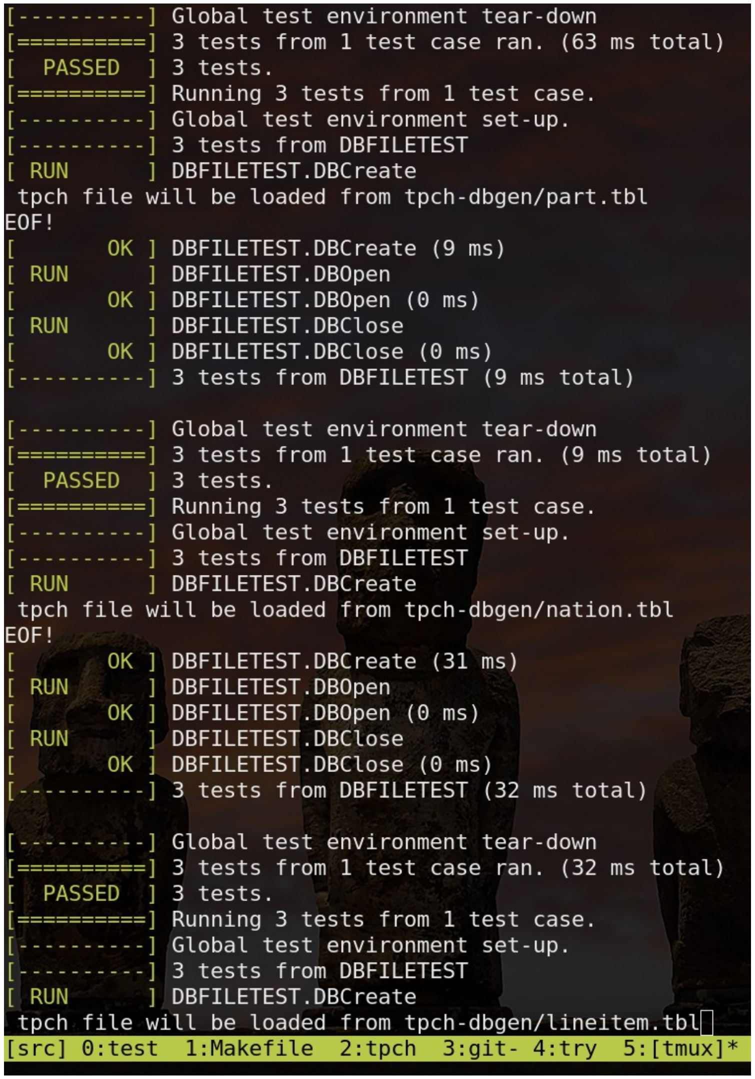
Outputs

Gtest Output (continuation screenshots)



```
┌[naman⊛0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> ./gtest.bin

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      db/catalog
 tpch files dir:        tpch-dbgen/
 heap files dir:        out/


[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/supplier.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (1 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (1 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (2 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/partsupp.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (62 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (62 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (63 ms total)
[src] 0:test  1:Makefile  2:tpch  3:git- 4:try  5:[tmux]*
```

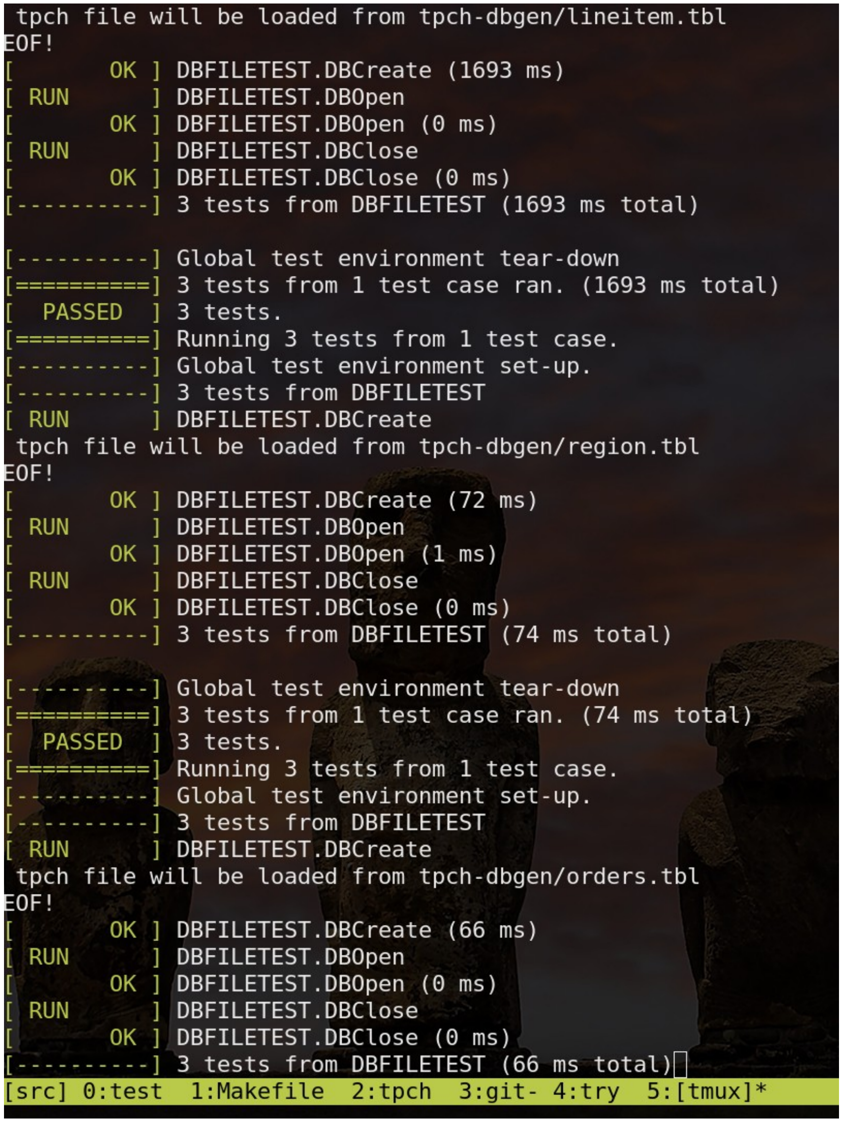
Screenshot 1



```
[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (63 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/part.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (9 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (9 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (9 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/nation.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (31 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (32 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (32 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/lineitem.tbl
[src] 0:test  1:Makefile  2:tpch  3:git- 4:try  5:[tmux]*
```

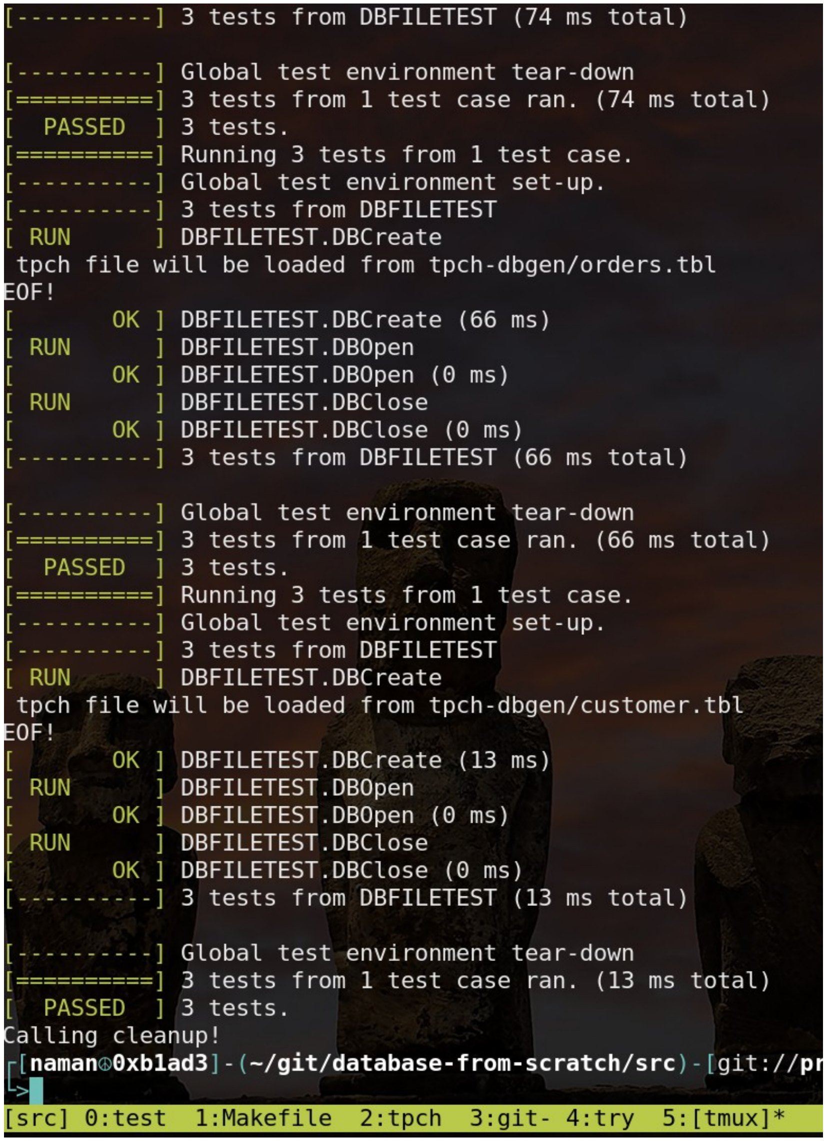
Screenshot 2

Gtest Output (continuation screenshots)



```
 tpch file will be loaded from tpch-dbgen/lineitem.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (1693 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (1693 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (1693 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/region.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (72 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (1 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (74 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (74 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/orders.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (66 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (66 ms total)
[src] 0:test  1:Makefile  2:tpch  3:git- 4:try  5:[tmux]*
```

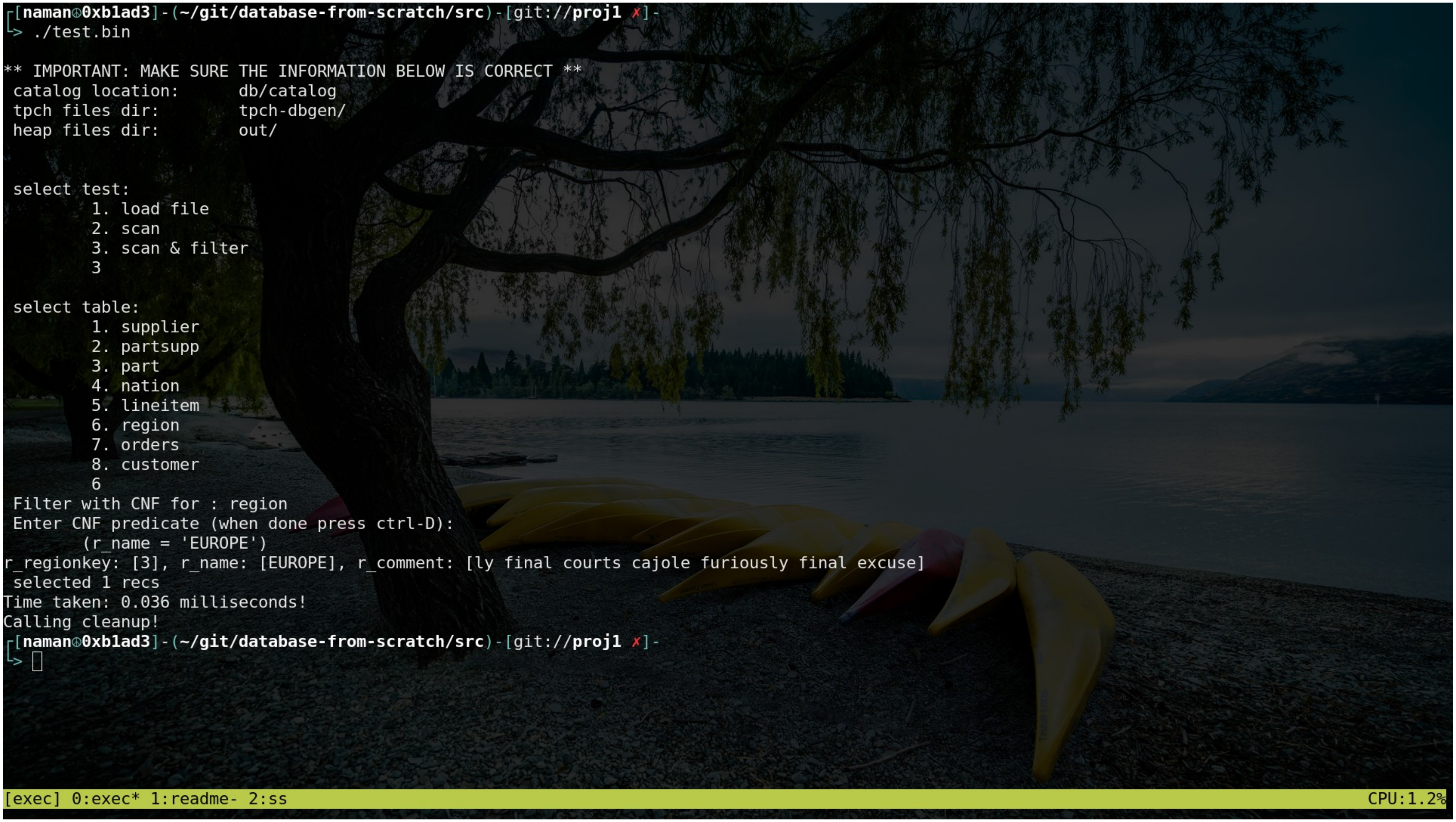
Screenshot 3



```
[----------] 3 tests from DBFILETEST (74 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (74 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/orders.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (66 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (66 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (66 ms total)
[  PASSED  ] 3 tests.
[==========] Running 3 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 3 tests from DBFILETEST
[ RUN      ] DBFILETEST.DBCreate
 tpch file will be loaded from tpch-dbgen/customer.tbl
EOF!
[       OK ] DBFILETEST.DBCreate (13 ms)
[ RUN      ] DBFILETEST.DBOpen
[       OK ] DBFILETEST.DBOpen (0 ms)
[ RUN      ] DBFILETEST.DBClose
[       OK ] DBFILETEST.DBClose (0 ms)
[----------] 3 tests from DBFILETEST (13 ms total)

[----------] Global test environment tear-down
[==========] 3 tests from 1 test case ran. (13 ms total)
[  PASSED  ] 3 tests.
Calling cleanup!
┌[naman☮0xb1ad3]-(~/git/database-from-scratch/src)-[git://pr
└>
[src] 0:test  1:Makefile  2:tpch  3:git- 4:try  5:[tmux]*
```

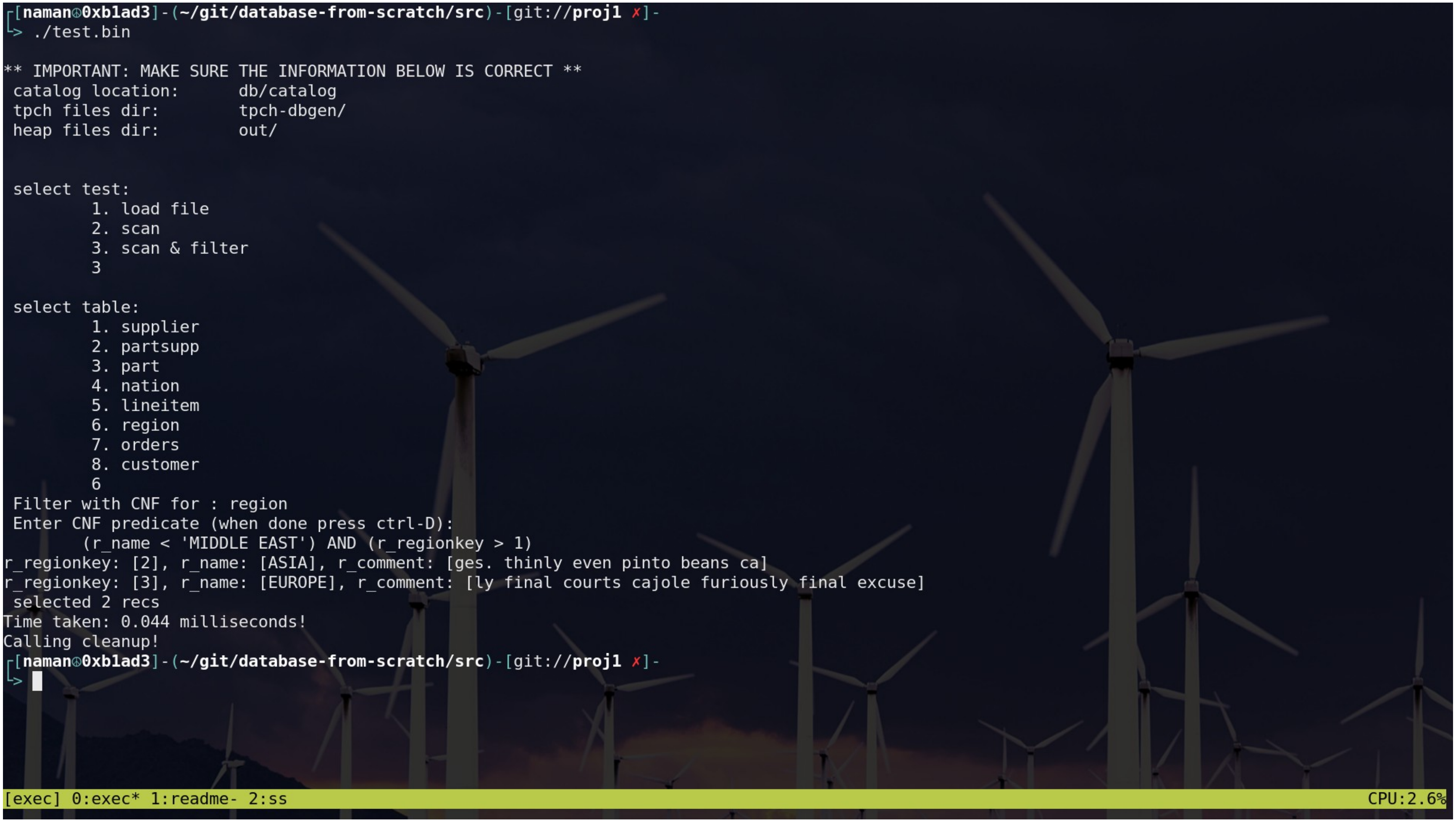
Screenshot 4

Query 1: (region)
(r_name = ‘EUROPE’)



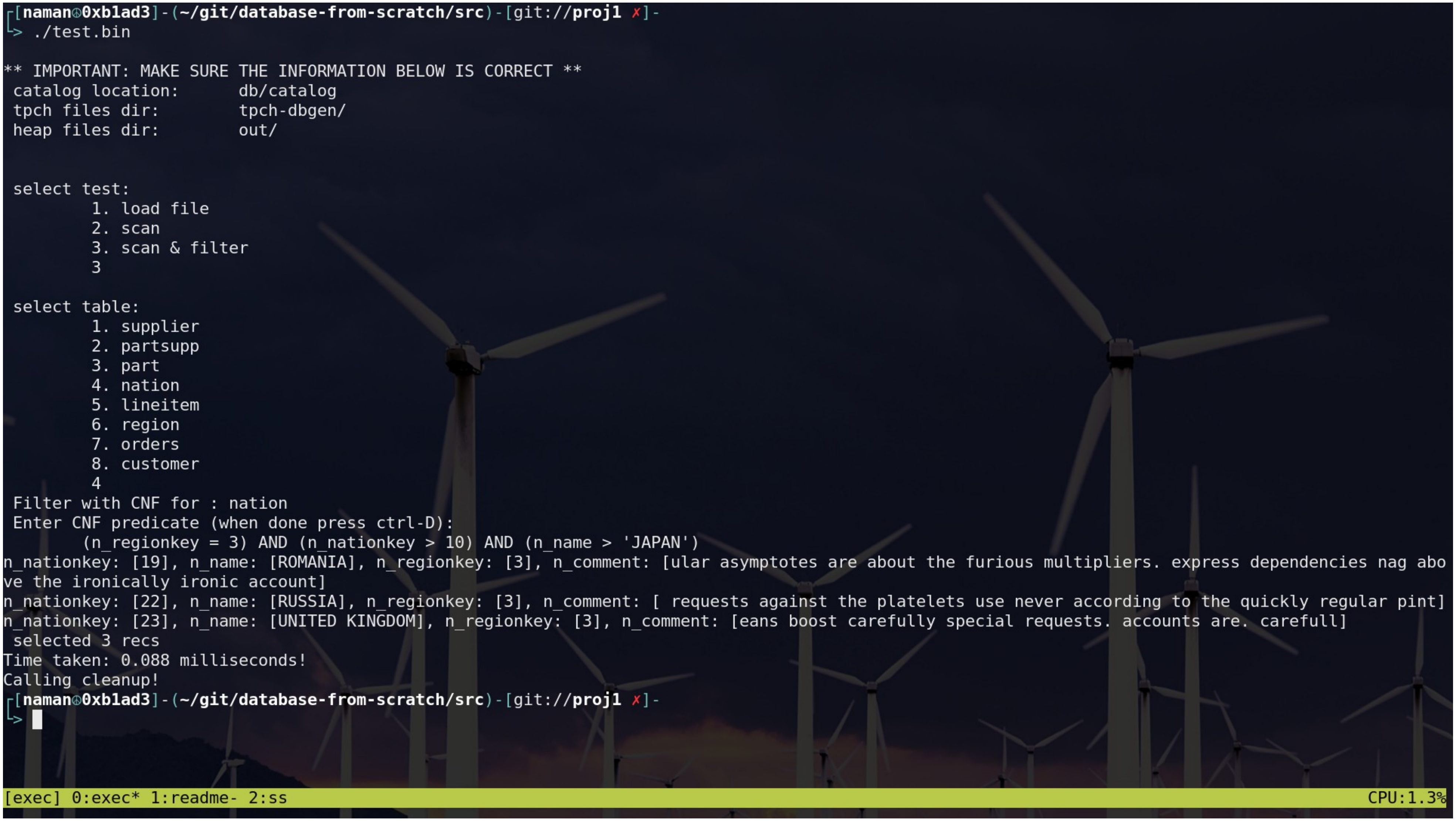
```
┌[naman⊕0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> ./test.bin

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      db/catalog
 tpch files dir:        tpch-dbgen/
 heap files dir:        out/


 select test:
         1. load file
         2. scan
         3. scan & filter
         3

 select table:
         1. supplier
         2. partsupp
         3. part
         4. nation
         5. lineitem
         6. region
         7. orders
         8. customer
         6
 Filter with CNF for : region
 Enter CNF predicate (when done press ctrl-D):
        (r_name = 'EUROPE')
r_regionkey: [3], r_name: [EUROPE], r_comment: [ly final courts cajole furiously final excuse]
 selected 1 recs
Time taken: 0.036 milliseconds!
Calling cleanup!
┌[naman⊕0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> 
```

Query 2: (region)
(r_name < ‘MIDDLE EAST’) AND (r_regionkey > 1)



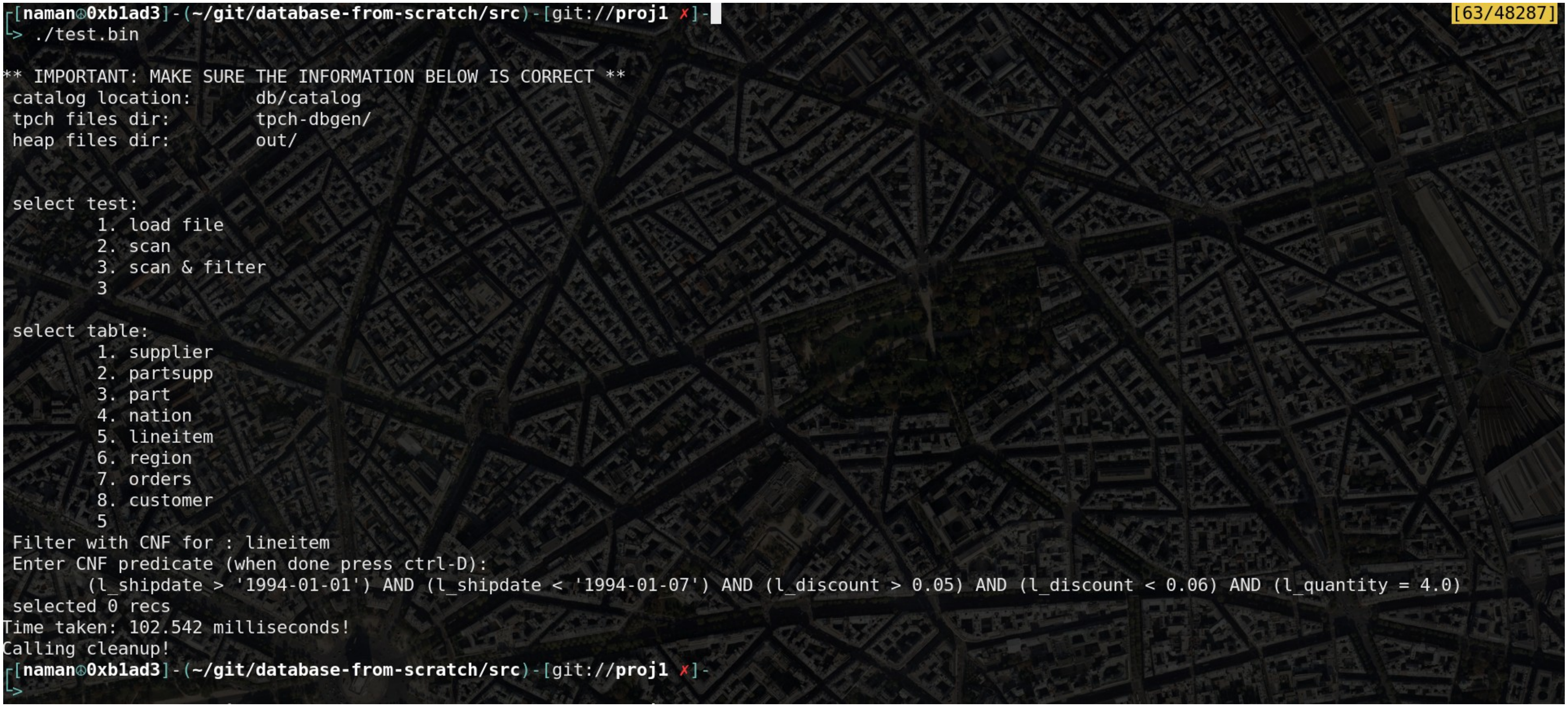
```
┌[naman⊛0xblad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> ./test.bin

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:       db/catalog
 tpch files dir:         tpch-dbgen/
 heap files dir:         out/


 select test:
         1. load file
         2. scan
         3. scan & filter
         3

 select table:
         1. supplier
         2. partsupp
         3. part
         4. nation
         5. lineitem
         6. region
         7. orders
         8. customer
         6
 Filter with CNF for : region
 Enter CNF predicate (when done press ctrl-D):
        (r_name < 'MIDDLE EAST') AND (r_regionkey > 1)
r_regionkey: [2], r_name: [ASIA], r_comment: [ges. thinly even pinto beans ca]
r_regionkey: [3], r_name: [EUROPE], r_comment: [ly final courts cajole furiously final excuse]
 selected 2 recs
Time taken: 0.044 milliseconds!
Calling cleanup!
┌[naman⊛0xblad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└>
[exec] 0:exec* 1:readme- 2:ss                                                CPU:2.6%
```

Query 3: (nation)
(n_regionkey =3) AND (n_nationkey > 10) AND (n_name > ‘JAPAN’)



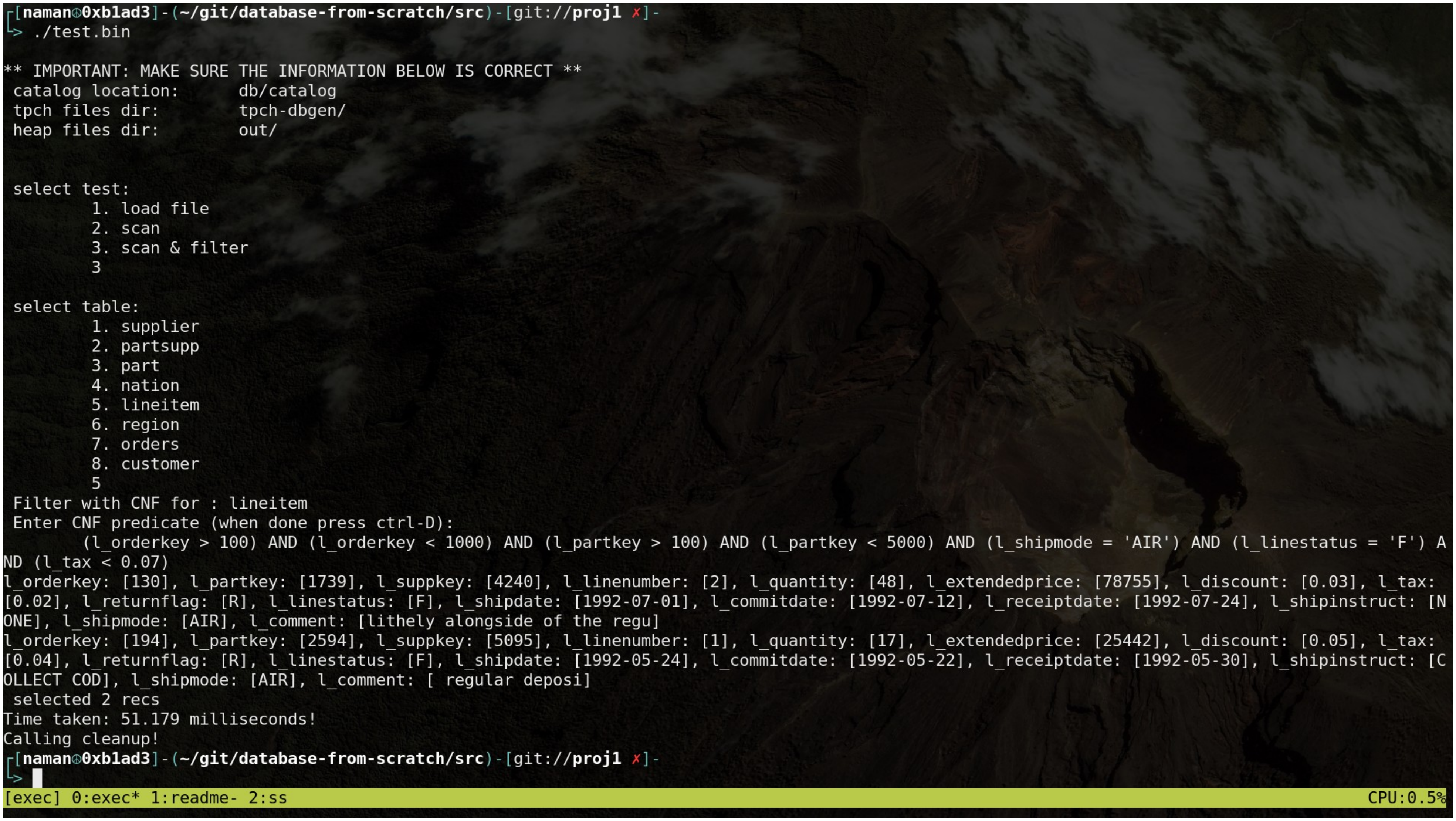
```
┌[naman☮0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> ./test.bin

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      db/catalog
 tpch files dir:        tpch-dbgen/
 heap files dir:        out/


 select test:
        1. load file
        2. scan
        3. scan & filter
        3

 select table:
        1. supplier
        2. partsupp
        3. part
        4. nation
        5. lineitem
        6. region
        7. orders
        8. customer
        4
 Filter with CNF for : nation
 Enter CNF predicate (when done press ctrl-D):
        (n_regionkey = 3) AND (n_nationkey > 10) AND (n_name > 'JAPAN')
n_nationkey: [19], n_name: [ROMANIA], n_regionkey: [3], n_comment: [ular asymptotes are about the furious multipliers. express dependencies nag abo
ve the ironically ironic account]
n_nationkey: [22], n_name: [RUSSIA], n_regionkey: [3], n_comment: [ requests against the platelets use never according to the quickly regular pint]
n_nationkey: [23], n_name: [UNITED KINGDOM], n_regionkey: [3], n_comment: [eans boost carefully special requests. accounts are. carefull]
 selected 3 recs
Time taken: 0.088 milliseconds!
Calling cleanup!
┌[naman☮0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> █

[exec] 0:exec* 1:readme- 2:ss                                                                                                              CPU:1.3%
```

Query 11: (lineitem)
(l_shipdate > ‘1994-01-01’) AND (l_shipdate < ‘1994-01-07’) AND (l_discount > 0.05) AND (l_discount < 0.06) AND (l_quantity = 4.0)



```
┌[naman⊕0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-                                                  [63/48287]
└> ./test.bin

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      db/catalog
 tpch files dir:        tpch-dbgen/
 heap files dir:        out/


 select test:
         1. load file
         2. scan
         3. scan & filter
         3

 select table:
         1. supplier
         2. partsupp
         3. part
         4. nation
         5. lineitem
         6. region
         7. orders
         8. customer
         5
 Filter with CNF for : lineitem
 Enter CNF predicate (when done press ctrl-D):
        (l_shipdate > '1994-01-01') AND (l_shipdate < '1994-01-07') AND (l_discount > 0.05) AND (l_discount < 0.06) AND (l_quantity = 4.0)
 selected 0 recs
Time taken: 102.542 milliseconds!
Calling cleanup!
┌[naman⊕0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└>
```

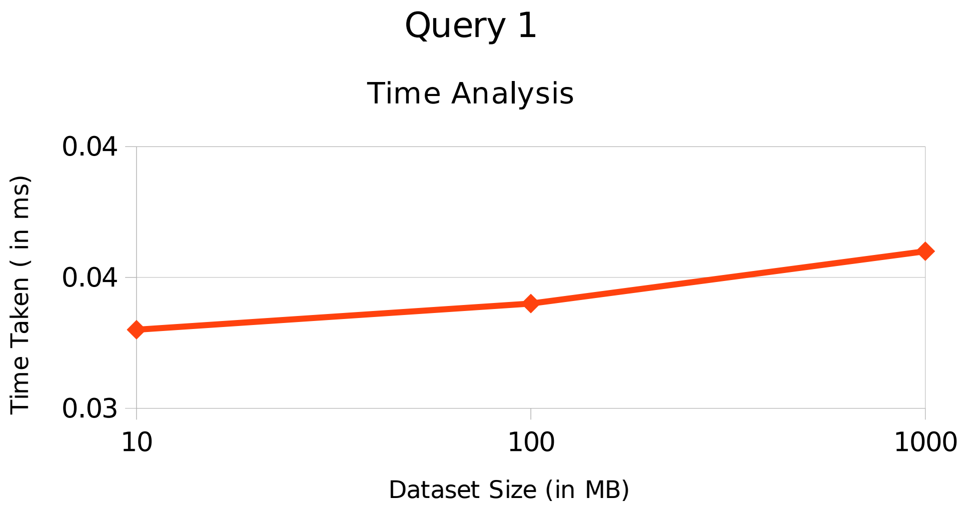
Query 12: (lineitem)
(l_orderkey > 100) AND (l_orderkey < 1000) AND (l_partkey > 100) AND (l_partkey < 5000) AND (l_shipmode = ‘AIR’) AND (l_linestatus = ‘F’) AND (l_tax < 0.07)



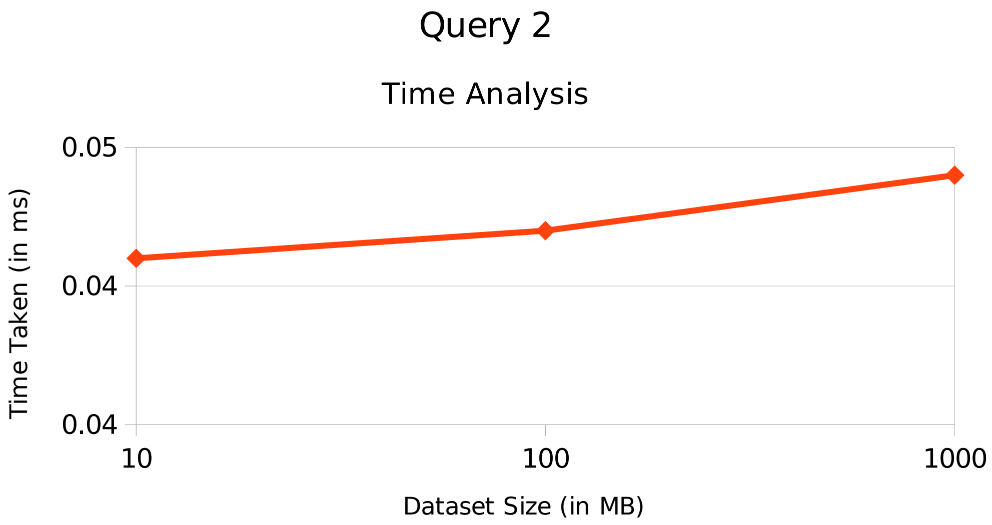
```
┌[naman⊕0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> ./test.bin

** IMPORTANT: MAKE SURE THE INFORMATION BELOW IS CORRECT **
 catalog location:      db/catalog
 tpch files dir:        tpch-dbgen/
 heap files dir:        out/


 select test:
        1. load file
        2. scan
        3. scan & filter
        3

 select table:
        1. supplier
        2. partsupp
        3. part
        4. nation
        5. lineitem
        6. region
        7. orders
        8. customer
        5
 Filter with CNF for : lineitem
 Enter CNF predicate (when done press ctrl-D):
        (l_orderkey > 100) AND (l_orderkey < 1000) AND (l_partkey > 100) AND (l_partkey < 5000) AND (l_shipmode = 'AIR') AND (l_linestatus = 'F') A
ND (l_tax < 0.07)
l_orderkey: [130], l_partkey: [1739], l_suppkey: [4240], l_linenumber: [2], l_quantity: [48], l_extendedprice: [78755], l_discount: [0.03], l_tax:
[0.02], l_returnflag: [R], l_linestatus: [F], l_shipdate: [1992-07-01], l_commitdate: [1992-07-12], l_receiptdate: [1992-07-24], l_shipinstruct: [N
ONE], l_shipmode: [AIR], l_comment: [lithely alongside of the regu]
l_orderkey: [194], l_partkey: [2594], l_suppkey: [5095], l_linenumber: [1], l_quantity: [17], l_extendedprice: [25442], l_discount: [0.05], l_tax:
[0.04], l_returnflag: [R], l_linestatus: [F], l_shipdate: [1992-05-24], l_commitdate: [1992-05-22], l_receiptdate: [1992-05-30], l_shipinstruct: [C
OLLECT COD], l_shipmode: [AIR], l_comment: [ regular deposi]
 selected 2 recs
Time taken: 51.179 milliseconds!
Calling cleanup!
┌[naman⊕0xb1ad3]-(~/git/database-from-scratch/src)-[git://proj1 ✗]-
└> 
[exec] 0:exec* 1:readme- 2:ss                                                                                                              CPU:0.5%
```

Analysis

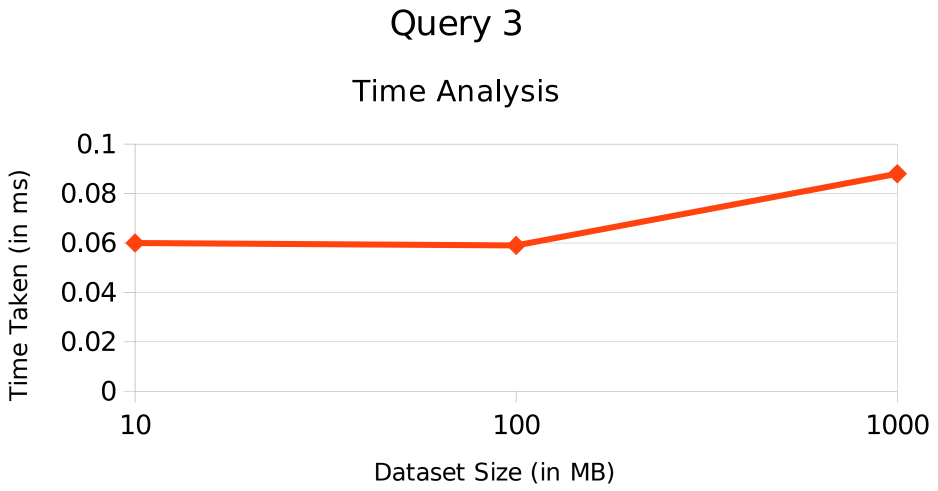
Query 1: (region)
(r_name = ‘EUROPE’)



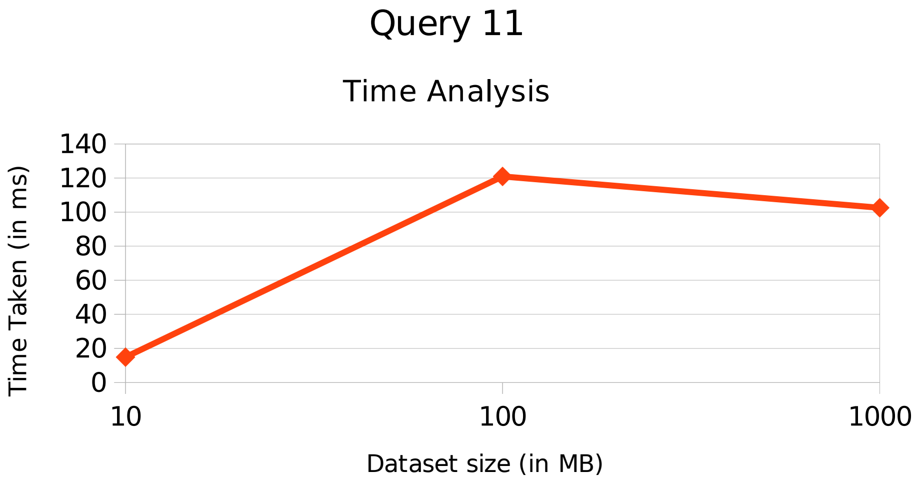
Query 2: (region)
(r_name < ‘MIDDLE EAST’) AND (r_regionkey > 1)



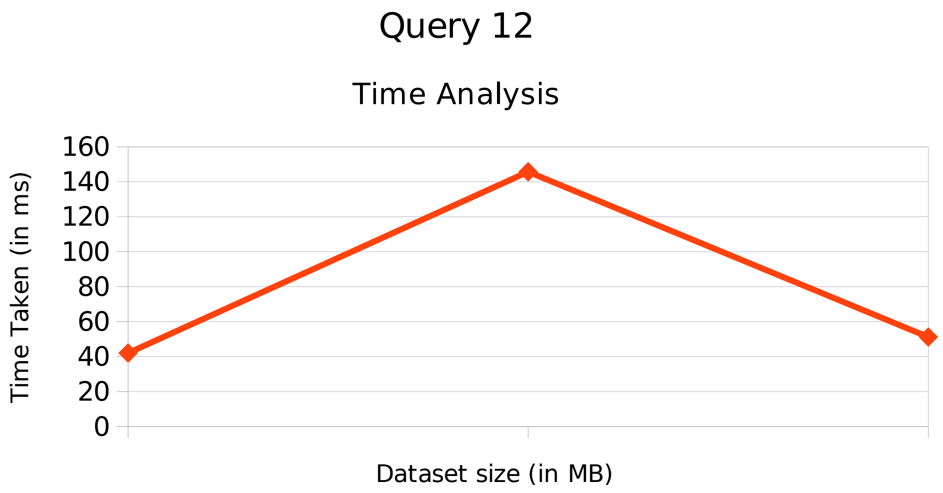
Query 3: (nation)
(n_regionkey =3) AND (n_nationkey > 10) AND (n_name > 'JAPAN')



Query 11: (lineitem)
(l_shipdate > '1994-01-01') AND (l_shipdate < '1994-01-07') AND (l_discount > 0.05) AND (l_discount < 0.06) AND (l_quantity = 4.0)



Query 12: (lineitem)
(l_orderkey > 100) AND (l_orderkey < 1000) AND (l_partkey > 100) AND (l_partkey < 5000) AND (l_shipmode = ‘AIR’) AND (l_linestatus = ‘F’) AND (l_tax < 0.07)


Query 12
Time Analysis
Time Taken (in ms)
Dataset size (in MB)