PostgreSQL Cluster

Recovery Guide

Version 1.0.0

Contents

Copyright Notice

Document Revision History

April 29, 2019

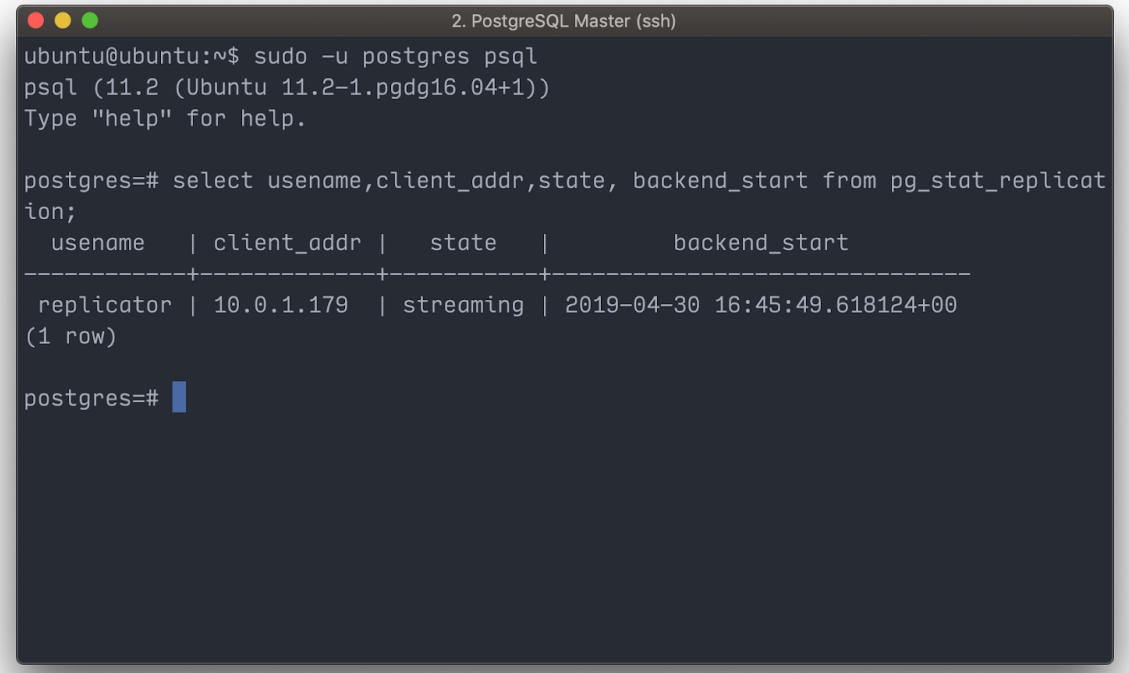
- Initial release of documentation

Replication status

Verify Primary

Check the replication status on the **primary** server by querying the **replication stats.** Run the following commands to check the primary server replication status

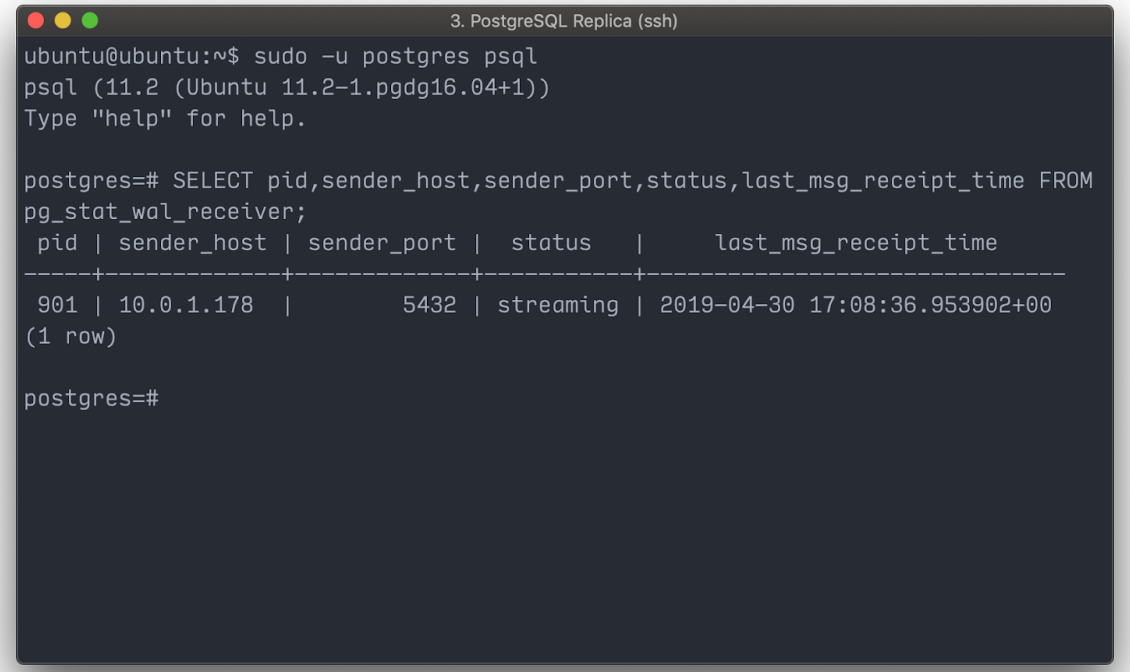
```
# Connect to the postgresql shell on primary server
sudo -u postgres psql

# Query for pg_stat_replication
select usename,client_addr,state,backend_start from
pg_stat_replication;
```



Verify Replica

Check the replication status on the **replica** server by querying the **write ahead logs (wal) receiver stats.** Run the following commands to check the replica server replication status

```
# Connect to postgres on replica server
sudo -u postgres psql

# Query pg_stat_wal_receiver to check if the primary server is sending
WAL files
SELECT pid,sender_host,sender_port,status,last_msg_receipt_time FROM
pg_stat_wal_receiver;
```



Promote Replica as Primary

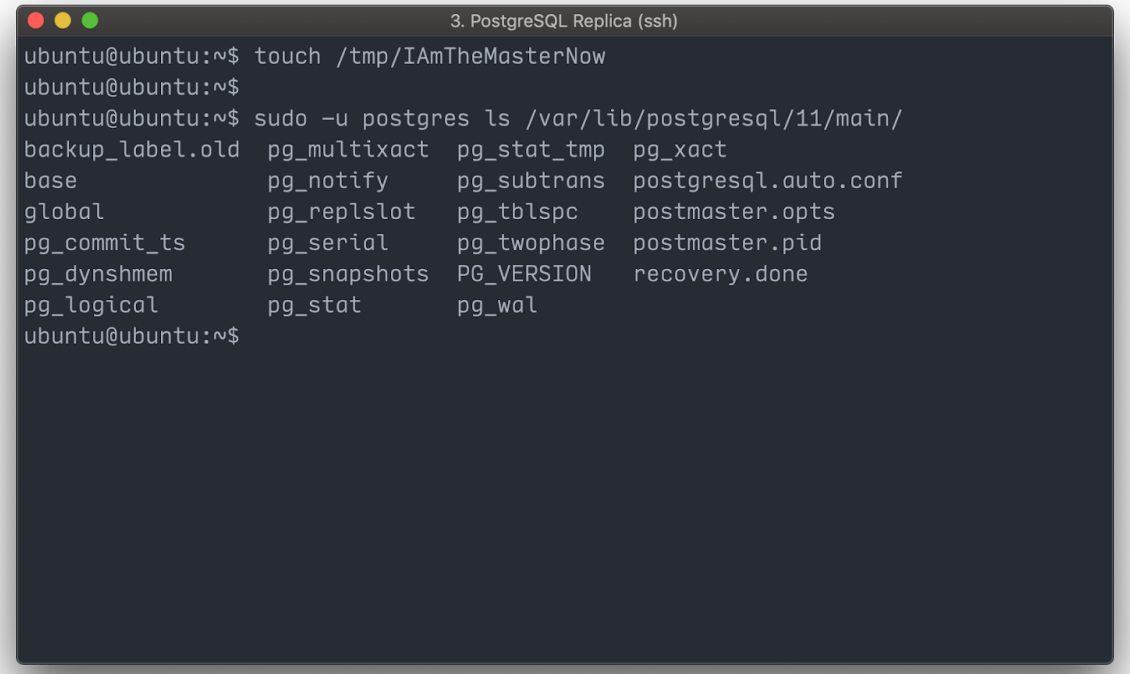
Promote the **replica** server as **primary** by creating a trigger file specified in **recovery.conf** located at `/var/lib/postgresql/11/main/recovery.conf` on **replica** server.

```
# Read the location of the trigger file specified in recovery.conf
sudo -u postgres cat /var/lib/postgresql/11/main/recovery.conf | head
-n 10 | grep trigger_file
```

Create an empty file located at the **trigger_file** path specified in the previous step. For example if the trigger_file is configured at '**`/tmp/IAmTheMasterNow`**' then create an empty file at that location.

Check if the promotion of replica to primary is successful by checking if the **recovery.conf** file located at `/var/lib/postgresql/11/main/` is renamed to **recovery.done**.

```
# Create an empty file at /tmp/IAmTheMasterNow
touch /tmp/IAmTheMasterNow

# Verify promotion of replica server is successful
sudo -u postgres ls /var/lib/postgresql/11/main/
```



After replica server is successfully promoted as primary server, stop the postgres process on the **old primary** server to avoid data loss.

```
# Disable postgres service
sudo systemctl disable pg

# Stop postgres service
sudo systemctl stop pg
```

2. PostgreSQL Master (ssh)

```
ubuntu@ubuntu:~$ sudo systemctl disable pg
Removed symlink /etc/systemd/system/multi-user.target.wants/pg.service.
ubuntu@ubuntu:~$
ubuntu@ubuntu:~$ sudo systemctl stop pg
ubuntu@ubuntu:~$
ubuntu@ubuntu:~$ sudo systemctl status pg
● pg.service - PostgreSQL database server
   Loaded: loaded (/etc/systemd/system/pg.service; disabled; vendor preset: enab
   Active: inactive (dead)
     Docs: man:postgres(1)

Apr 30 13:23:16 ubuntu systemd[1]: Stopping PostgreSQL database server...
Apr 30 13:23:16 ubuntu postgres[665]: 2019-04-30 17:23:16.245 UTC [665] LOG:  re
Apr 30 13:23:16 ubuntu postgres[665]: 2019-04-30 17:23:16.246 UTC [665] LOG:  ab
Apr 30 13:23:16 ubuntu postgres[665]: 2019-04-30 17:23:16.248 UTC [665] LOG:  ba
Apr 30 13:23:16 ubuntu postgres[665]: 2019-04-30 17:23:16.249 UTC [767] LOG:  sh
Apr 30 13:23:16 ubuntu postgres[665]: 2019-04-30 17:23:16.331 UTC [1175] replica
Apr 30 13:23:16 ubuntu postgres[665]: 2019-04-30 17:23:16.332 UTC [1176] replica
Apr 30 13:23:16 ubuntu systemd[1]: Stopped PostgreSQL database server.
Apr 30 13:30:48 ubuntu systemd[1]: Stopped PostgreSQL database server.
```

Disable Replication Mode

If the primary server fails and the replica server is promoted as the new primary server, the old primary server has to be disabled to avoid data conflicts resulting in data loss.

After disabling the old primary server and If a new server is not readily available to deploy as new replica server, then the replication has to be disabled on the new primary server until a new replica is available.

```
# Open the postgresql.conf
sudo vim /etc/postgresql/11/main/postgresql.conf


# Disable Write Ahead Log (WAL) settings from line 50

      # wal_level = replica
      # fsync = on
      # archive_mode = on
      # archive_command = 'rsync -a %p
      postgres@REPLICA_IP_ADDRESS:~/master_wal/%f'
      # full_page_writes = on
      # max_wal_size = 1GB
      # min_wal_size = 80MB

# Disable Replication settings from line 63

      # max_wal_senders = 4
      # wal_keep_segments = 0

# Disable HotStandby on line 81

      # hot_standby = on
```

3. PostgreSQL Replica (ssh)

```
# Write Ahead Log (WAL)
# https://www.postgresql.org/docs/10/static/runtime-config-wal.html
#------------------------------------------------------------------------------

# wal_level = replica
# fsync = on
# archive_mode = on
# archive_command = 'rsync -a %p postgres@REPLICA_IP_ADDRESS:~/master_wal/%f'
# full_page_writes = on
# max_wal_size = 1GB
# min_wal_size = 80MB


#------------------------------------------------------------------------------
# REPLICATION
#------------------------------------------------------------------------------

# max_wal_senders = 4
# wal_keep_segments = 0
                                                                  46,2          36%
```

3. PostgreSQL Replica (ssh)

```
                                # and 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                      # "off" disallows queries during recover
y
                                        # (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
                                                                  81,1          58%
```

Remove the **recovery.done** file located in `/var/lib/postgresql/11/main/` directory.

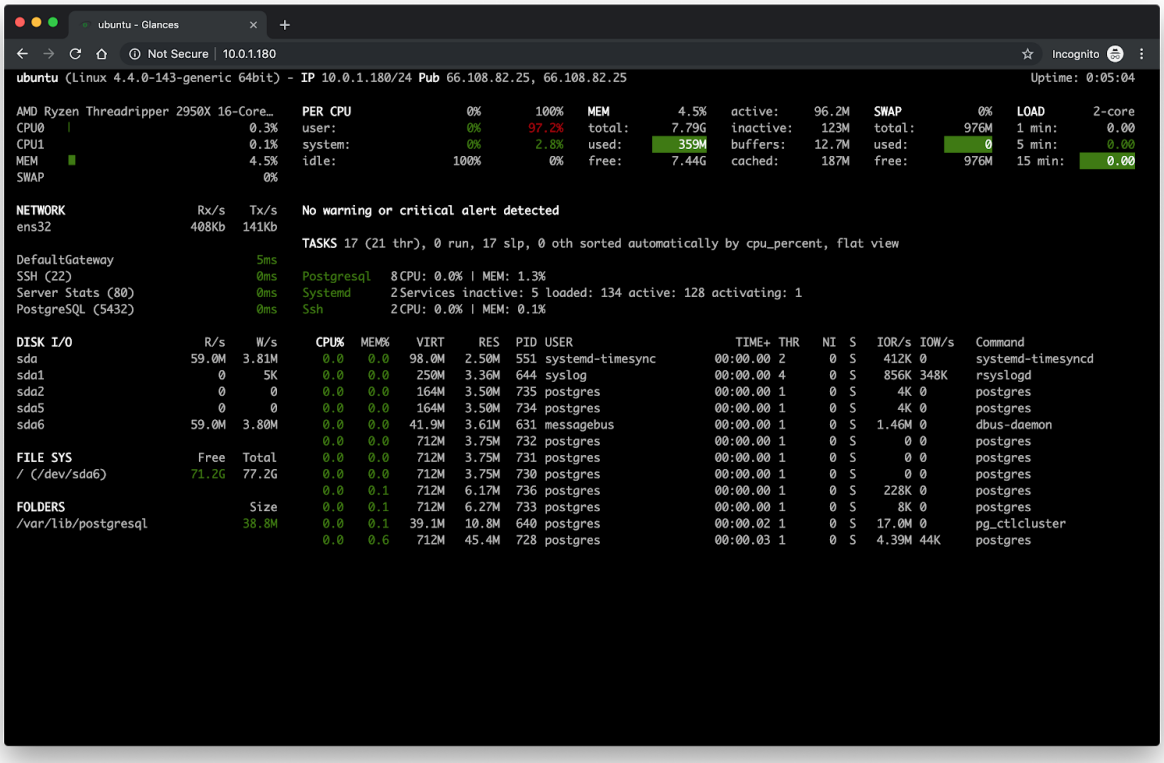
```
# Remove recovery.done
sudo -u postgres rm /var/lib/postgresql/11/main/recovery.done
```

Add Replica Server

Deploying

Deploy the PostgreSQL OVA on your platform as you would any other OVA. Refer to your platform's documentation for instructions on deploying OVA files.

Open the **new replica** server IP address in a web browser to view the server stats page,



Enable Replication Mode

Setup Primary Server

If **Replication** is disabled on the primary server, enable it to setup the new postgresql server as a replica.

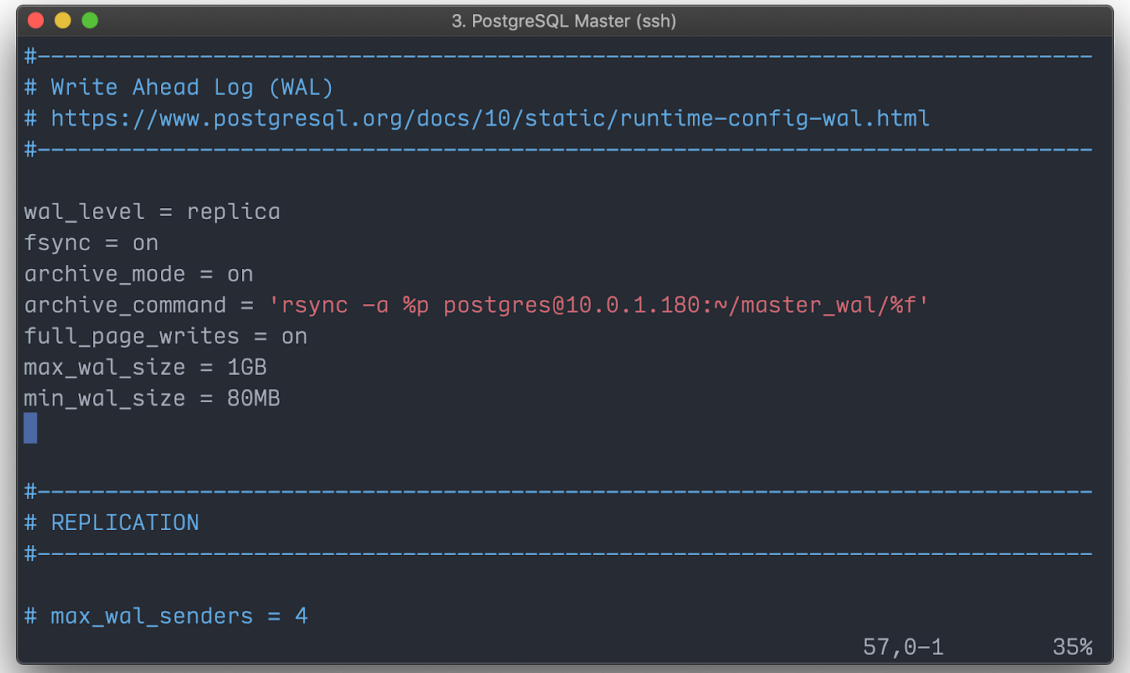
NOTE From this point the replica server that was **promoted to primary** is referred as **primary** and **newly created postgresql** server is referred as **replica**

```
# Stop Postgres service on the primary server
sudo systemctl stop pg


# Open the postgresql.conf
sudo vim /etc/postgresql/11/main/postgresql.conf
```

Enable the Replication and Write Ahead Log settings that are disabled in **Disable Replication Mode** step.

If the **replica** server IP address is **10.0.1.180** then the archive command,
archive_command = 'rsync -a %p postgres@**10.0.1.180**:~/master_wal/%f'

```
# Enable Write Ahead Log (WAL) settings from line 50

wal_level = replica
fsync = on
archive_mode = on
archive_command = 'rsync -a %p postgres@10.0.1.180:~/master_wal/%f'
full_page_writes = on
max_wal_size = 1GB
min_wal_size = 80MB
```



```
# Enable Replication settings from line 63

max_wal_senders = 4
wal_keep_segments = 0
```

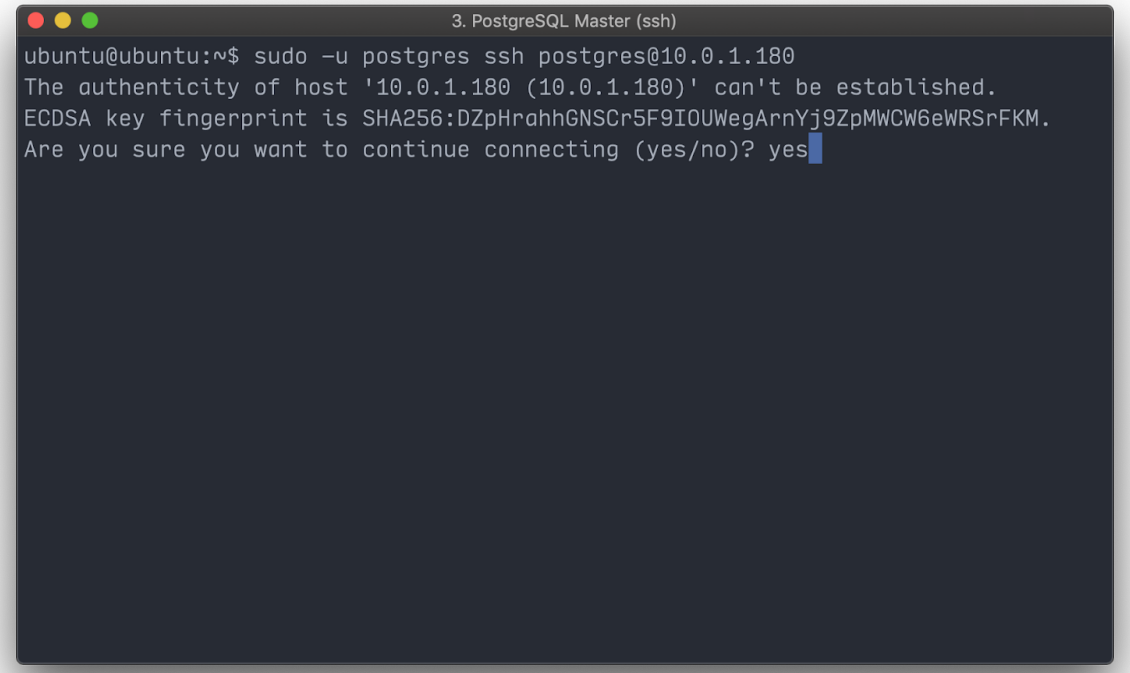
Update known_hosts

SSH into **replica** server from **primary** server using postgres user.

```
# SSH into replica server
sudo -u postgres ssh postgres@REPLICA_IP_ADDRESS
```

For example, if the replica server IP address is **`10.0.1.180`** then the command would look like

```
sudo -u postgres ssh postgres@10.0.1.180
```

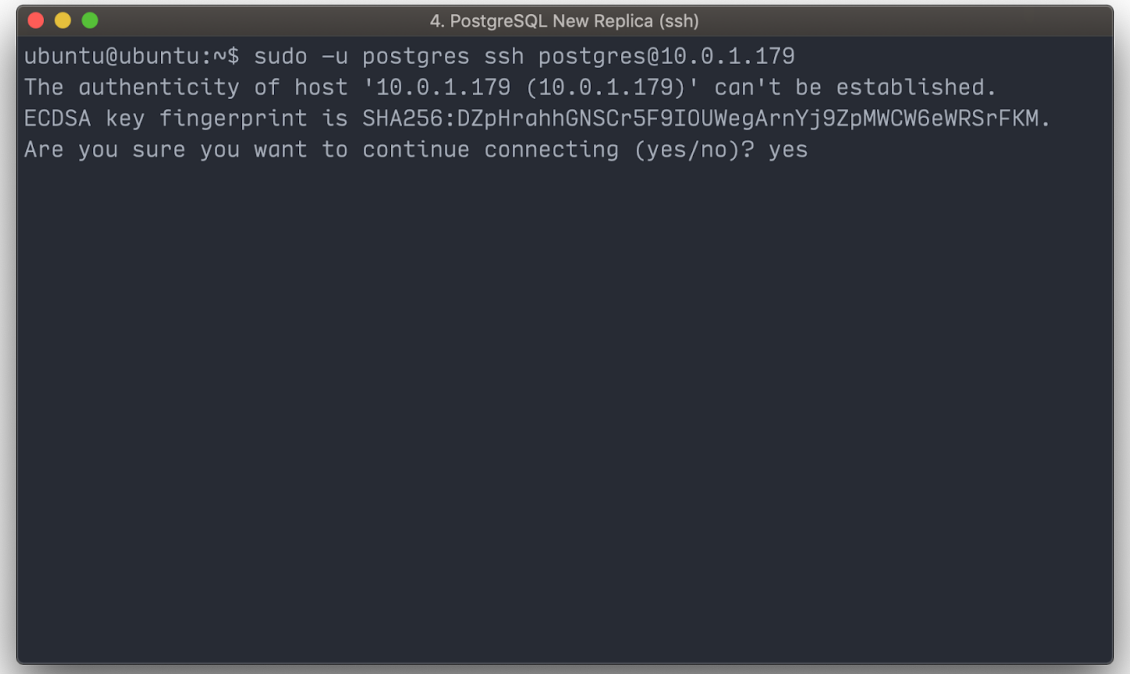


SSH into **primary** server from **replica** server using postgres user.

```
# SSH into primary server
sudo -u postgres ssh postgres@PRIMARY_IP_ADDRESS
```

For example, if the primary server IP address is **`10.0.1.179`** then the command would look like

```
sudo -u postgres ssh postgres@10.0.1.179
```



Reboot the primary server after enabling replication settings and known_hosts.

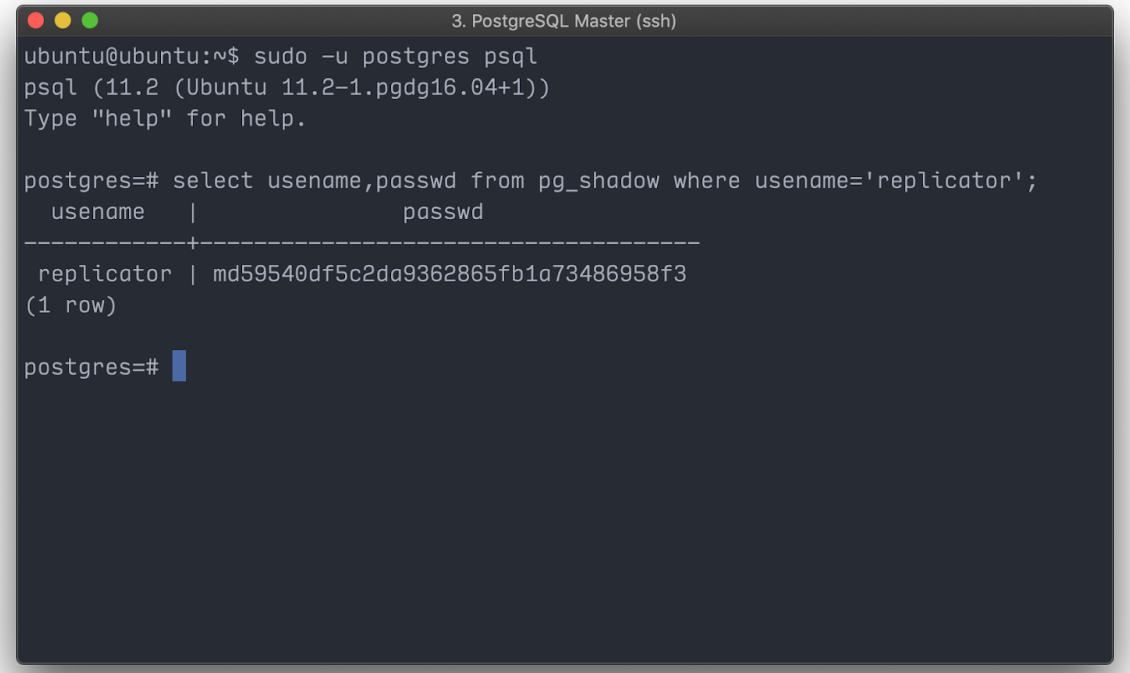
```
# Reboot
sudo reboot
```

Verify Replication User

Verify the replication user on the **primary** server exists with password configured during initial setup.

For example if replication user created during the initial setup is named **`replicator`** then

```
# Open postgres shell
sudo -u postgres psql

# Check if replicator user exists
select usename,passwd from pg_shadow where usename='replicator';

#   usename    |                 passwd
# ------------+-------------------------------------
#  replicator | md59540df5c2da9362865fb1a73486958f3
```



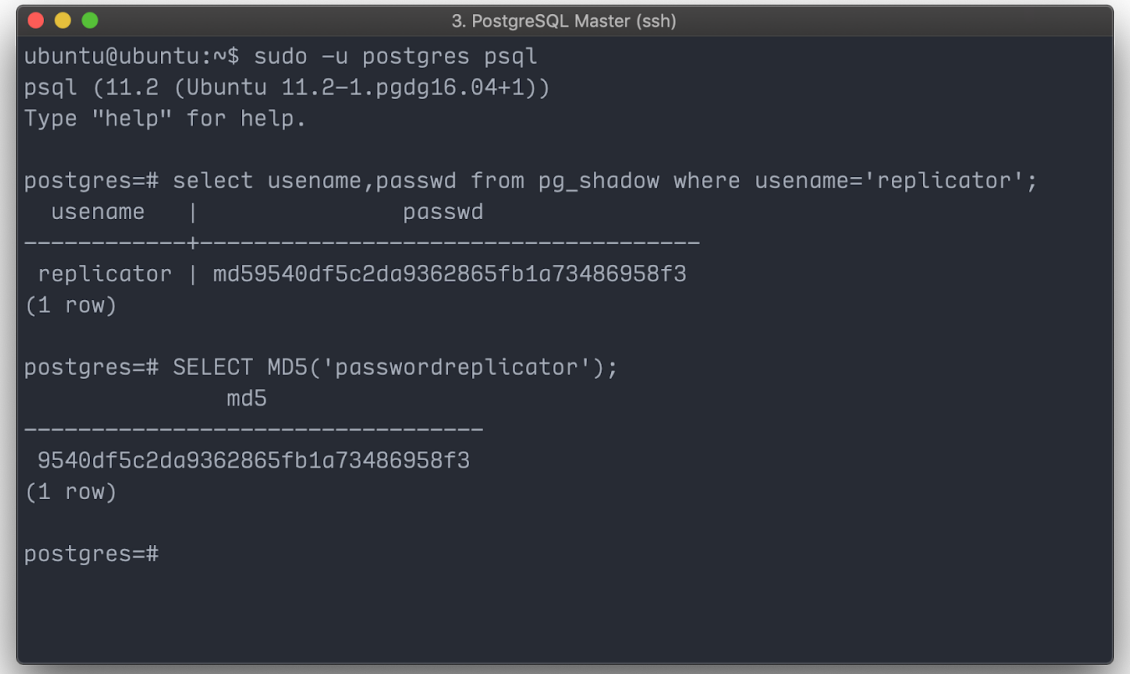
Confirm the replication user password by generating the encrypted password hash of the password, username pair and match it with **passwd** in the previous step.

For example if the replication user's username is **`replicator`** and password is **`password`**,

```
# Generate encrypted password hash by concatenating password, username
SELECT MD5('passwordreplicator');

#                md5
# ----------------------------------
#  9540df5c2da9362865fb1a73486958f3
```

This **md5** should match the **passwd** value from previous step after removing md5 prefix.

```
Encrypted hash : md59540df5c2da9362865fb1a73486958f3
MD5 hash       :    9540df5c2da9362865fb1a73486958f3
```



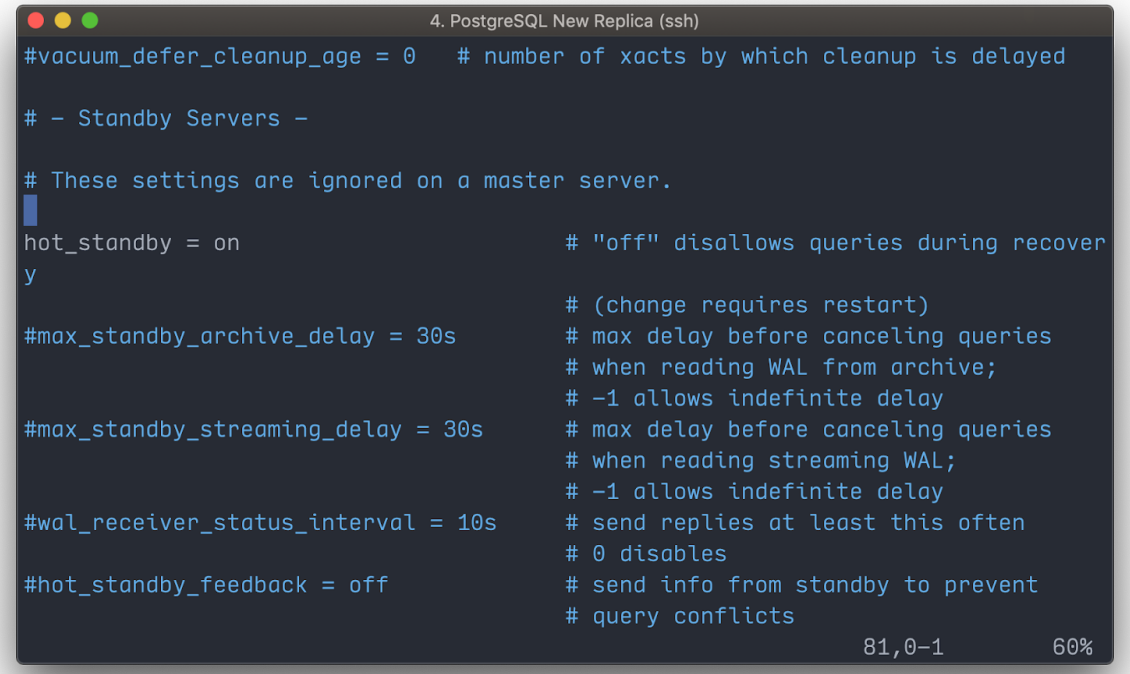
Setup Replica Server

Open the **postgresql.conf** on the replica server located in `/etc/postgresql/11/main/` directory.

```
# Open the postgresql.conf
sudo vim /etc/postgresql/11/main/postgresql.conf
```

Go to **`hot_standby`** on **line 82** and un-comment the line.

```
# Enable Replica as a hot standby
hot_standby  = on
```



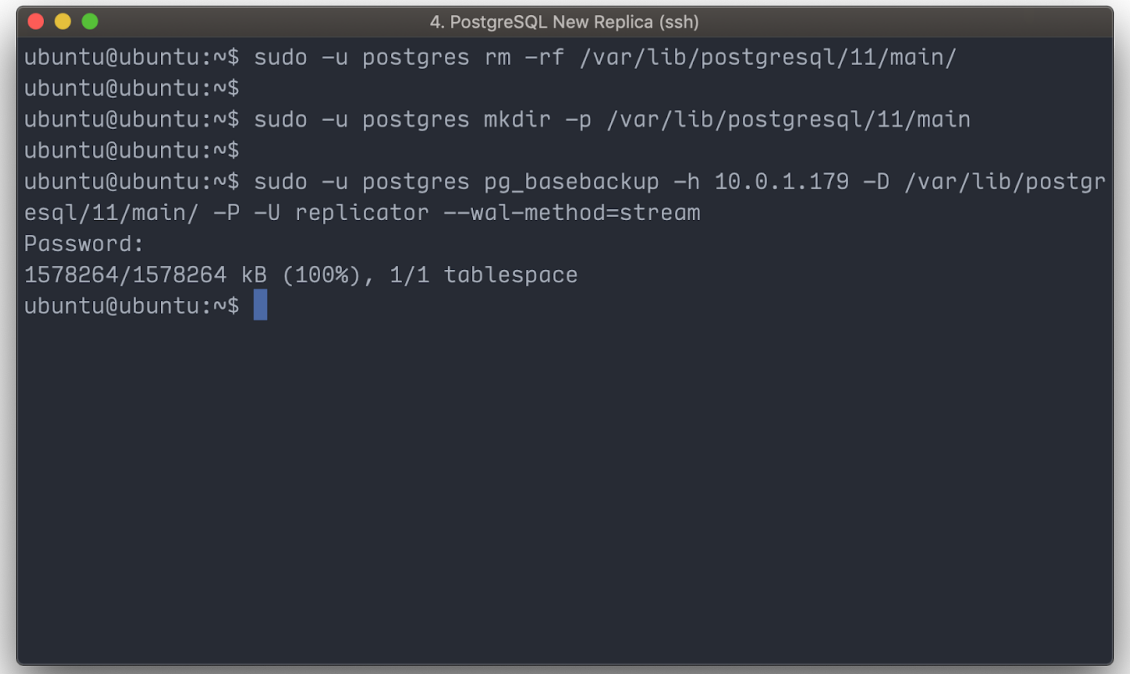
Sync the PG_DATA directory on replica server located at `/var/lib/postgresql/11/main` with the **primary** server.

```
# Clean data directory
sudo -u postgres rm -rf /var/lib/postgresql/11/main/

# Create main directory
sudo -u postgres mkdir -p /var/lib/postgresql/11/main

# Set 0700 permissions
sudo -u postgres chmod 0700 /var/lib/postgresql/11/main
```

Run a remote base backup with **pg_basebackup** on the replica server to sync the data directory from primary server to replica server,

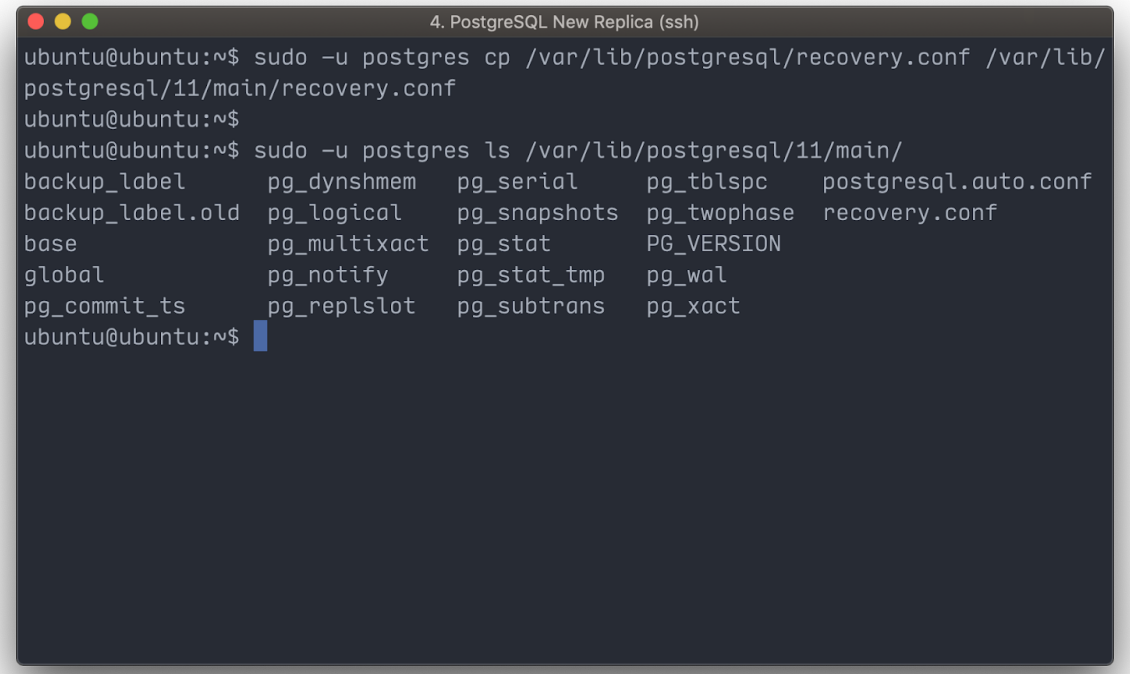
When asked for password, Enter the password of the replicator user verified in the previous steps. For example, if the IP address of the **primary** server is **`10.0.1.179`**, then the backup command would be

```
sudo -u postgres pg_basebackup -h 10.0.1.179 -D
/var/lib/postgresql/11/main/ -P -U replicator --wal-method=stream
```



Copy **recovery.conf** located at /var/lib/postgresql to /var/lib/postgresql/11/main/recovery.conf.

```
sudo -u postgres cp /var/lib/postgresql/recovery.conf
/var/lib/postgresql/11/main/recovery.conf
```



Generate Recovery.conf

Edit the **recovery.conf** located at **`/var/lib/postgresql/11/main/recovery.conf`**,

```
sudo -u postgres vim /var/lib/postgresql/11/main/recovery.conf
```

1. Set **standby_mode** to on line 6
   `standby_mode = 'on'`

2. Update the primary_conninfo on line 7 with the replication user's username, password and the primary server IP address. For Example, if the primary server IP address is **`10.0.1.179`** and the **`replicator`** password is **`password`**, the **primary_conninfo** would look like
   `primary_conninfo = 'host=`**`10.0.1.179`**` port=5432 user=replicator password=`**`password`**`'`

3. Update **trigger_file** on line 8 with a file path so when the file exists at the specified path. The specified file should not exist on replica server.
   `trigger_file = '/tmp/IAmTheMasterNow'`

4. Update **archive_cleanup_command** on line 9 with the following command
   `archive_cleanup_command = 'pg_archivecleanup /var/lib/postgresql/master_wal %r'`

4. PostgreSQL New Replica (ssh)

```
#------------------------------------------------------------------------------
# StandBy Replica
# Enable to following 3 lines to setup standby replica
#------------------------------------------------------------------------------

standby_mode          = 'on'
primary_conninfo      = 'host=10.0.1.179 port=5432 user=replicator password=pass
word'
trigger_file = '/tmp/IAmTheMasterNow'
archive_cleanup_command = 'pg_archivecleanup /var/lib/postgresql/master_wal %r'



#------------------------------------------------------------------------------
# Recovery
# Enable to following to recover frim WAL files
#------------------------------------------------------------------------------

# recovery_target_time = '2018-10-01 18:00:00 EDT'
                                                                  10,0-1        Top
```

Enable postgres service on the **replica** server and reboot.

```
# Enable postgres service
sudo systemctl enable pg

# Reboot
sudo reboot
```

After the reboot is successful run the commands on **page 5** to verify the replication status.

Point-In-Time Recovery (PITR)

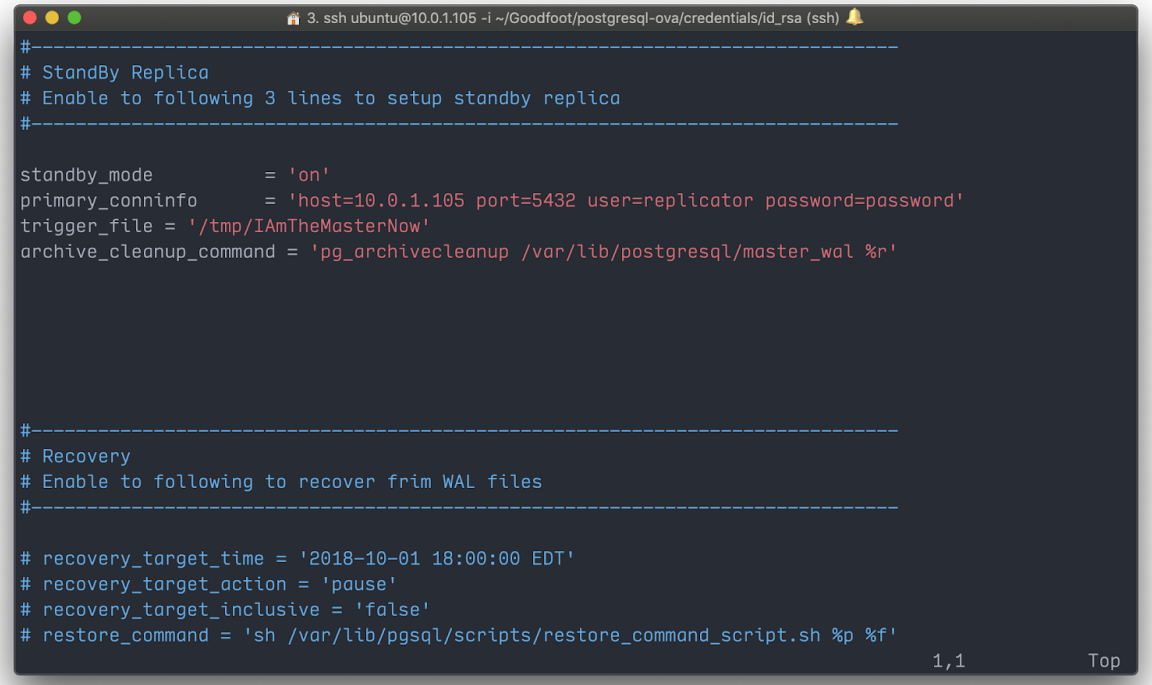
Point in time recovery allows the database to be recovered to a previous point in time.
In order to rollback the database, SSH into the primary server and shutdown the postgres service with the following command,

```
# Stop postgres service on Primary Server
sudo systemctl stop pg
```

SSH into the Replica server and follow the steps specified in **Promote Replica as Primary** section to promote Replica as Primary server.

SSH into the newly promoted primary server and open the **recovery.conf** located at **/var/lib/postgresql/recovery.conf.**

```
# Backup /var/lib/postgresql/recovery.conf
sudo -u postgres cp /var/lib/postgresql/recovery.conf
/var/lib/postgresql/recovery.conf.backup

# Open recovery.conf as postgres user
sudo -u postgres vim /var/lib/postgresql/recovery.conf
```



Disable the **Standby Replica** mode options by commenting the lines **6, 7, 8, 9**.

3. ssh ubuntu@10.0.1.105 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)

```
1 #------------------------------------------------------------------------------
2 # StandBy Replica
3 # Enable to following 3 lines to setup standby replica
4 #------------------------------------------------------------------------------
5
6 # standby_mode          = 'on'
7 # primary_conninfo      = 'host=10.0.1.105 port=5432 user=replicator password=password'
8 # trigger_file = '/tmp/IAmTheMasterNow'
9 # archive_cleanup_command = 'pg_archivecleanup /var/lib/postgresql/master_wal %r'
10
11
12
13
14
15
16 #------------------------------------------------------------------------------
17 # Recovery
18 # Enable to following to recover frim WAL files
19 #------------------------------------------------------------------------------
20
21 # recovery_target_time = '2018-10-01 18:00:00 EDT'
22 # recovery_target_action = 'pause'
23 # recovery_target_inclusive = 'false'
24 # restore_command = 'sh /var/lib/pgsql/scripts/restore_command_script.sh %p %f'
                                                                          9,2           Top
```

Enable the Recovery options by un-commenting and updating the following,

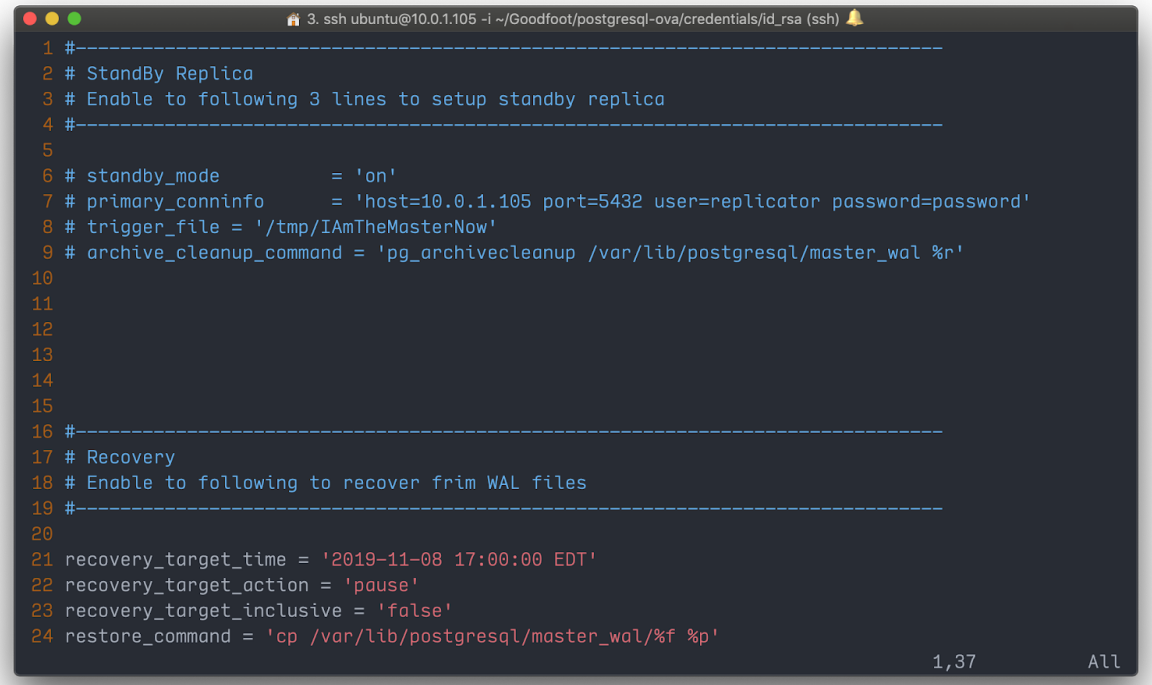
recovery_target_time
Specify the time upto which the database needs to be restored in the following format `YYYY-MM-DD HH:MM:SS TIMEZONE`.

For example if the database is to be restored to November 5th, 2019 5.00 PM Eastern then the timestamp would be **2019-11-08 17:00:00 EDT**

recovery_target_action
Set to **pause** the database after completing the recovery process.

recovery_target_inclusive
When set to **false**, the recovery process is stopped just before the specified recovery_target_time.

restore_command
Shell command to import the WAL files from the **master_wal** folder located at **/var/lib/postgresql/master_wal/.**

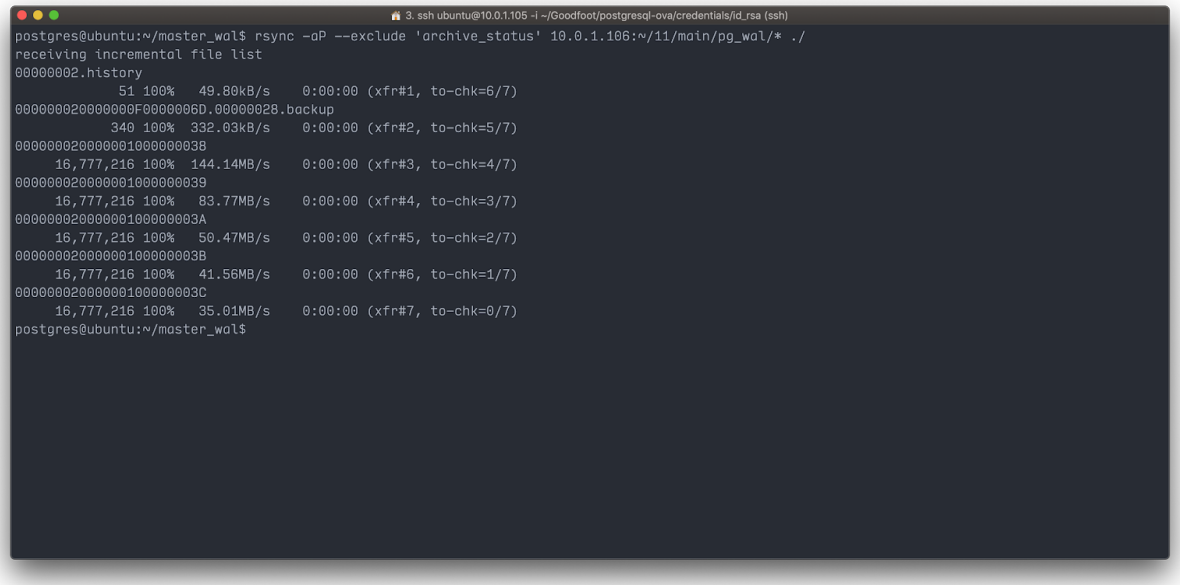


Copy the update recovery.conf to postgres database folder located at **/var/lib/postgresql/11/main/** on the Replica server.

```
sudo -u postgres cp /var/lib/postgresql/recovery.conf
/var/lib/postgresql/11/main/recovery.conf
```

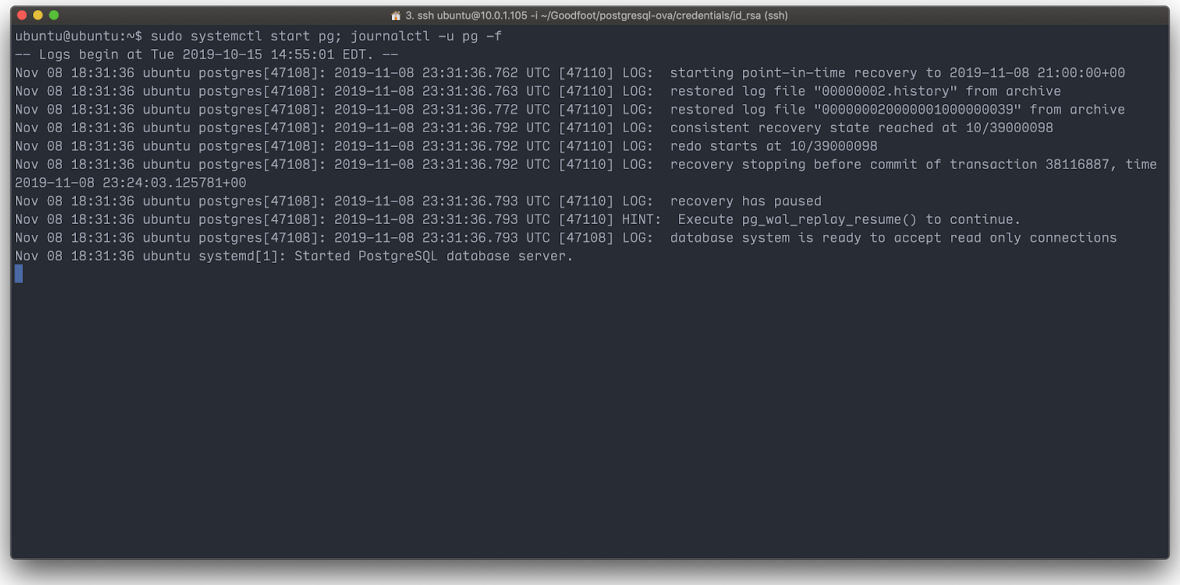
Copy the missing WAL files from the Primary server **`pg_wal`** folder into the **`master_wal`** folder on Replica server.

```
# Sync WAL files from Primary server to Replica server
rsync -aP --exclude 'archive_status' PRIMARY_IP:~/11/main/pg_wal/* ./
```

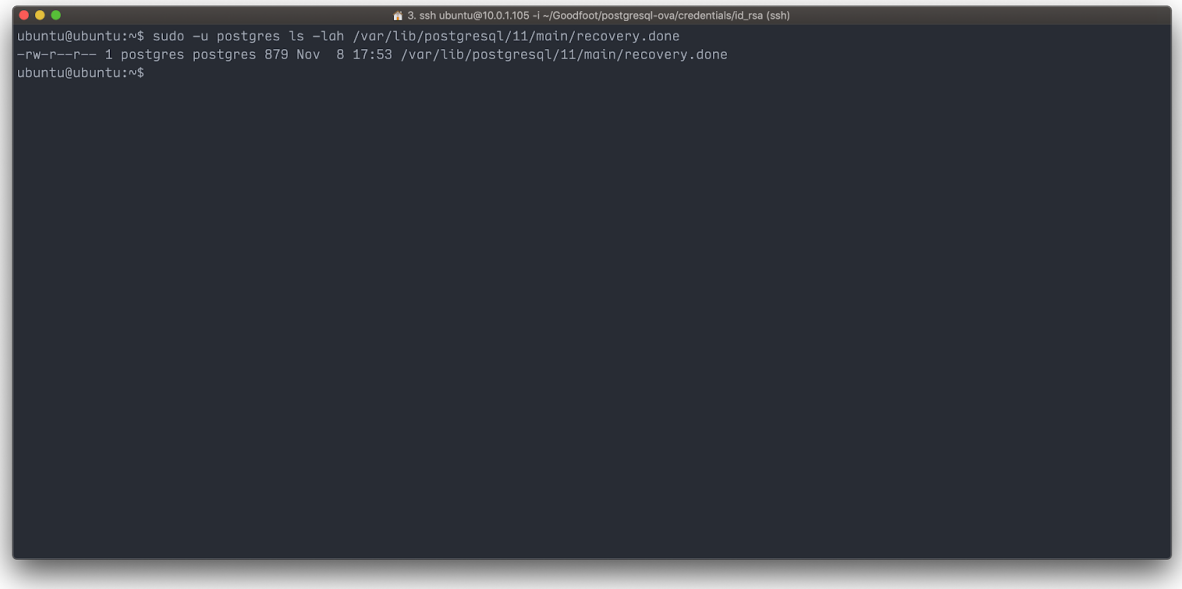


Start the postgres service on the Replica server to start the PITR recovery process.

```
# Start pg service
sudo systemctl start pg

# Check the logs to see if recovery is completed.
journalctl -u pg -f
```

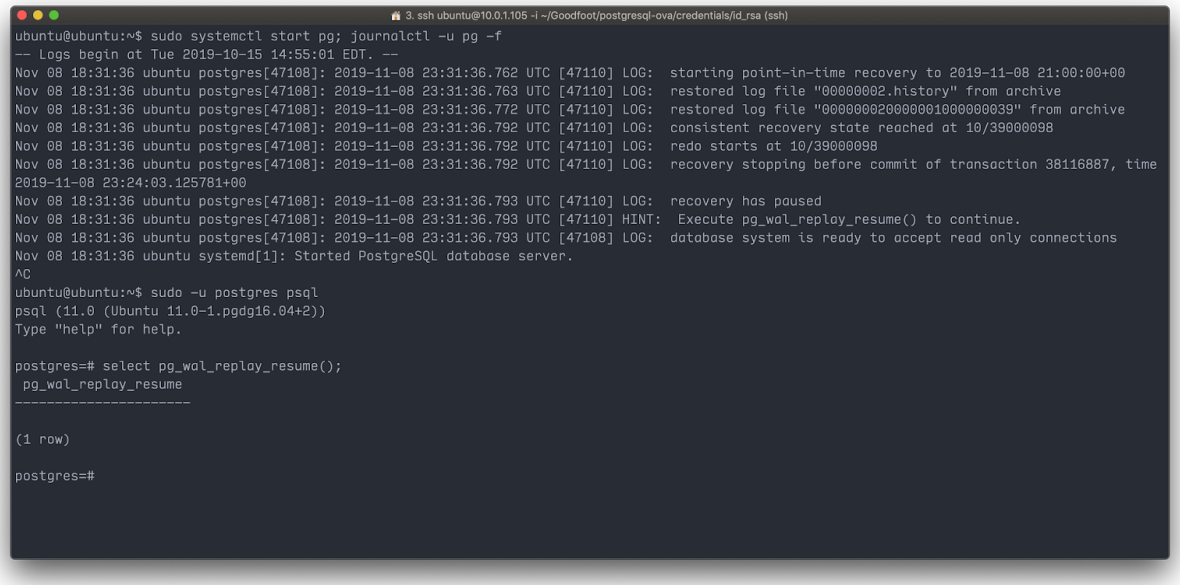


If the recovery is completed, **recovery.conf** located at /var/lib/postgresql/11/main/recovery.conf would be renamed to **recovery.done.**



After the postgres database is successfully restored, It would be paused until **`pg_wal_replay_resume`** command is executed. Open the postgresql shell as the postgres user and run the following command,

```
# Open psql shell
sudo -u postgres psql

# Resume regular operations
select pg_wal_replay_resume();
```



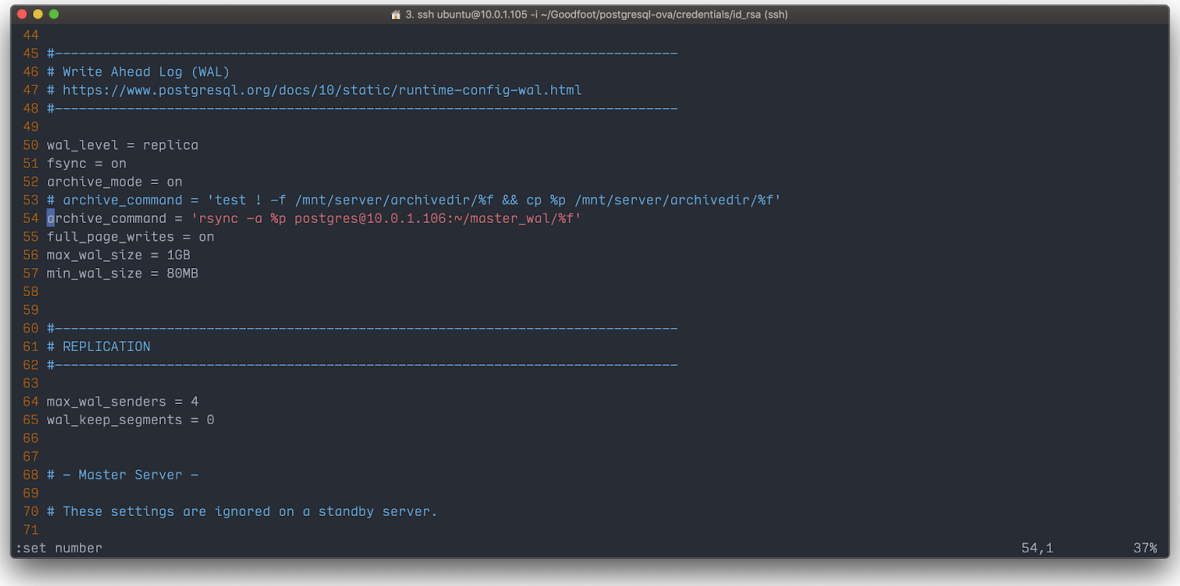
After recovery is completed follow the steps to convert Replica server to Primary server and Primary server to Replica server.

Convert Replica to Primary

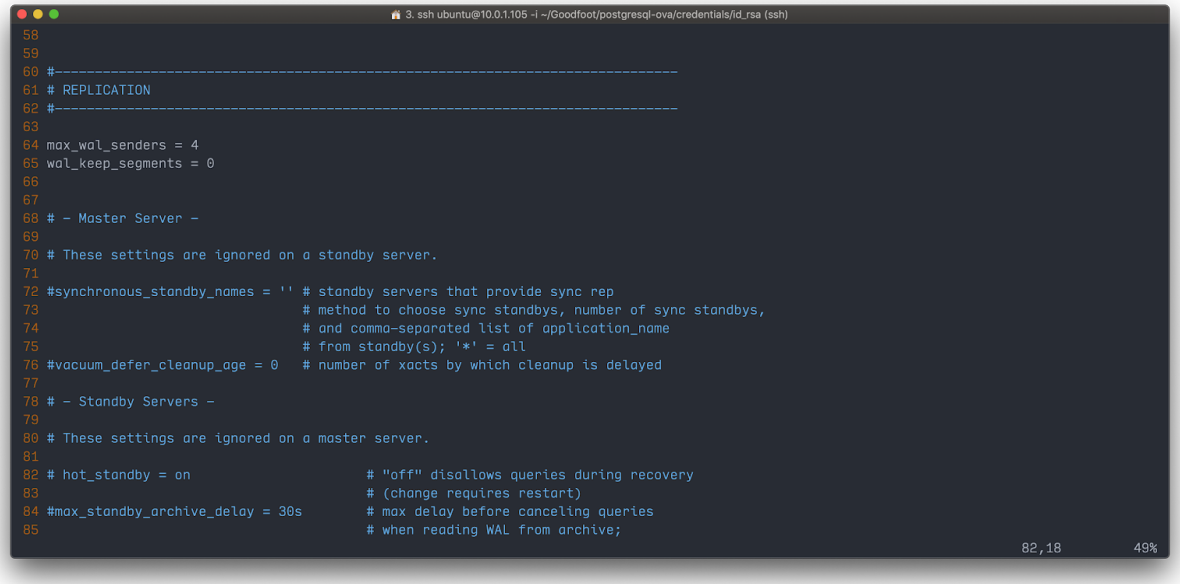
Open the postgresql.conf in Replica server located at **/etc/postgresql/11/main/postgresql.conf.**

For example if the **Primary** server IP is **`10.0.1.106`** and **Replica** server IP is **`10.0.1.105`**. Un-comment **archive_command** on line 54, to promote the Replica server as the new Primary server

```
archive_command = 'rsync -a %p postgres@PRIMARY_IP:~/master_wal/%f'
```



Disable **hot_standby** on line **82** by commenting it out.



Remove the **recovery.done** file located at /var/lib/postgresql/11/main.

```
# Remove recovery.done
sudo -u postgres rm /var/lib/postgresql/11/main/recovery.done
```

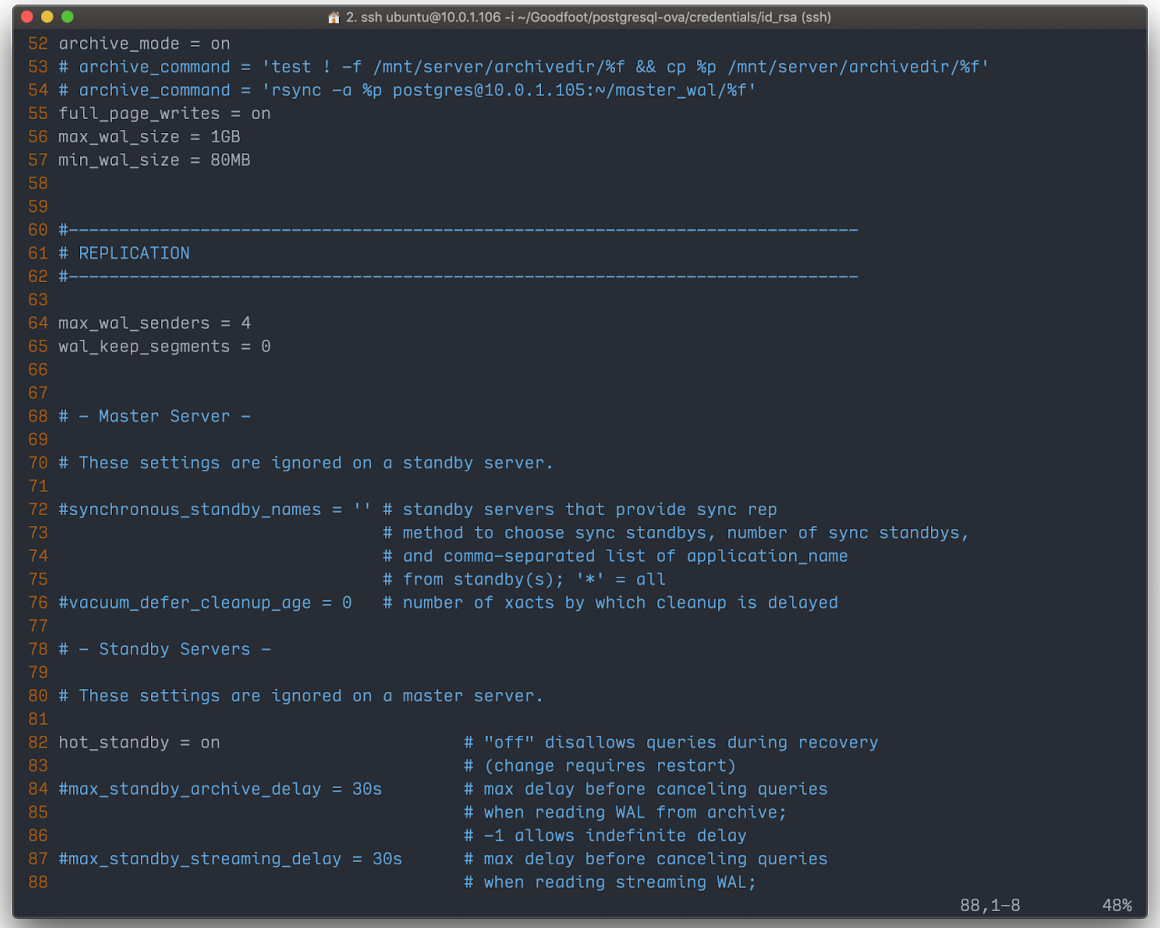
Convert Primary to Replica

SSH Into the Primary server and open the postgresql.conf located at **/etc/postgresql/11/main/postgresql.conf.**

Comment **archive_command** on line 54, to convert the Primary server as the new Replica server

```
# archive_command = 'rsync -a %p postgres@PRIMARY_IP:~/master_wal/%f'
```

Un-comment hot_standby on line 82



Generate the **recovery.conf** file at /var/lib/postgresql/11/main/recovery.conf based on the instruction in the **Generate Recovery.conf** section under Add Replica Server on **Page 25**.

```
2. ssh ubuntu@10.0.1.106 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)
#------------------------------------------------------------------------------
# StandBy Replica
# Enable to following 3 lines to setup standby replica
#------------------------------------------------------------------------------

standby_mode          = 'on'
primary_conninfo      = 'host=10.0.1.105 port=5432 user=replicator password=password'
trigger_file = '/tmp/IAmTheMasterNow'
archive_cleanup_command = 'pg_archivecleanup /var/lib/postgresql/master_wal %r'









#------------------------------------------------------------------------------
# Recovery
# Enable to following to recover frim WAL files
#------------------------------------------------------------------------------

# recovery_target_time = '2019-10-18 16:00:00 EDT'
# recovery_target_action = 'pause'
# recovery_target_inclusive = 'false'
# restore_command = 'cp /var/lib/postgresql/master_wal/%f %p'
                                                                      7,2           All
```

Restart the now Primary and Replica servers and following the instructions in **Replication Status** section to verify everything is working correctly.