Debug Postgres with GDB

Table Of Contents:

About GDB:

GDB, the GNU Project debugger, allows you to see what is going on 'inside' another program while it executes, or what it was doing at the moment it crashed.

GDB can do four main things to help you catch bugs:

- Start your program, specifying anything that might affect its behavior.
- Make your program stop on specified conditions.
- Examine what has happened, when your program has stopped.
- Change things in your program, so you can experiment with correcting the effects of one bug and go on to learn about another.

Attaching GDB to Postgres:

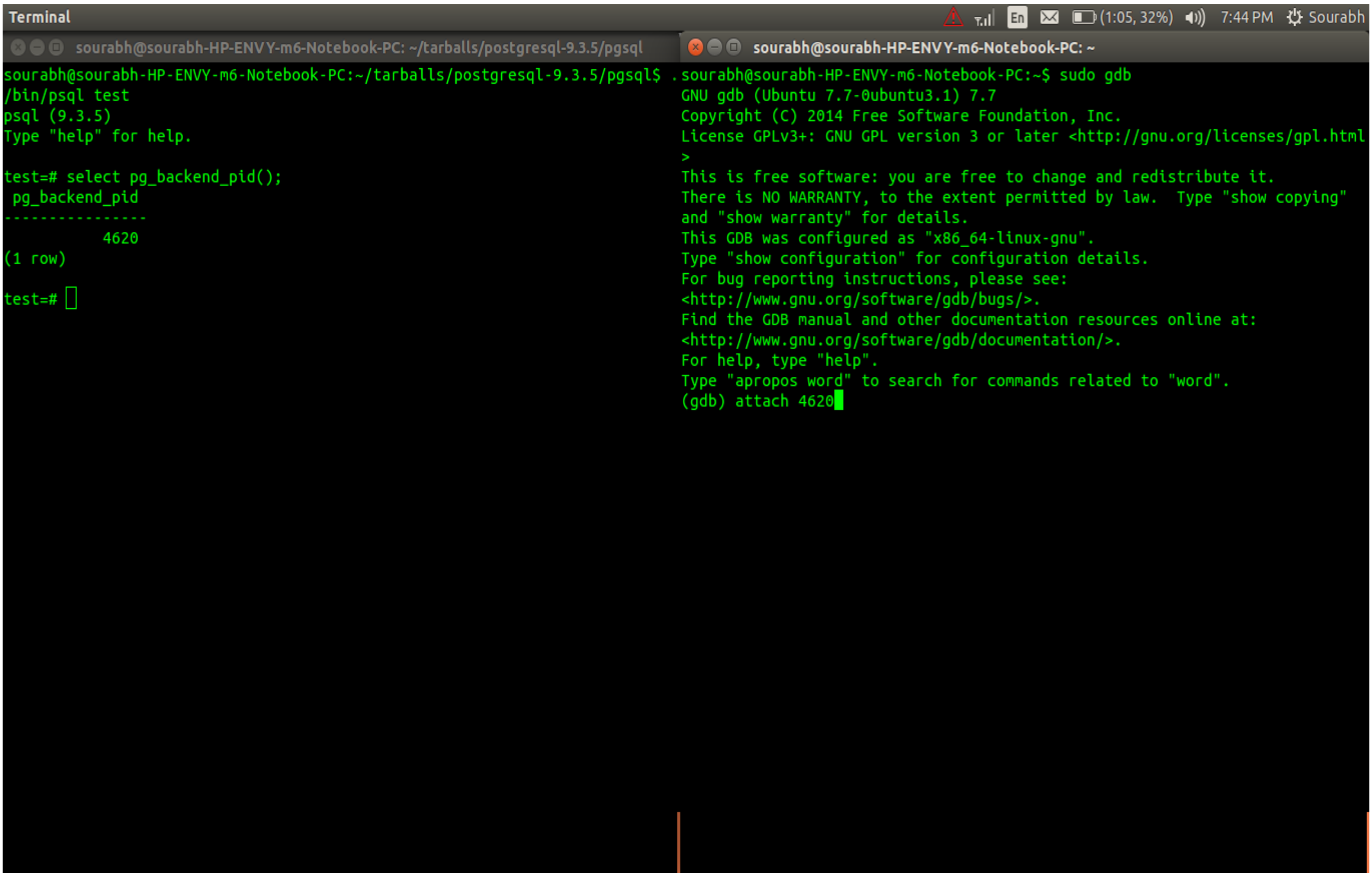
Open terminal and type

```
./bin/postgres -D data &
./bin/psql test
```

Find the Process id of this client session by typing

```
test=# SELECT pg_backend_pid();
```

This is the Process ID to which you need to attach the GDB and start debugging. The Snapshot is shown.



Terminal
En
(1:05, 32%)
7:44 PM
Sourabh
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~/tarballs/postgresql-9.3.5/pgsql
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~

```
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~/tarballs/postgresql-9.3.5/pgsql$ .
/bin/psql test
psql (9.3.5)
Type "help" for help.

test=# select pg_backend_pid();
 pg_backend_pid
----------------
           4620
(1 row)

test=# 
```

```
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~$ sudo gdb
GNU gdb (Ubuntu 7.7-0ubuntu3.1) 7.7
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html
>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
(gdb) attach 4620
```

GDB is opened in a separate terminal window. You are likely to need superuser permission to attach to the Postgres process if the UID's do not match.

To attach GDB to the process (PID=4620, in this case), type the following command

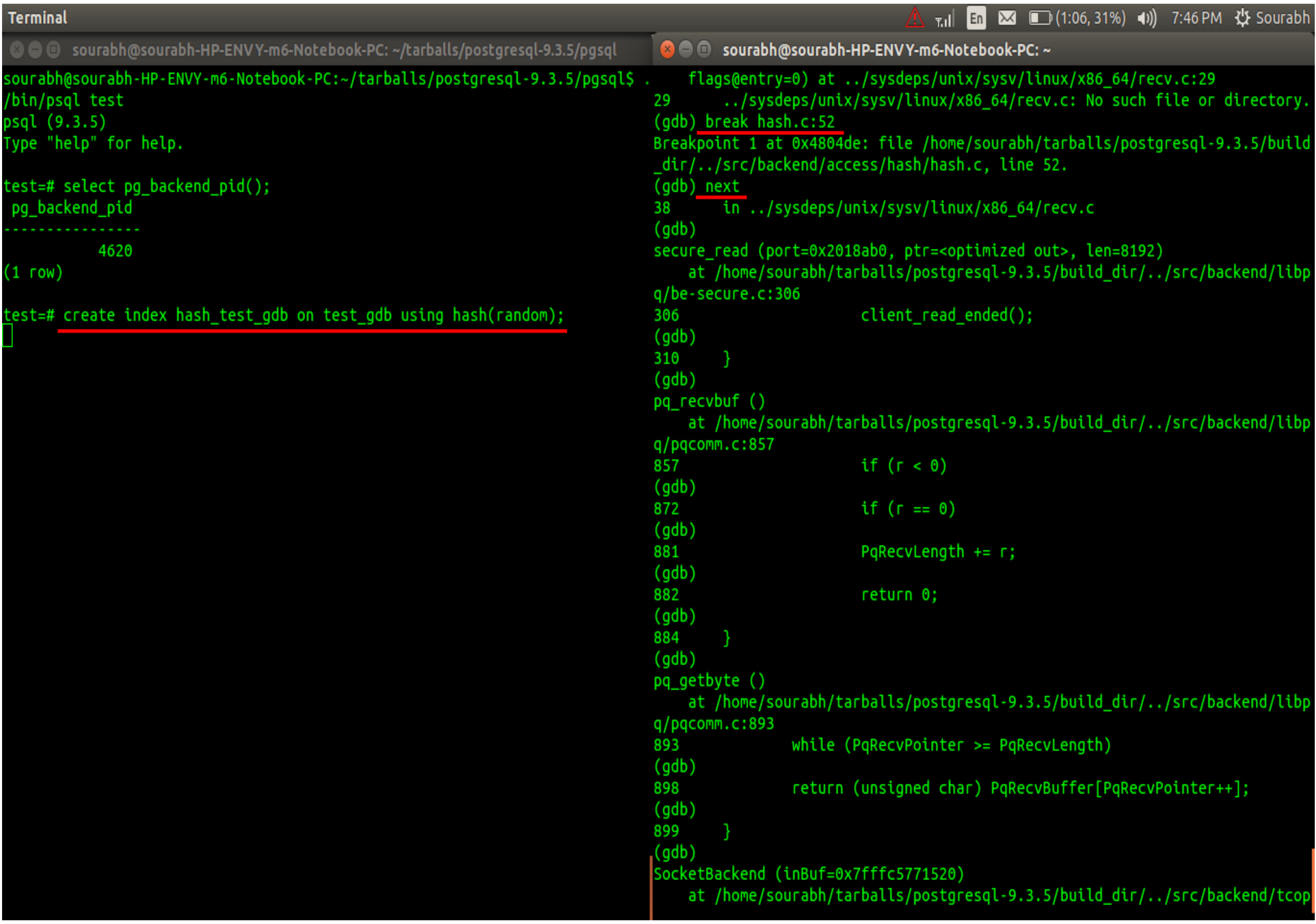
```
(gdb) attach 4620
```

Now that GDB is attached to the process, we can start the debugging process. Let us start by setting a breakpoint. In this example, I have set the breakpoint to line number 52 in the file hash.c.This function is invoked when we create a hash index for a column. The way to set breakpoint is as follows

```
(gdb) break hash.c:52
```

Note:
1. You can also set it to break when it calls a particular function.
2. b can be used instead of break.
3. Any number of breakpoints can be set.

Now type the command to Create a hash index at the psql prompt and start debugging.
Here's the screenshot.

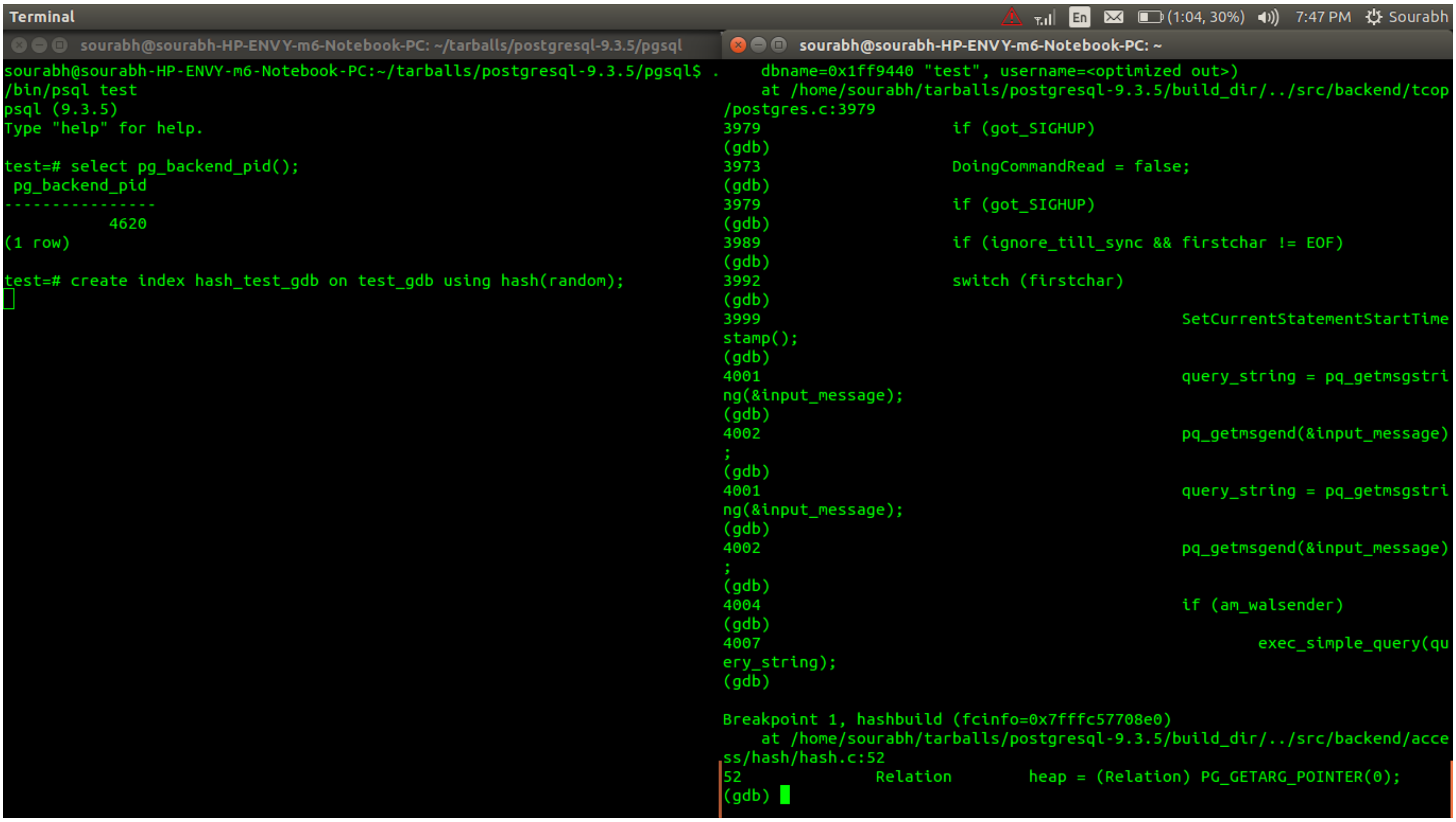

Terminal
(1:06, 31%)
7:46 PM
Sourabh
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~/tarballs/postgresql-9.3.5/pgsql
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~

```
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~/tarballs/postgresql-9.3.5/pgsql$ .
/bin/psql test
psql (9.3.5)
Type "help" for help.

test=# select pg_backend_pid();
 pg_backend_pid
----------------
           4620
(1 row)

test=# create index hash_test_gdb on test_gdb using hash(random);
```

```
    flags@entry=0) at ../sysdeps/unix/sysv/linux/x86_64/recv.c:29
29      ../sysdeps/unix/sysv/linux/x86_64/recv.c: No such file or directory.
(gdb) break hash.c:52
Breakpoint 1 at 0x4804de: file /home/sourabh/tarballs/postgresql-9.3.5/build
_dir/../src/backend/access/hash/hash.c, line 52.
(gdb) next
38      in ../sysdeps/unix/sysv/linux/x86_64/recv.c
(gdb)
secure_read (port=0x2018ab0, ptr=<optimized out>, len=8192)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/libp
q/be-secure.c:306
306                             client_read_ended();
(gdb)
310     }
(gdb)
pq_recvbuf ()
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/libp
q/pqcomm.c:857
857                             if (r < 0)
(gdb)
872                             if (r == 0)
(gdb)
881                             PqRecvLength += r;
(gdb)
882                             return 0;
(gdb)
884     }
(gdb)
pq_getbyte ()
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/libp
q/pqcomm.c:893
893             while (PqRecvPointer >= PqRecvLength)
(gdb)
898             return (unsigned char) PqRecvBuffer[PqRecvPointer++];
(gdb)
899     }
(gdb)
SocketBackend (inBuf=0x7fffc5771520)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/tcop
```

- The 'next' option allows us to check the next line of code that will be executed. 'n' can be used instead. This option jumps function calls
- The 'step' option (or 's') is used to enter into the function definition on function call
- Pressing 'Enter' repeats the last command

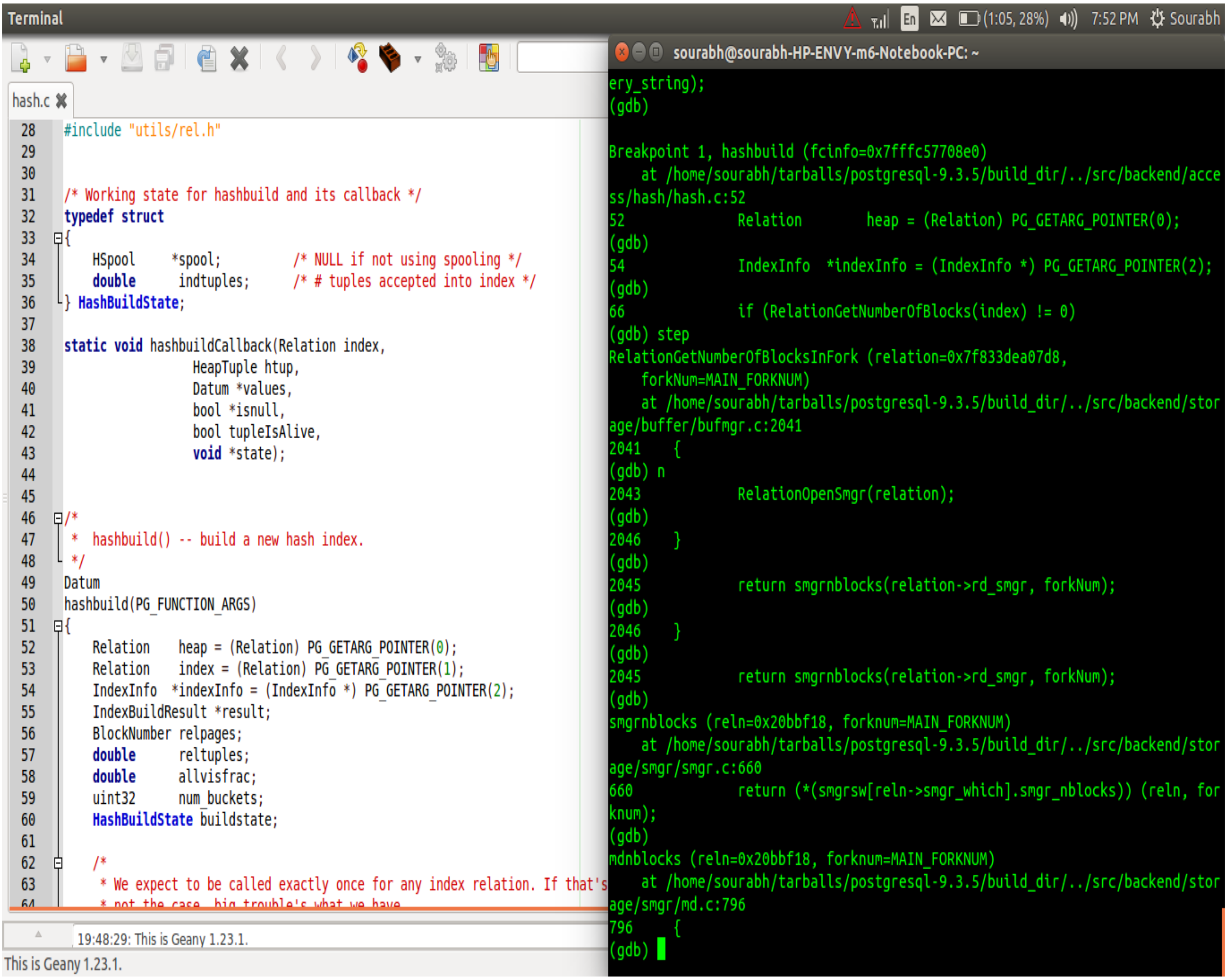
In the above screenshot, on typing next and pressing enter a few times, the codes related to accepting and processing the query are shown. Next we show the part where we reach the line number 52 in hash.c


Terminal
(1:04, 30%)
7:47 PM
Sourabh
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~/tarballs/postgresql-9.3.5/pgsql
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~

```
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~/tarballs/postgresql-9.3.5/pgsql$ .
/bin/psql test
psql (9.3.5)
Type "help" for help.

test=# select pg_backend_pid();
 pg_backend_pid
----------------
           4620
(1 row)

test=# create index hash_test_gdb on test_gdb using hash(random);
```

```
    dbname=0x1ff9440 "test", username=<optimized out>)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/tcop
/postgres.c:3979
3979                    if (got_SIGHUP)
(gdb)
3973                    DoingCommandRead = false;
(gdb)
3979                    if (got_SIGHUP)
(gdb)
3989                    if (ignore_till_sync && firstchar != EOF)
(gdb)
3992                    switch (firstchar)
(gdb)
3999                                            SetCurrentStatementStartTime
stamp();
(gdb)
4001                                            query_string = pq_getmsgstri
ng(&input_message);
(gdb)
4002                                            pq_getmsgend(&input_message)
;
(gdb)
4001                                            query_string = pq_getmsgstri
ng(&input_message);
(gdb)
4002                                            pq_getmsgend(&input_message)
;
(gdb)
4004                                            if (am_walsender)
(gdb)
4007                                                    exec_simple_query(qu
ery_string);
(gdb)

Breakpoint 1, hashbuild (fcinfo=0x7fffc57708e0)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/acce
ss/hash/hash.c:52
52              Relation        heap = (Relation) PG_GETARG_POINTER(0);
(gdb) 
```

At this point, We have reached out first (And in this case, the only) breakpoint. Keep pressing enter (next) till you reach the line “if(RelationGetNumberofBlocks(index) !=0)”
If you want to skip debugging this function call and go to the next line, type 'next'. Else type 'step'. Here we choose to go into the function definition by typing 'step'.



```
#include "utils/rel.h"


/* Working state for hashbuild and its callback */
typedef struct
{
    HSpool     *spool;         /* NULL if not using spooling */
    double      indtuples;     /* # tuples accepted into index */
} HashBuildState;

static void hashbuildCallback(Relation index,
                  HeapTuple htup,
                  Datum *values,
                  bool *isnull,
                  bool tupleIsAlive,
                  void *state);


/*
 *  hashbuild() -- build a new hash index.
 */
Datum
hashbuild(PG_FUNCTION_ARGS)
{
    Relation    heap = (Relation) PG_GETARG_POINTER(0);
    Relation    index = (Relation) PG_GETARG_POINTER(1);
    IndexInfo  *indexInfo = (IndexInfo *) PG_GETARG_POINTER(2);
    IndexBuildResult *result;
    BlockNumber relpages;
    double      reltuples;
    double      allvisfrac;
    uint32      num_buckets;
    HashBuildState buildstate;

    /*
     * We expect to be called exactly once for any index relation. If that's
     * not the case, big trouble's what we have.
```

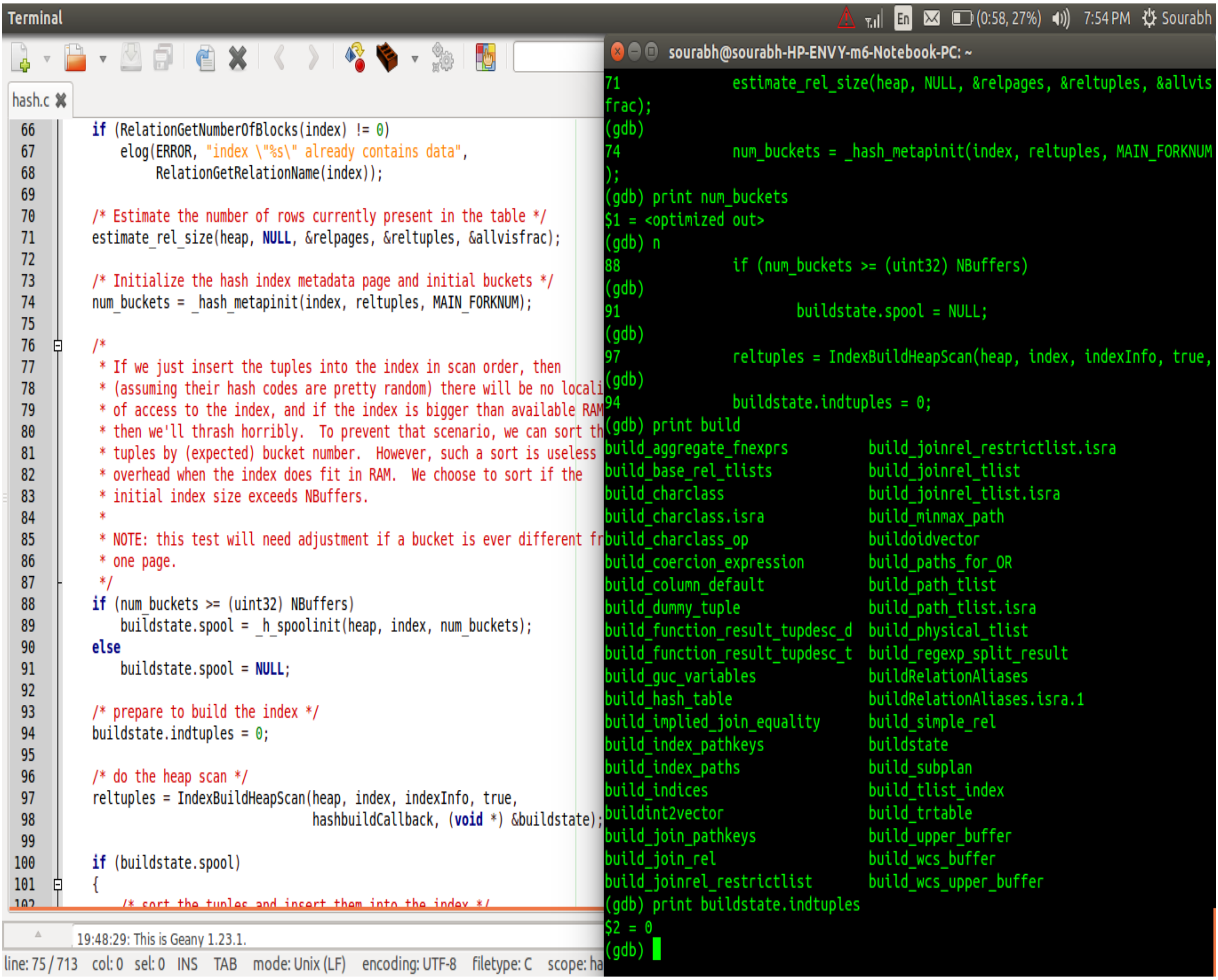
19:48:29: This is Geany 1.23.1.

This is Geany 1.23.1.

```
ery_string);
(gdb)

Breakpoint 1, hashbuild (fcinfo=0x7fffc57708e0)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/acce
ss/hash/hash.c:52
52          Relation        heap = (Relation) PG_GETARG_POINTER(0);
(gdb)
54          IndexInfo  *indexInfo = (IndexInfo *) PG_GETARG_POINTER(2);
(gdb)
66          if (RelationGetNumberOfBlocks(index) != 0)
(gdb) step
RelationGetNumberOfBlocksInFork (relation=0x7f833dea07d8,
    forkNum=MAIN_FORKNUM)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/stor
age/buffer/bufmgr.c:2041
2041    {
(gdb) n
2043            RelationOpenSmgr(relation);
(gdb)
2046    }
(gdb)
2045            return smgrnblocks(relation->rd_smgr, forkNum);
(gdb)
2046    }
(gdb)
2045            return smgrnblocks(relation->rd_smgr, forkNum);
(gdb)
smgrnblocks (reln=0x20bbf18, forknum=MAIN_FORKNUM)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/stor
age/smgr/smgr.c:660
660         return (*(smgrsw[reln->smgr_which].smgr_nblocks)) (reln, for
knum);
(gdb)
mdnblocks (reln=0x20bbf18, forknum=MAIN_FORKNUM)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/stor
age/smgr/md.c:796
796     {
(gdb) 
```

What if you want to check the values of variables that are in the program? Type 'next' and reach upto line 94, where we can see an assignment statement: buildstate.indtuples=0

To see the value of the variable, type 'print' followed by the variable name as shown in the next screenshot.



```
66      if (RelationGetNumberOfBlocks(index) != 0)
67          elog(ERROR, "index \"%s\" already contains data",
68               RelationGetRelationName(index));
69  
70      /* Estimate the number of rows currently present in the table */
71      estimate_rel_size(heap, NULL, &relpages, &reltuples, &allvisfrac);
72  
73      /* Initialize the hash index metadata page and initial buckets */
74      num_buckets = _hash_metapinit(index, reltuples, MAIN_FORKNUM);
75  
76      /*
77       * If we just insert the tuples into the index in scan order, then
78       * (assuming their hash codes are pretty random) there will be no locali
79       * of access to the index, and if the index is bigger than available RAM
80       * then we'll thrash horribly.  To prevent that scenario, we can sort th
81       * tuples by (expected) bucket number.  However, such a sort is useless
82       * overhead when the index does fit in RAM.  We choose to sort if the
83       * initial index size exceeds NBuffers.
84       *
85       * NOTE: this test will need adjustment if a bucket is ever different fr
86       * one page.
87       */
88      if (num_buckets >= (uint32) NBuffers)
89          buildstate.spool = _h_spoolinit(heap, index, num_buckets);
90      else
91          buildstate.spool = NULL;
92  
93      /* prepare to build the index */
94      buildstate.indtuples = 0;
95  
96      /* do the heap scan */
97      reltuples = IndexBuildHeapScan(heap, index, indexInfo, true,
98                                     hashbuildCallback, (void *) &buildstate);
99  
100     if (buildstate.spool)
101     {
102         /* sort the tuples and insert them into the index */
```

19:48:29: This is Geany 1.23.1.

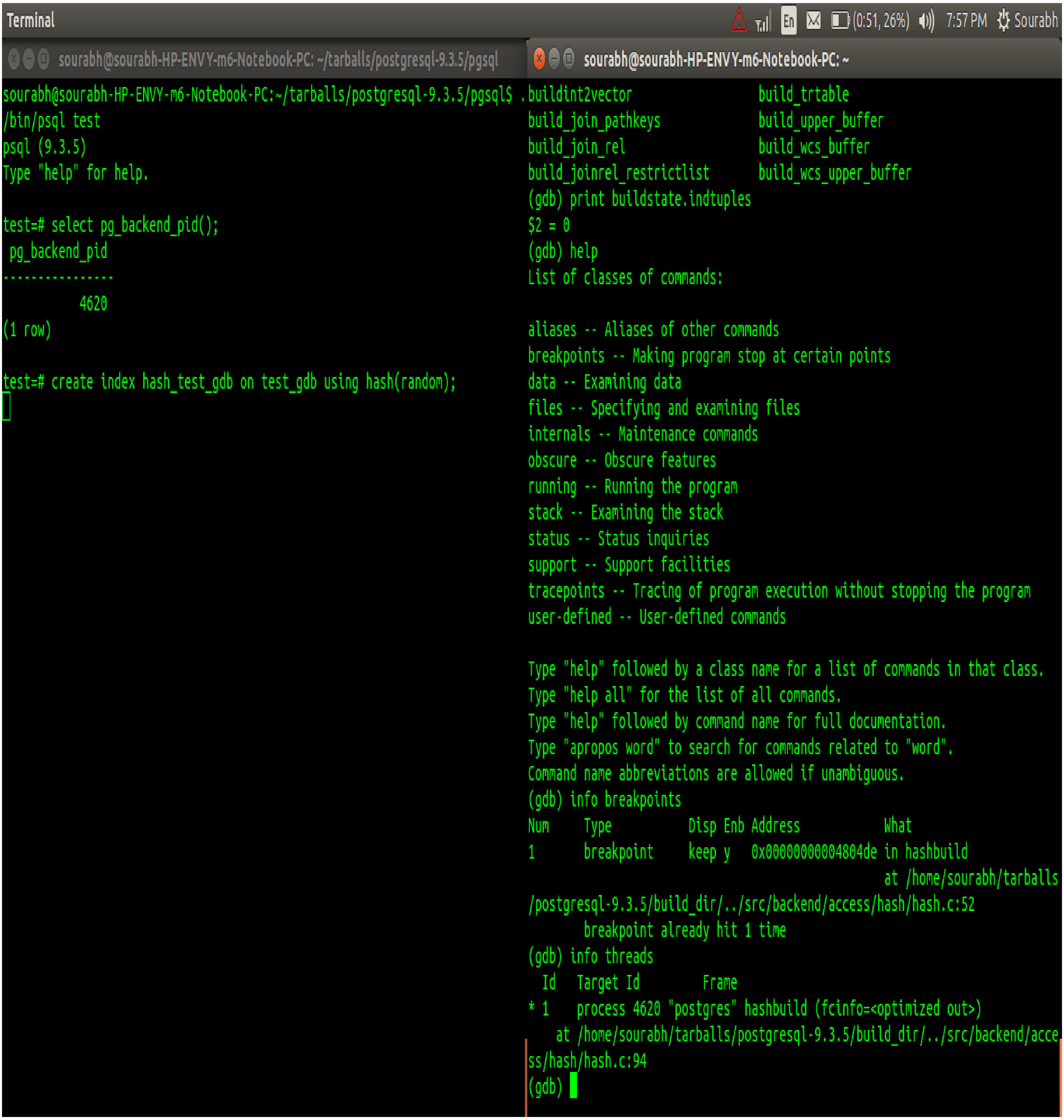
line: 75 / 713   col: 0   sel: 0   INS   TAB   mode: Unix (LF)   encoding: UTF-8   filetype: C   scope: ha

```
71              estimate_rel_size(heap, NULL, &relpages, &reltuples, &allvis
frac);
(gdb) 
74              num_buckets = _hash_metapinit(index, reltuples, MAIN_FORKNUM
);
(gdb) print num_buckets
$1 = <optimized out>
(gdb) n
88              if (num_buckets >= (uint32) NBuffers)
(gdb) 
91                      buildstate.spool = NULL;
(gdb) 
97              reltuples = IndexBuildHeapScan(heap, index, indexInfo, true,
(gdb) 
94              buildstate.indtuples = 0;
(gdb) print build
build_aggregate_fnexprs          build_joinrel_restrictlist.isra
build_base_rel_tlists            build_joinrel_tlist
build_charclass                  build_joinrel_tlist.isra
build_charclass.isra             build_minmax_path
build_charclass_op               buildoidvector
build_coercion_expression        build_paths_for_OR
build_column_default             build_path_tlist
build_dummy_tuple                build_path_tlist.isra
build_function_result_tupdesc_d  build_physical_tlist
build_function_result_tupdesc_t  build_regexp_split_result
build_guc_variables              buildRelationAliases
build_hash_table                 buildRelationAliases.isra.1
build_implied_join_equality      build_simple_rel
build_index_pathkeys             buildstate
build_index_paths                build_subplan
build_indices                    build_tlist_index
buildint2vector                  build_trtable
build_join_pathkeys              build_upper_buffer
build_join_rel                   build_wcs_buffer
build_joinrel_restrictlist       build_wcs_upper_buffer
(gdb) print buildstate.indtuples
$2 = 0
(gdb) 
```

Here are some other oprions that are available in GDB:

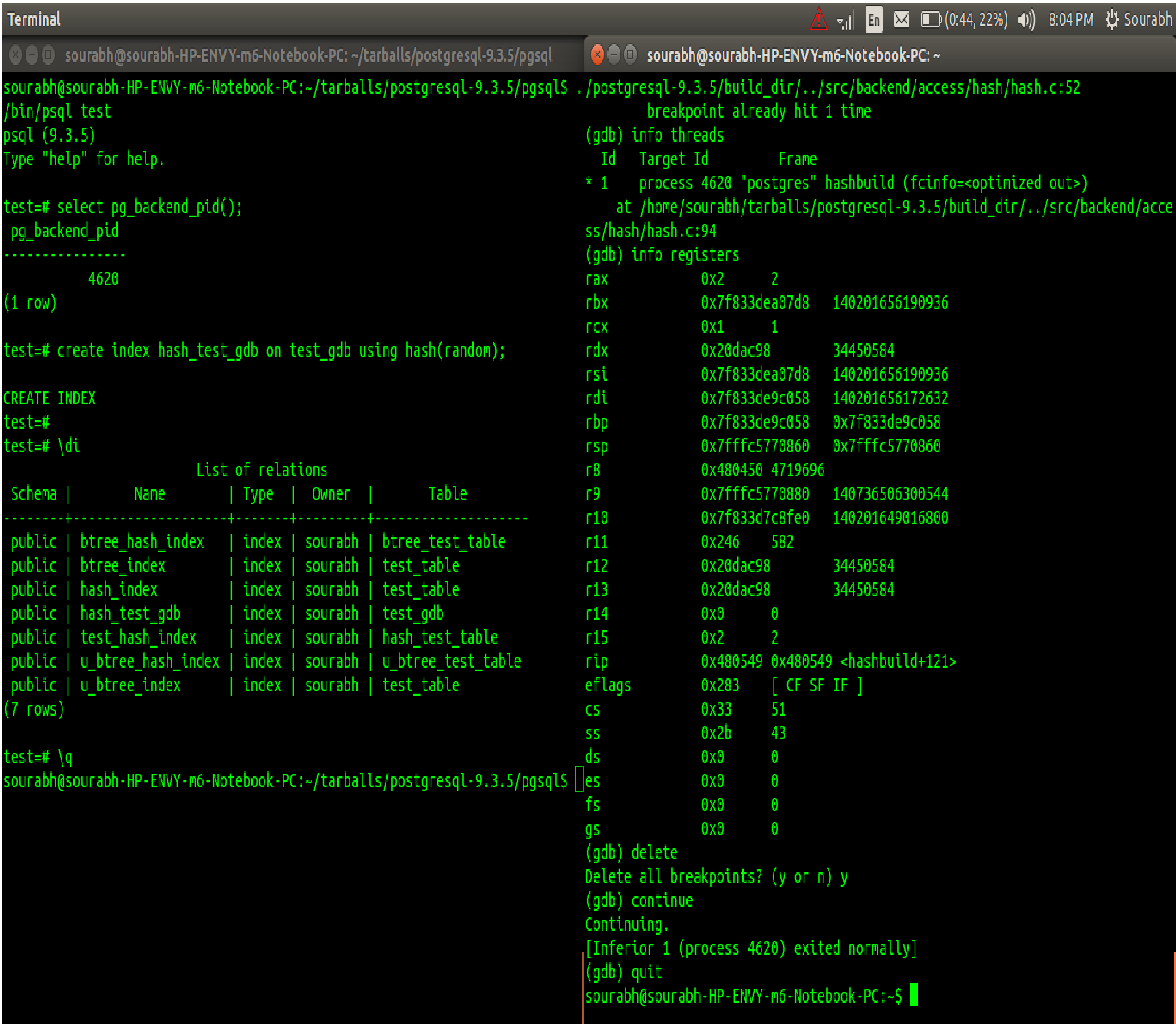
- help
  - List GDB Commands
- run
  - Start program execution from the beginning
- info breakpoints
  - List Breakpoints
- info threads
  - List threads in use
- info registers
  - List Registers in use
- until
  - Continue until you reach a particular line number
- where
  - Shows Line Number and Function you are in
- backtrace
  - Prints Stack backtrace
- delete
  - Deletes all breakpoints
- continue
  - Continue Execution to next breakpoint
- quit
  - Exit GDB Debugger

Usage of some of the commands are shown in the snapshots that follow:


Terminal
En
(0:51, 26%)
7:57 PM
Sourabh
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~/tarballs/postgresql-9.3.5/pgsql
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~

```
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~/tarballs/postgresql-9.3.5/pgsql$ .
/bin/psql test
psql (9.3.5)
Type "help" for help.

test=# select pg_backend_pid();
 pg_backend_pid
----------------
           4620
(1 row)

test=# create index hash_test_gdb on test_gdb using hash(random);
```

```
buildint2vector                  build_trtable
build_join_pathkeys              build_upper_buffer
build_join_rel                   build_wcs_buffer
build_joinrel_restrictlist       build_wcs_upper_buffer
(gdb) print buildstate.indtuples
$2 = 0
(gdb) help
List of classes of commands:

aliases -- Aliases of other commands
breakpoints -- Making program stop at certain points
data -- Examining data
files -- Specifying and examining files
internals -- Maintenance commands
obscure -- Obscure features
running -- Running the program
stack -- Examining the stack
status -- Status inquiries
support -- Support facilities
tracepoints -- Tracing of program execution without stopping the program
user-defined -- User-defined commands

Type "help" followed by a class name for a list of commands in that class.
Type "help all" for the list of all commands.
Type "help" followed by command name for full documentation.
Type "apropos word" to search for commands related to "word".
Command name abbreviations are allowed if unambiguous.
(gdb) info breakpoints
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x00000000004804de in hashbuild
                                                   at /home/sourabh/tarballs
/postgresql-9.3.5/build_dir/../src/backend/access/hash/hash.c:52
        breakpoint already hit 1 time
(gdb) info threads
  Id   Target Id         Frame
* 1    process 4620 "postgres" hashbuild (fcinfo=<optimized out>)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/acce
ss/hash/hash.c:94
(gdb) 
```


Terminal
En
(0:44, 22%)
8:04 PM
Sourabh
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~/tarballs/postgresql-9.3.5/pgsql
sourabh@sourabh-HP-ENVY-m6-Notebook-PC: ~

```
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~/tarballs/postgresql-9.3.5/pgsql$ .
/bin/psql test
psql (9.3.5)
Type "help" for help.

test=# select pg_backend_pid();
 pg_backend_pid
----------------
           4620
(1 row)

test=# create index hash_test_gdb on test_gdb using hash(random);

CREATE INDEX
test=#
test=# \di
                         List of relations
 Schema |        Name        | Type  |  Owner  |       Table
--------+--------------------+-------+---------+-------------------
 public | btree_hash_index   | index | sourabh | btree_test_table
 public | btree_index        | index | sourabh | test_table
 public | hash_index         | index | sourabh | test_table
 public | hash_test_gdb      | index | sourabh | test_gdb
 public | test_hash_index    | index | sourabh | hash_test_table
 public | u_btree_hash_index | index | sourabh | u_btree_test_table
 public | u_btree_index      | index | sourabh | test_table
(7 rows)

test=# \q
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~/tarballs/postgresql-9.3.5/pgsql$ 
```

```
/postgresql-9.3.5/build_dir/../src/backend/access/hash/hash.c:52
        breakpoint already hit 1 time
(gdb) info threads
  Id   Target Id         Frame
* 1    process 4620 "postgres" hashbuild (fcinfo=<optimized out>)
    at /home/sourabh/tarballs/postgresql-9.3.5/build_dir/../src/backend/acce
ss/hash/hash.c:94
(gdb) info registers
rax            0x2      2
rbx            0x7f833dea07d8   140201656190936
rcx            0x1      1
rdx            0x20dac98        34450584
rsi            0x7f833dea07d8   140201656190936
rdi            0x7f833de9c058   140201656172632
rbp            0x7f833de9c058   0x7f833de9c058
rsp            0x7fffc5770860   0x7fffc5770860
r8             0x480450 4719696
r9             0x7fffc5770880   140736506300544
r10            0x7f833d7c8fe0   140201649016800
r11            0x246    582
r12            0x20dac98        34450584
r13            0x20dac98        34450584
r14            0x0      0
r15            0x2      2
rip            0x480549 0x480549 <hashbuild+121>
eflags         0x283    [ CF SF IF ]
cs             0x33     51
ss             0x2b     43
ds             0x0      0
es             0x0      0
fs             0x0      0
gs             0x0      0
(gdb) delete
Delete all breakpoints? (y or n) y
(gdb) continue
Continuing.
[Inferior 1 (process 4620) exited normally]
(gdb) quit
sourabh@sourabh-HP-ENVY-m6-Notebook-PC:~$ 
```

Finally, when we type 'continue', the rest of the program gets executed since we have no more breakpoints and the index gets created.

Type Quit to exit the GDB Debugger.