

# **Faculdade de Tecnologia Senac Goiás**

MATHEUS OLIVEIRA RODRIGUES

## **PROJETO INTEGRADOR**

Elias Batista Ferreira

Goiânia  
**2017**

## O que é replicação?

Replicação de banco de dados consiste na replicação (cópia) de um banco de dados ou atualização para um ou mais *sites* (outros bancos de dados) podem estar em outra localização lógica ou geográfica de forma que todos tenham a mesma informação permitindo a alta disponibilidade, confiabilidade, desempenho e/ou balanceamento de carga.

## Tipos de replicações?

Existem os tipos síncronas e assíncronas.

No tipo síncrona, a replicação da ação é feita instantaneamente. Se alguma cópia do banco é alterada, essa alteração será imediatamente aplicada a todos os outros bancos dentro da transação. A replicação síncrona é apropriada em aplicações comerciais onde é exigido um nível de atualização muito preciso em todos os servidores envolvidos.

No tipo assíncrona, o replicador monta um histórico das ações a serem replicadas e em um determinado momento é feita a replicação entre as bases de dados relacionadas. A alteração será propagada e aplicada para outra base em um segundo passo, dentro de uma transação separada. Esta poderá ocorrer em segundos, minutos, horas ou até dias depois, dependendo da configuração.

## Vantagens e desvantagens:

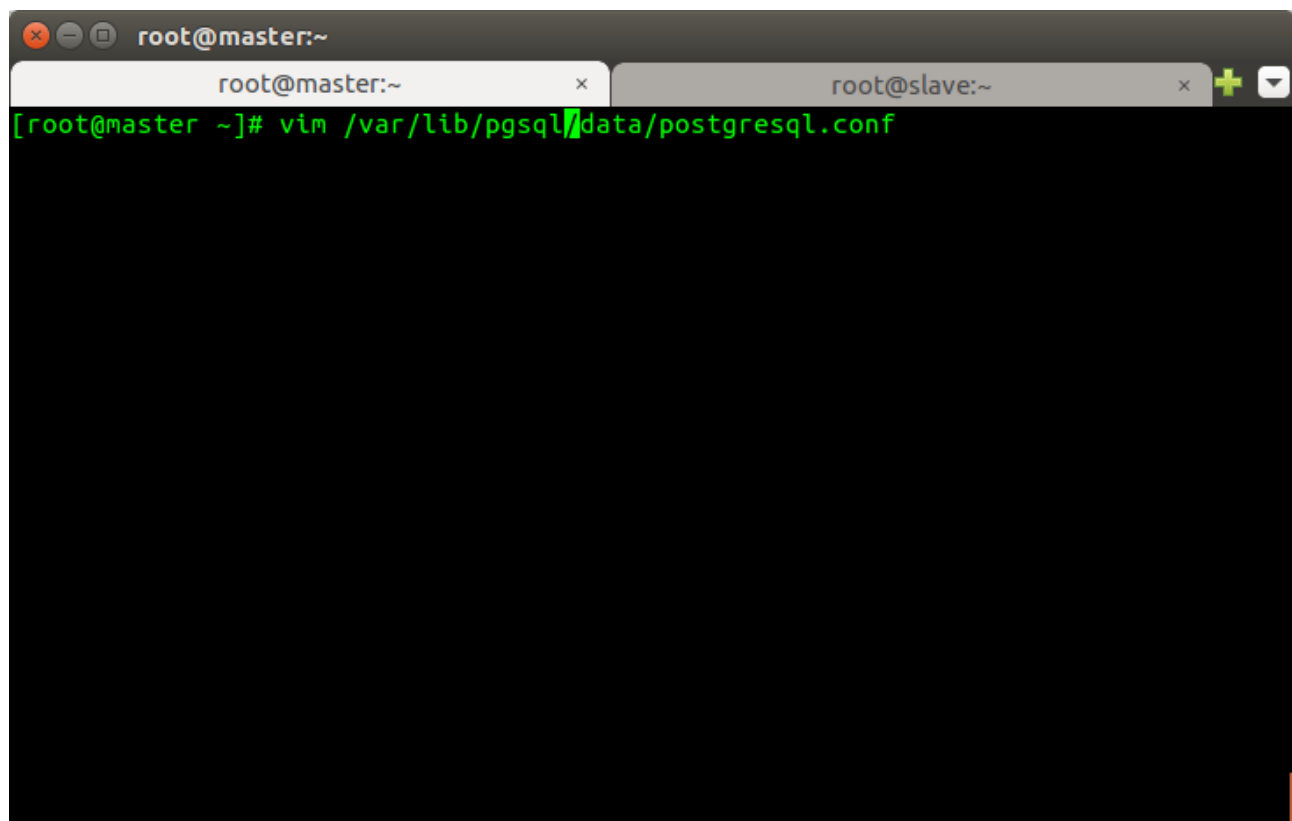
Disponibilidade, se um *site* falhar outro pode assumir o fornecimento e armazenamento de informações;

Paralelismo aumentado, se acontece um maior volume de consulta em um *site* e este possui outros *sites* replicados, as consultas podem ser direcionadas aos mais próximos que possuem os mesmos dados que o principal.

Maior sobrecarga na atualização, em um sistema onde a replicação acontece é necessário garantir que todos os *sites* estejam sincronizados e ter consistência nos dados, caso contrário, pode acontecer uma computação errônea o que é inaceitável. Dessa forma sempre que um *site* é atualizado, esses novos dados devem ser propagados para os demais *sites* isso pode gerar uma sobrecarga sendo necessário de uma grande quantidade de banda.

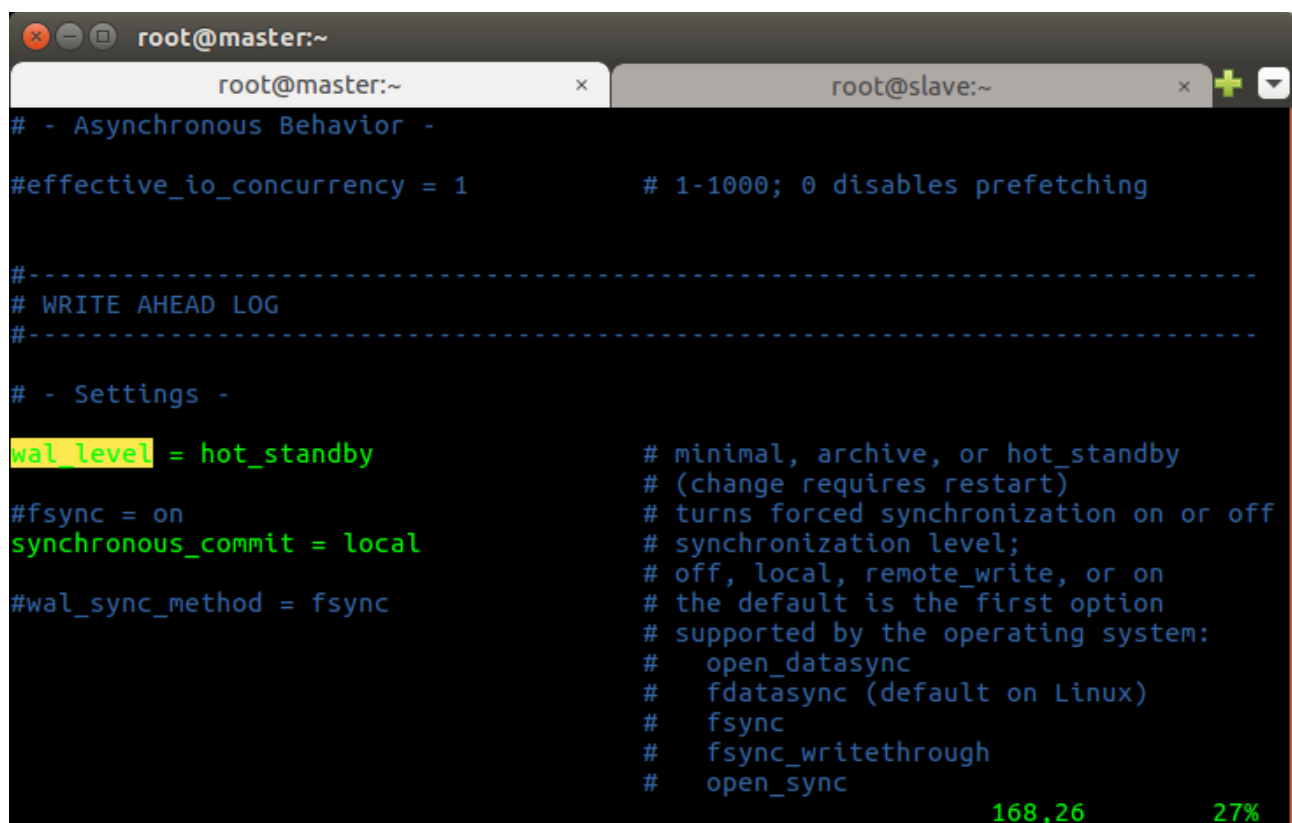
## Tutorial de configuração do servidor para realizar replicação

Sistema usado CentOS 7 Minimal.



A terminal window with two tabs: 'root@master:~' and 'root@slave:~'. The 'root@master:~' tab is active. The command `[root@master ~]# vim /var/lib/pgsql/data/postgresql.conf` is entered at the prompt.

```
root@master:~  
[root@master ~]# vim /var/lib/pgsql/data/postgresql.conf
```



A terminal window with two tabs: 'root@master:~' and 'root@slave:~'. The 'root@master:~' tab is active. The contents of the `postgresql.conf` file are displayed, showing sections for asynchronous behavior, write ahead log, and settings. The `wal_level = hot_standby` and `synchronous_commit = local` lines are highlighted in green. The bottom right corner shows '168,26' and '27%'.

```
root@master:~  
# - Asynchronous Behavior -  
#effective_io_concurrency = 1          # 1-1000; 0 disables prefetching  
  
#-----  
# WRITE AHEAD LOG  
#-----  
  
# - Settings -  
wal_level = hot_standby               # minimal, archive, or hot_standby  
#                                     # (change requires restart)  
#fsync = on                           # turns forced synchronization on or off  
synchronous_commit = local           # synchronization level;  
#                                     # off, local, remote_write, or on  
#wal_sync_method = fsync               # the default is the first option  
#                                     # supported by the operating system:  
#   open_datasync  
#   fdatasync (default on Linux)  
#   fsync  
#   fsync_writethrough  
#   open_sync  
  
168,26                               27%
```

```
root@master:~
root@master:~ x root@slave:~ x + v
#commit_siblings = 5 # range 1-1000
# - Checkpoints -
#checkpoint_segments = 3 # in logfile segments, min 1, 16MB each
#checkpoint_timeout = 5min # range 30s-1h
#checkpoint_completion_target = 0.5 # checkpoint target duration, 0.0 - 1.0
#checkpoint_warning = 30s # 0 disables
# - Archiving -
archive_mode = on # allows archiving to be done
# (change requires restart)
archive_command = 'cp %p /var/lib/pgsql/archive/%f' # command to use
to archive a logfile segment
# placeholders: %p = path of file to archive
# %f = file name only
# e.g. 'test ! -f /mnt/server/archivedir/%f && c
p %p /mnt/server/archivedir/%f'
#archive_timeout = 0 # force a logfile segment switch after this
# number of seconds; 0 disables
196,50 32%
```

```
root@master:~
root@master:~ x root@slave:~ x + v
# - Sending Server(s) -
# Set these on the master and on any standby that will send replication data.
max_wal_senders = 2 # max number of walsender processes
# (change requires restart)
wal_keep_segments = 10 # in logfile segments, 16MB each; 0 disables
#replication_timeout = 60s # in milliseconds; 0 disables
# - Master Server -
# These settings are ignored on a standby server.
synchronous_standby_names = 'slave' # standby servers that provide sync rep
# comma-separated list of application_name
# from standby(s); '*' = all
#vacuum_defer_cleanup_age = 0 # number of xacts by which cleanup is delayed
# - Standby Servers -
# These settings are ignored on a master server.
#hot_standby = off # "on" allows queries during recovery
214,22 37%
```

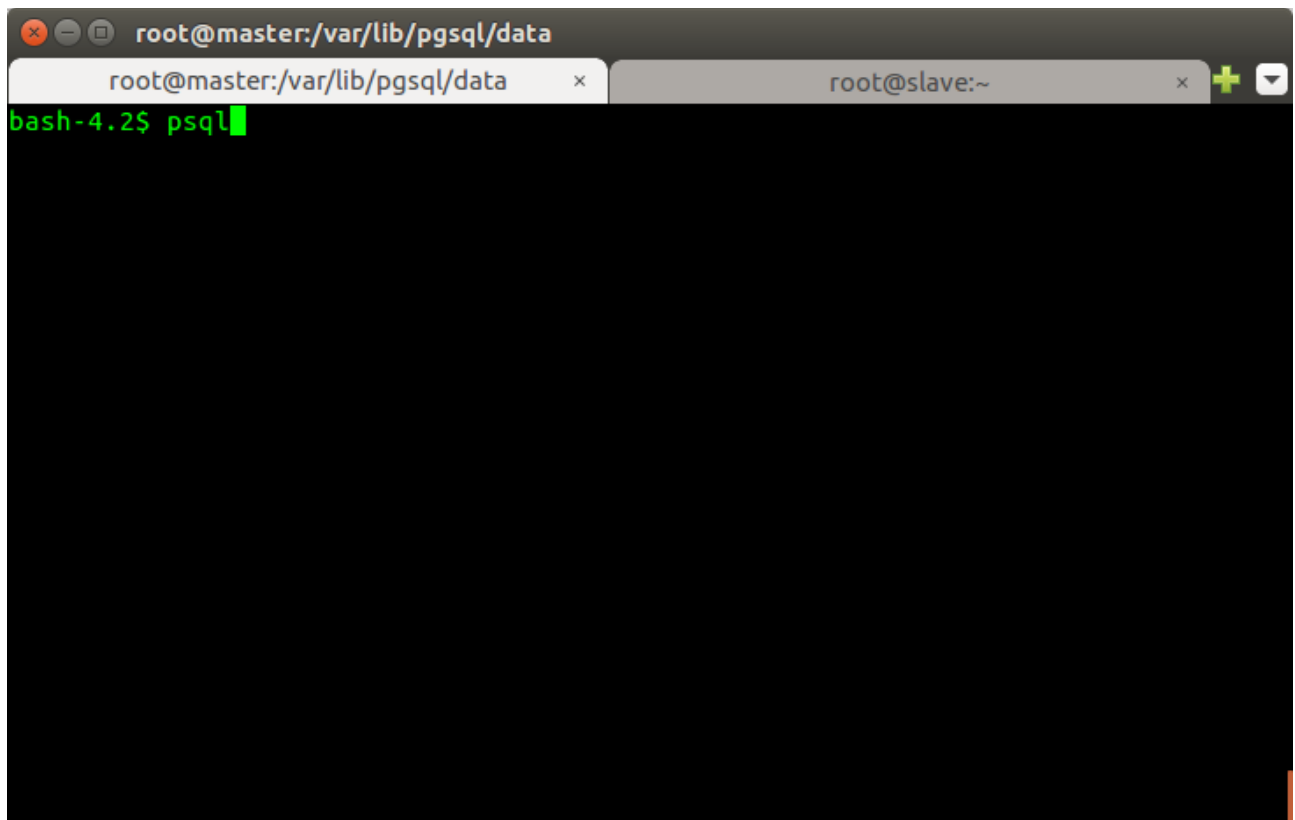
```
root@master:~
root@slave:~
#external_pid_file = '' # write an extra PID file
# (change requires restart)

#-----
# CONNECTIONS AND AUTHENTICATION
#-----

# - Connection Settings -

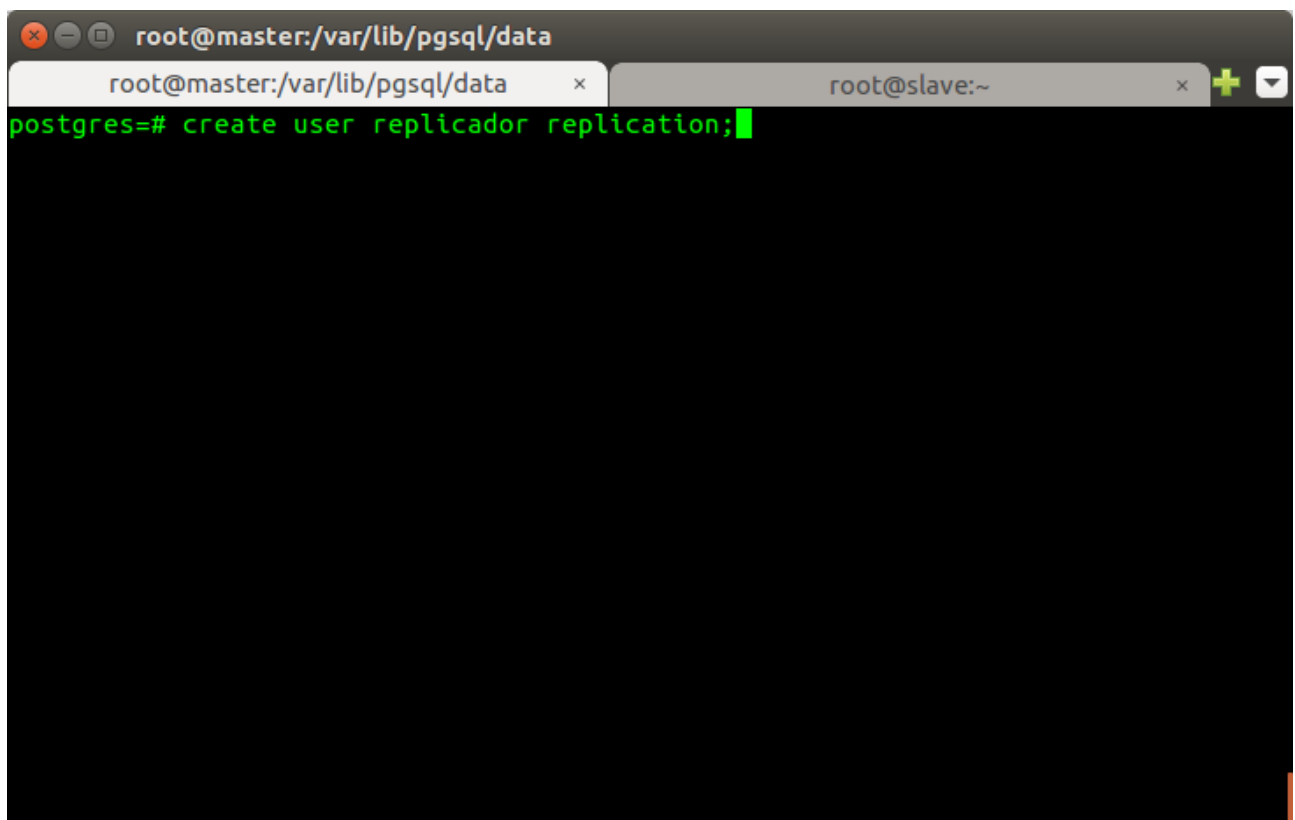
listen_addresses = 'localhost, 192.168.0.10, 192.168.10.12' # what I
P address(es) to listen on;
# comma-separated list of addresses;
# defaults to 'localhost'; use '*' for a
ll
# (change requires restart)
#port = 5432 # (change requires restart)
# Note: In RHEL/Fedora installations, you can't set the port number here;
# adjust it in the service file instead.
max_connections = 100 # (change requires restart)
# Note: Increasing max_connections costs ~400 bytes of shared memory per
# connection slot, plus lock space (see max_locks_per_transaction).
#superuser_reserved_connections = 3 # (change requires restart)
"/var/lib/pgsql/data/postgresql.conf" 577L, 19900C 59,1 8%
```

```
root@master:/var/lib/pgsql/data
root@slave:~
[root@master data]# su postgres
```



A terminal window with a dark background. The title bar shows 'root@master:/var/lib/pgsql/data'. There are two tabs: 'root@master:/var/lib/pgsql/data' and 'root@slave:~'. The prompt is 'bash-4.2\$' and the command 'psql' has been entered, with a green cursor at the end.

```
root@master:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:~
bash-4.2$ psql
```



A terminal window with a dark background. The title bar shows 'root@master:/var/lib/pgsql/data'. There are two tabs: 'root@master:/var/lib/pgsql/data' and 'root@slave:~'. The prompt is 'postgres=#' and the command 'create user replicador replication;' has been entered, with a green cursor at the end.

```
root@master:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:~
postgres=# create user replicador replication;
```

```
root@master:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data x root@slave:~ x + v
[root@master data]# vim pg_hba.conf
```

```
root@master:~
root@master:~ x root@slave:/var/lib/pgsql/data x + v
#
# If you want to allow non-local connections, you need to add more
# "host" records. In that case you will also need to make PostgreSQL
# listen on a non-local interface via the listen_addresses
# configuration parameter, or via the -i or -h command line switches.
#
# TYPE DATABASE USER ADDRESS METHOD
# "local" is for Unix domain socket connections only
local all all peer
# IPv4 local connections:
host all all 127.0.0.1/32 ident
# IPv6 local connections:
host all all ::1/128 ident
# Allow replication connections from localhost, by a user with the
# replication privilege.
#local replication postgres peer
#host replication postgres 127.0.0.1/32 ident
#host replication postgres ::1/128 ident
host replication replicador 192.168.0.12/27 trust
host replication replicador 192.168.0.10/27 trust
"/var/lib/pgsql/data/pg_hba.conf" 91L, 4374C 91,1 Fin
```

```
root@master:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data x root@slave:~ x + v
[root@master data]# service postgresql restart
Redirecting to /bin/systemctl restart postgresql.service
[root@master data]#
```

```
root@slave:~
root@master:/var/lib/pgsql/data x root@slave:~ x + v
[root@slave ~]# service postgresql stop
```



```
root@slave:~  
root@master:/var/lib/pgsql/data x root@slave:~ x  
[root@slave ~]# cd /var/lib/pgsql/data/
```

```
root@slave:/var/lib/pgsql/data  
root@master:/var/lib/pgsql/data x root@slave:/var/lib/pgsql/data x  
[root@slave ~]# cd /var/lib/pgsql/data/  
[root@slave data]# ls  
base          pg_ident.conf  pg_serial      pg_tblspc      postgresql.conf  
global        pg_log         pg_snapshots   pg_twophase  
pg_clog       pg_multixact   pg_stat_tmp    PG_VERSION  
pg_hba.conf   pg_notify      pg_subtrans    pg_xlog  
[root@slave data]# rm -rf *  
[root@slave data]# ls  
[root@slave data]#
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
[root@slave data]# su postgres
bash-4.2$
```

```
root@slave:/var/lib/pgsql/data
root@master:~
root@slave:/var/lib/pgsql/data
[root@slave data]# su - postgres
Ultimo login:Sex Jun  9 00:32:06 BRT 2017em pts/0
-bash-4.2$ pg_basebackup -h 192.168.0.10 -U replicador -D /var/lib/pgsql/data -P
--xlog
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
[root@slave data]# vim postgresql.conf
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
# These settings are ignored on a standby server.
synchronous_standby_names = 'slave'      # standby servers that provide sync rep
                                         # comma-separated list of application_name
                                         # from standby(s); '*' = all
#vacuum_defer_cleanup_age = 0           # number of xacts by which cleanup is delayed
# - Standby Servers -
# These settings are ignored on a master server.
hot_standby = on                        # "on" allows queries during recovery
                                         # (change requires restart)
#max_standby_archive_delay = 30s        # max delay before canceling queries
                                         # when reading WAL from archive;
                                         # -1 allows indefinite delay
#max_standby_streaming_delay = 30s      # max delay before canceling queries
                                         # when reading streaming WAL;
                                         # -1 allows indefinite delay
#wal_receiver_status_interval = 10s     # send replies at least this often
                                         # 0 disables
#hot_standby_feedback = off            # send info from standby to prevent
                                         # query conflicts
230,16                                39%
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
[root@slave data]# cp /usr/share/pgsql/recovery.conf.sample /var/lib/pgsql/data/
recovery.conf
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
[root@slave data]# vim recovery.conf
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
# It is important that the command return nonzero exit status on failure.
# The command *will* be asked for log files that are not present in the
# archive; it must return nonzero when so asked.
#
# NOTE that the basename of %p will be different from %f; do not
# expect them to be interchangeable.
#
restore_command = 'scp 192.168.0.10:/var/lib/pgsql/archive/%f %p'
# e.g. 'cp /mnt/server/archivedir/%f %p'
#
# archive_cleanup_command
#
# specifies an optional shell command to execute at every restartpoint.
# This can be useful for cleaning up the archive of a standby server.
#
#archive_cleanup_command = ''
#
# recovery_end_command
#
# specifies an optional shell command to execute at completion of recovery.
# This can be useful for cleaning up after the restore_command.
##
58,1 32%
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
## STANDBY SERVER PARAMETERS
#-----
#
# standby_mode
#
# When standby_mode is enabled, the PostgreSQL server will work as a
# standby. It will continuously wait for the additional XLOG records, using
# restore_command and/or primary_conninfo.
#
standby_mode = on
#
# primary_conninfo
#
# If set, the PostgreSQL server will try to connect to the primary using this
# connection string and receive XLOG records continuously.
#
primary_conninfo = 'host=192.168.0.10 port=5432 user=replicador application_name
=slave' # e.g. 'host=localhost port=5432'
#
#
# By default, a standby server keeps restoring XLOG records from the
# primary indefinitely. If you want to stop the standby mode, finish recovery
# and open the system in read/write mode, specify path to a trigger file.
99,1 89%
```

```
root@slave:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
[root@slave data]# service postgresql restart
```

```
root@master:/var/lib/pgsql/data
root@master:/var/lib/pgsql/data
root@slave:/var/lib/pgsql/data
postgres=# create database teste;
CREATE DATABASE
postgres=#
```

## **Bibliografia:**

<http://eulerto.blogspot.com.br/2010/11/replicacao-no-postgresql.html>

<http://www.devmedia.com.br/introducao-a-replicacao-e-alta-disponibilidade-no-postgresql/6140>

Sistema de Banco de Dados Abraham Silberschatz

<https://www.youtube.com/watch?v=J2VqnkToPzI>