Hands on Lab
Part 2 – PgPool-II Cluster

DBTech's Virtual Workshop on Distributed, Replicated, Embedded and Mobile Databases (Deliverable #28).

Antonio César Gómez Lora

Dpto. Lenguajes y Ciencias de la Computación

Universidad de Málaga

Introduction

We are going to create a PgPool-II Cluster using three machines. The first one (192.168.192.1) will contain PgPool-II. The second and third machines (192.168.192.2 and 3) will contain the PostgreSQL replicas.

We start now using the previously created and cloned virtual machines. Those machines have Debian (5.0.5/5.0.6), MySQL, PostgreSQL and PgPool-II installed.

Creating the cluster

Start the three machines and login as root on third machine.

We must edit postgresql.conf (Fig. 1).

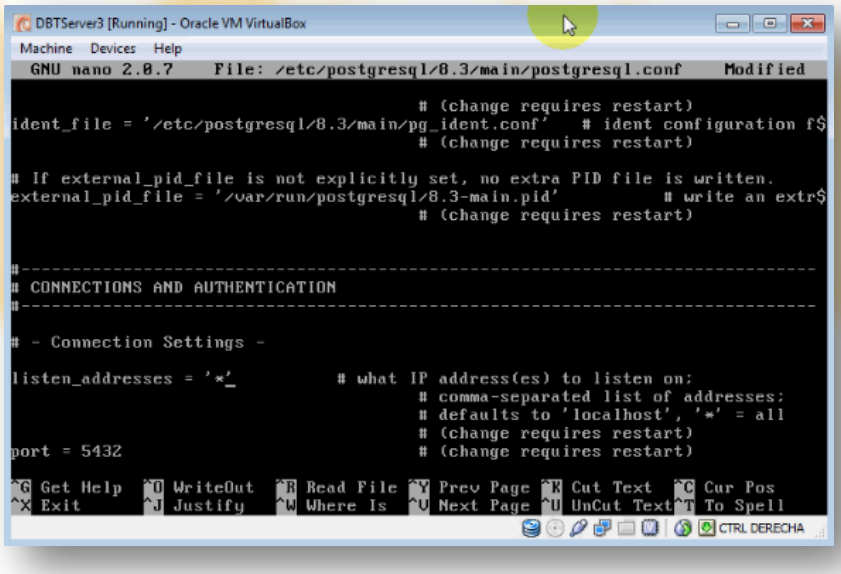

Fig. 1. postgresql.conf

So execute:

nano /etc/postgresql/8.3/main/postgresql.conf

uncomment and change listen_addresses to:

listen_addresses = '*'

Now edit pg_hba.conf (Fig. 2), execute:

nano /etc/postgresql/8.3/main/pg_hba.conf

Add the folowing line at the end of the file:

host all postgres 192.168.192.0/24 trust

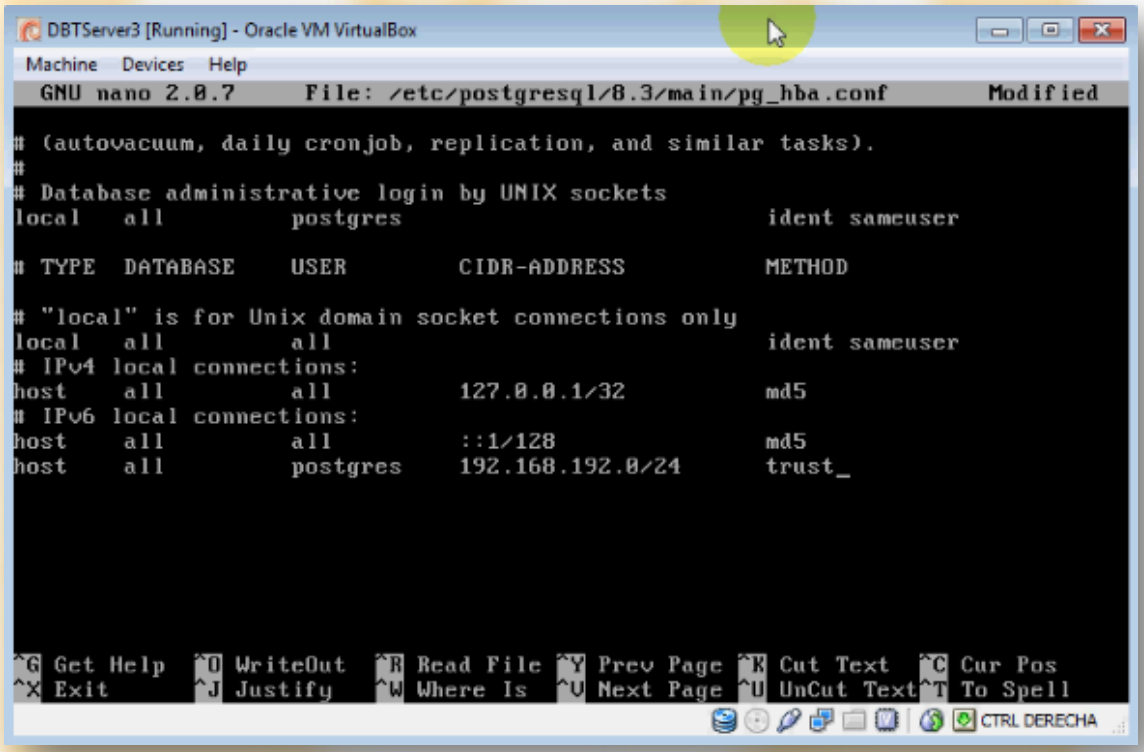

Fig. 2. pg_hba.conf

Repeat the same process on the second machine, editing postgresql.conf and pg_hba.conf.

On first machine edit pgpool.conf (Fig. 3), execute:

nano /etc/pgpool.conf

Change listen_addresses, replication_mode and load_balance_mode to:

listen_adresses = '*'

replication_mode = true

load_balance_mode = true

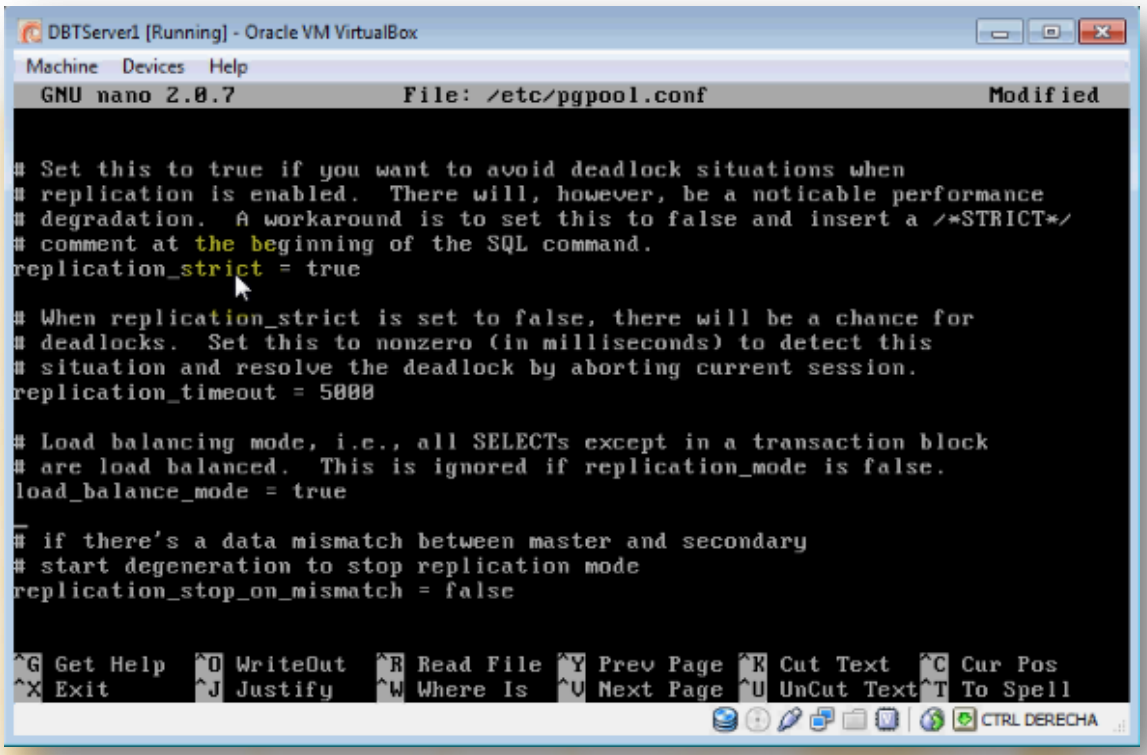

Fig. 3. pgpool.conf

Also include near the end of the file (Fig. 4):

backend_hostname0 = '192.168.192.2'

backend_port0 = 5432

backend_weight0 = 1

backend_hostname1 = '192.168.192.3'

backend_port1 = 5432

backend_weight1 = 1

```
DBTServer1 [Running] - Oracle VM VirtualBox
Machine  Devices  Help
  GNU nano 2.0.7            File: /etc/pgpool.conf                    Modified

# backend_hostname, backend_port, backend_weight
# here are examples
#backend_hostname0 = 'host1'
#backend_port0 = 5432
#backend_weight0 = 1
#backend_hostname1 = 'host2'
#backend_port1 = 5433
#backend_weight1 = 1

backend_hostname0 = '192.168.192.2'
backend_port0 = 5432
backend_weight0 = 1
backend_hostname1 = '192.168.192.3'
backend_port1 = 5432
backend_weight1 = 1_

# - HBA -

# If true, use pool_hba.conf for client authentication. In pgpool-II
# 1.1, the default value is false. The default value will be true in

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
CTRL DERECHA
```

Fig. 4. backends on pgpool.conf

Now its time to restart postgresql service on all machines (Fig. 5), execute on each machine:

/etc/init.d/postgresql-8.3 restart

```
DBTServer2 [Running] - Oracle VM VirtualBox
Machine  Devices  Help
# Database administrative login by UNIX sockets
local   all         postgres                        ident sameuser

# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD

# "local" is for Unix domain socket connections only
local   all         all                               ident sameuser
# IPv4 local connections:
host    all         all         127.0.0.1/32          md5
# IPv6 local connections:
host    all         all         ::1/128               md5
host    all         postgres    192.168.192.0/24      trust

[ Wrote 84 lines ]

debian:~# /etc/init.d/postgresql-8.3 restart
Restarting PostgreSQL 8.3 database server: main.
debian:~# _
CTRL DERECHA
```

Fig. 5. Restarting postgreSQL

On first machine also restart pgpool service (Fig. 6), execute:

/etc/init.d/pgpool2 restart

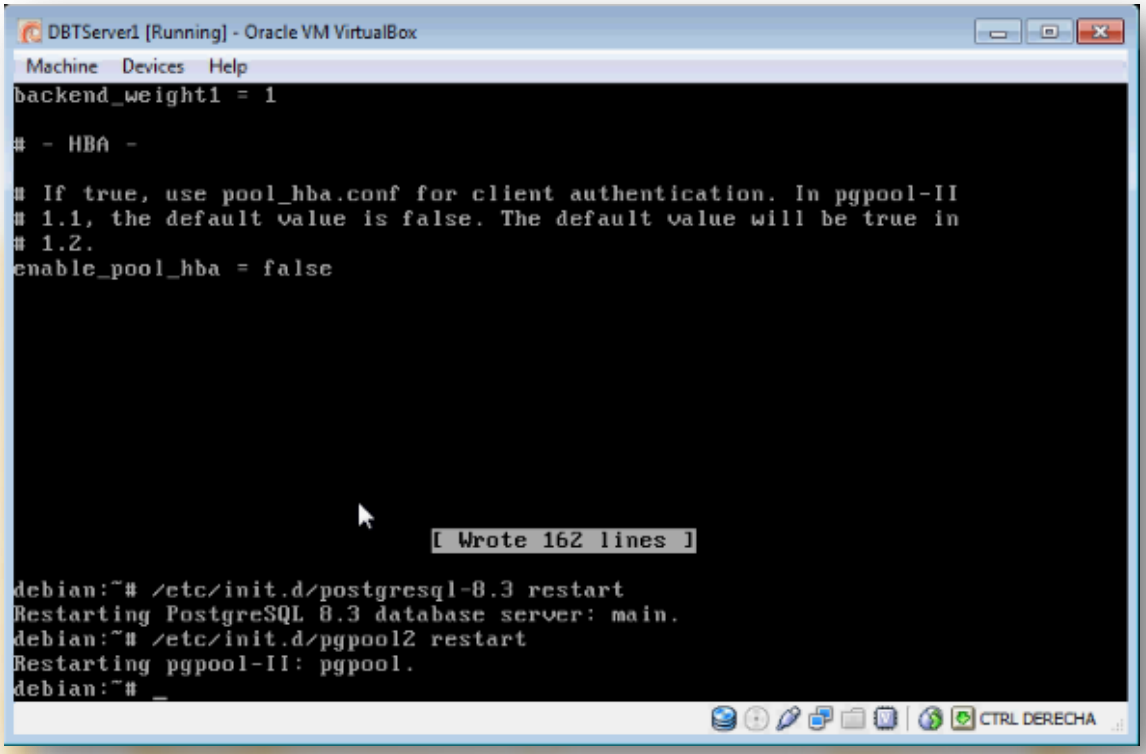

Fig. 6. Restarting pgpool-II

Testing the cluster

Verify that PostgreSQL is clean (Fig. 7), execute on second and third machine:

su postgres

psql -l

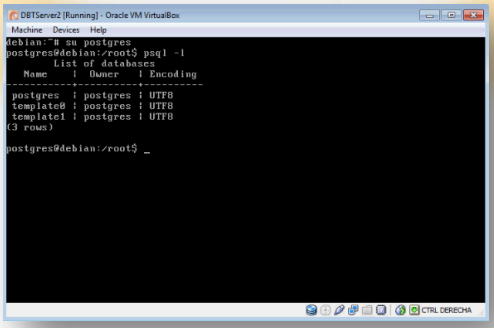

Fig. 7. Clean postgreSQL

On first machine execute:

su postgres

psql -l -p 5433

create a database using pgpool2 (Fig. 8),

execute:

createdb -p 5433 pgptest

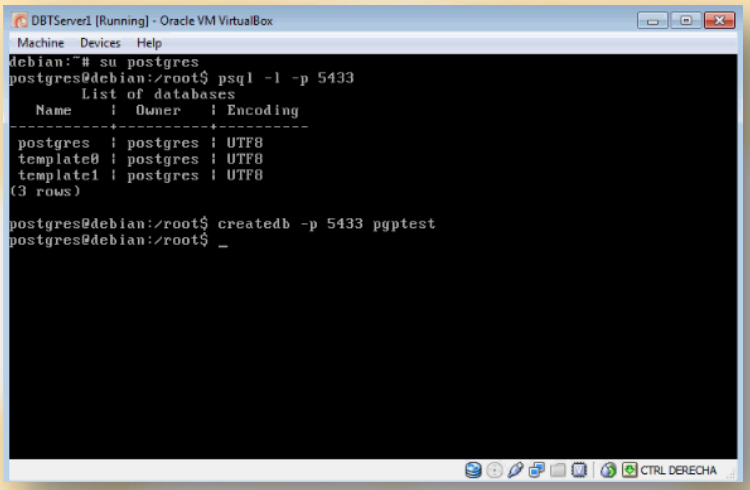

Fig. 8. Create pgptest

Let's see if pgptest exists on storage nodes (Fig. 9), second and third machines, on them execute:

psql -l

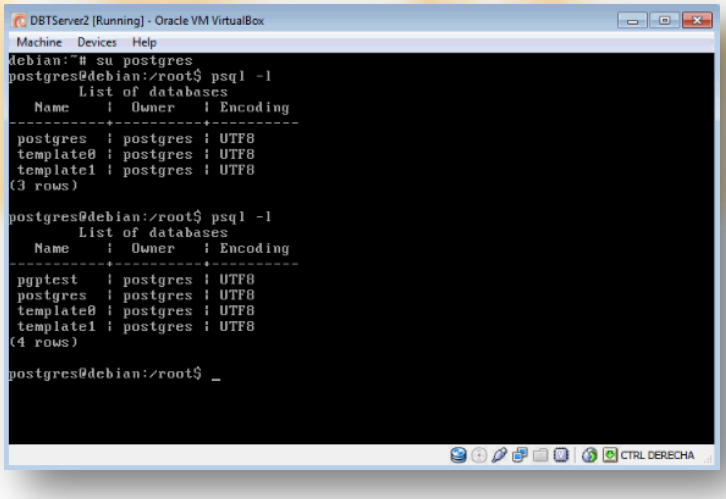

Fig. 9. Database replicated

Now let's monitorize second and third machine activity using htop (Fig. 10) or top.

If htop is not installed you can install it by executing

```
sudo apt-get install htop
```

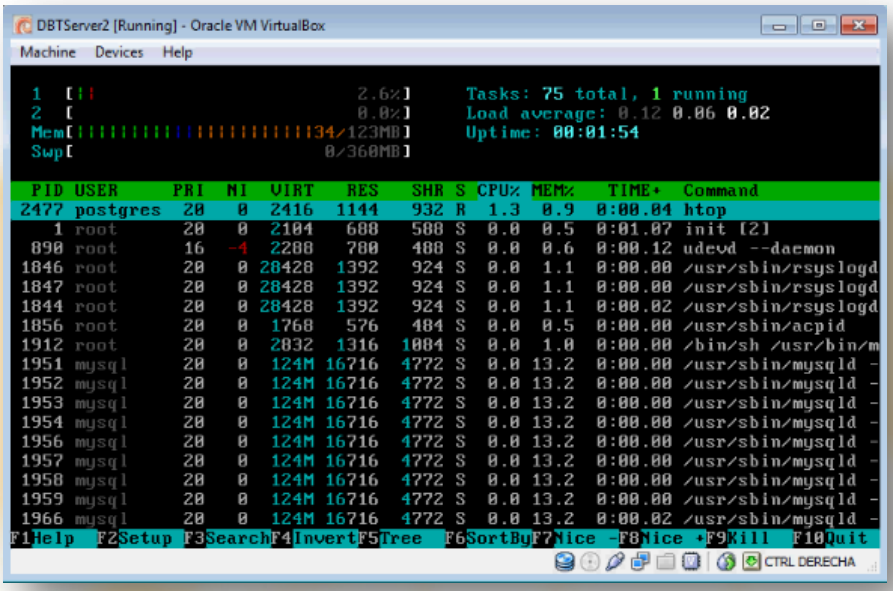

Fig. 10. Htop

Let's start psql interpreter connecting to pgpool on first machine, execute:

```
psql -p 5433 pgptest
```

Now let's create a simple table, insert a value and duplicate the table contents, so we get a little big table (Fig. 11).

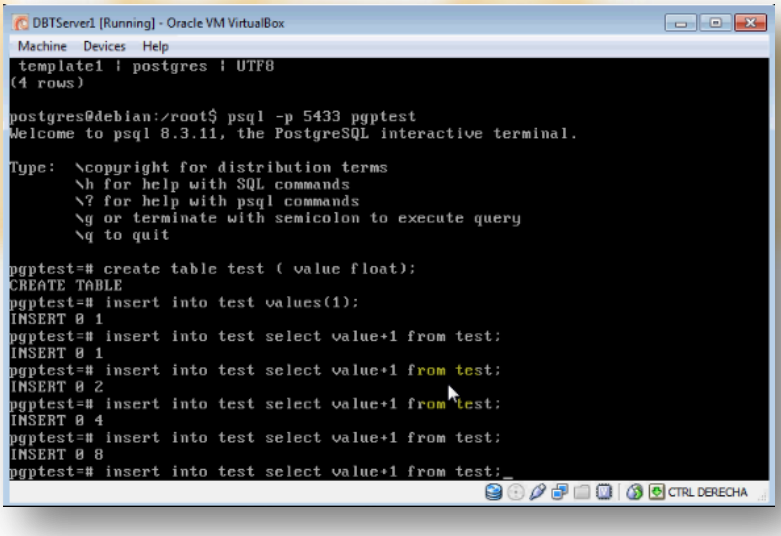

Fig. 11. Test table

Now we create two "heavy" views (Fig. 12), with no meaning.

```
pgptest=# insert into test select value+1 from test;
INSERT 0 128
pgptest=# insert into test select value+1 from test;
INSERT 0 256
pgptest=# insert into test select value+1 from test;
INSERT 0 512
pgptest=# insert into test select value+1 from test;
INSERT 0 1024
pgptest=# insert into test select value+1 from test;
INSERT 0 2048
pgptest=# insert into test select value+1 from test;
INSERT 0 4096
pgptest=# select count(*) from test;
 count
-------
  8192
(1 row)

pgptest=# create view v1 as select count(*) from test t1, test t2 where t1.value
 < t2.value;
CREATE VIEW
pgptest=# create view v2 as select count(*) from test t1, test t2 where t1.value
 > 2*t2.value;
CREATE VIEW
pgptest=# _
```

Fig. 12. Little heavy views

Now open a second terminal session on first machine (by pressing ALT+F2), login, change to user postgres and start the psql interpreter (Fig. 13), execute:

su postgres

psql -p 5433 pgptest

```
Debian GNU/Linux 5.0 debian tty2

debian login: root
Password:
Last login: Fri Sep 10 12:10:19 CEST 2010 on tty1
Linux debian 2.6.26-2-686 #1 SMP Thu Aug 19 03:44:10 UTC 2010 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
debian:~# su postgres
postgres@debian:/root$ psql -p 5433 pgptest
Welcome to psql 8.3.11, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit

pgptest=# select * from v2;_
```

Fig. 13. Second session

Using ALT+F1 and ALT+F2 alternate between sessions (Fig. 14), and prepare but do not execute on first session:

```
select * from v1;
```

And on second one:

```
select * from v2;
```

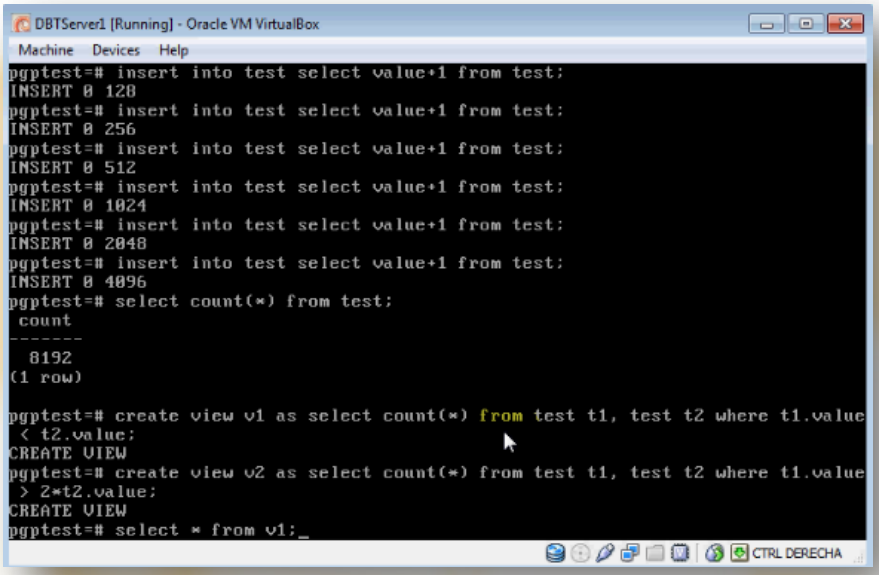

Fig. 14. First session

When ready press ENTER and quickly alternate to other session and press ENTER again. You will see activity on both machines (Fig. 15). In this case second session's query was launched first. Pgpool sends the query to the first free database (the one on the left), and when first session launch its query Pgpool sends it to the second datatabase (the one on the right), beacuse the first one was working. If you launch first a query and when it finishs the second one you will see that they are executed by the first database.

Fig. 15. Processing

Now let's see that both databases contains the same table created through pgpool2 (Fig. 16).

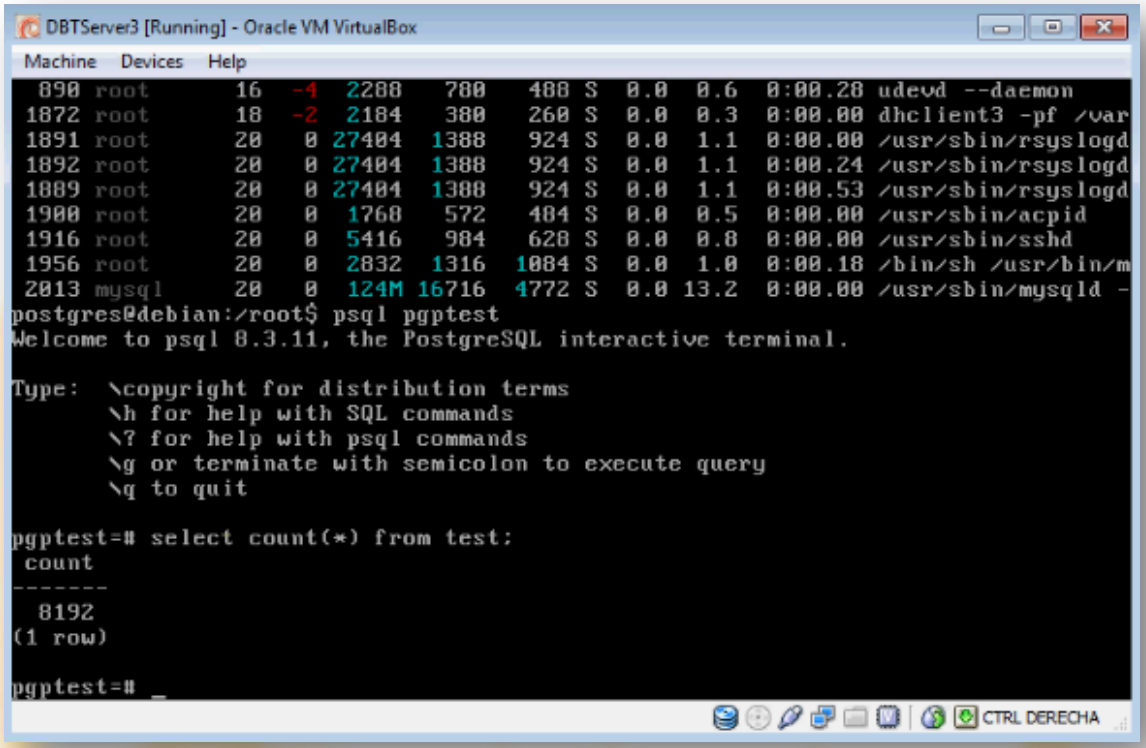

Fig. 16. Replicated

Proposed exercises

Repeat the proposed exercises 1 and 2 for MySQL Cluster, using now PgPool-II.

1. If you click on VirtualBox network icon (on the bottom of each virtual machine status bar) you can view network interfaces and disconnect and reconnect whenever you want. Disconnect a virtual machine and try to determine what happens with replicated data if both replicas continue modifying the replicated data. First disconnect cluster manager, and after testing it, disconnect the first storage node and finally the second storage node. If cluster lost its integrity regenerate the replicated table. If a data node cannot reconnect then restart the services properly (as it has been done in this document).
2. Start transactions on storage nodes and try to analyze what kind of transactional model it follows.

And now:

3. Consulting the PgPool manuals try to configure PgPool to perform parallel queries.

Figure Index

Fig. 1. postgresql.conf 1
Fig. 2. pg_hba.conf 2
Fig. 3. pgpool.conf 3
Fig. 4. backends on pgpool.conf 4
Fig. 5. Restarting postgreSQL 4
Fig. 6. Restarting pgpool-II 5
Fig. 7. Clean postgreSQL 5
Fig. 8. Create pgptest 6
Fig. 9. Database replicated 6
Fig. 10. Htop 7
Fig. 11. Test table 7
Fig. 12. Little heavy views 8
Fig. 13. Second session 8
Fig. 14. First session 9
Fig. 15. Processing 9
Fig. 16. Replicated 10