

PostgreSQL Cluster

Recovery Guide

Version 1.0.0

Contents

Copyright Notice	3
Document Revision History	4
Replication status	5
Verify Primary	5
Verify Replica	6
Promote Replica as Primary	7
Disable Replication Mode	10
Add Replica Server	13
Deploying	13
Enable Replication Mode	14
Setup Primary Server	14
Update known_hosts	17
Verify Replication User	20
Setup Replica Server	22
Generate Recovery.conf	25
Point-In-Time Recovery (PITR)	28
Convert Replica to Primary	36
Convert Primary to Replica	38

Copyright Notice

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without express written permission. Under the law, reproducing includes translating into another language or format.

The software is protected by United States copyright laws and international treaty provision. Therefore, you must treat the software like any other copyrighted material (e.g. a book or sound recording).

Document Revision History

April 29, 2019

- Initial release of documentation

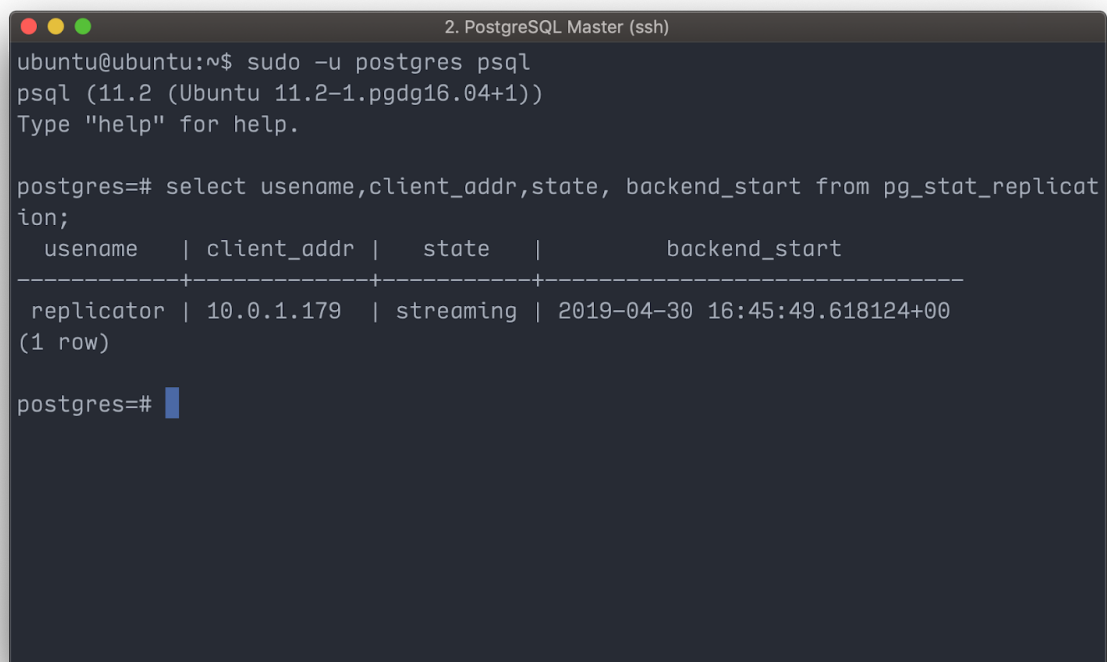
Replication status

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 username,client_addr,state,backend_start from
pg_stat_replication;
```



```
2. PostgreSQL Master (ssh)
ubuntu@ubuntu:~$ sudo -u postgres psql
psql (11.2 (Ubuntu 11.2-1.pgdg16.04+1))
Type "help" for help.

postgres=# select username,client_addr,state, backend_start from pg_stat_replicat
ion;
 username | client_addr | state   | backend_start
-----+-----+-----+-----
 replicator | 10.0.1.179  | streaming | 2019-04-30 16:45:49.618124+00
(1 row)

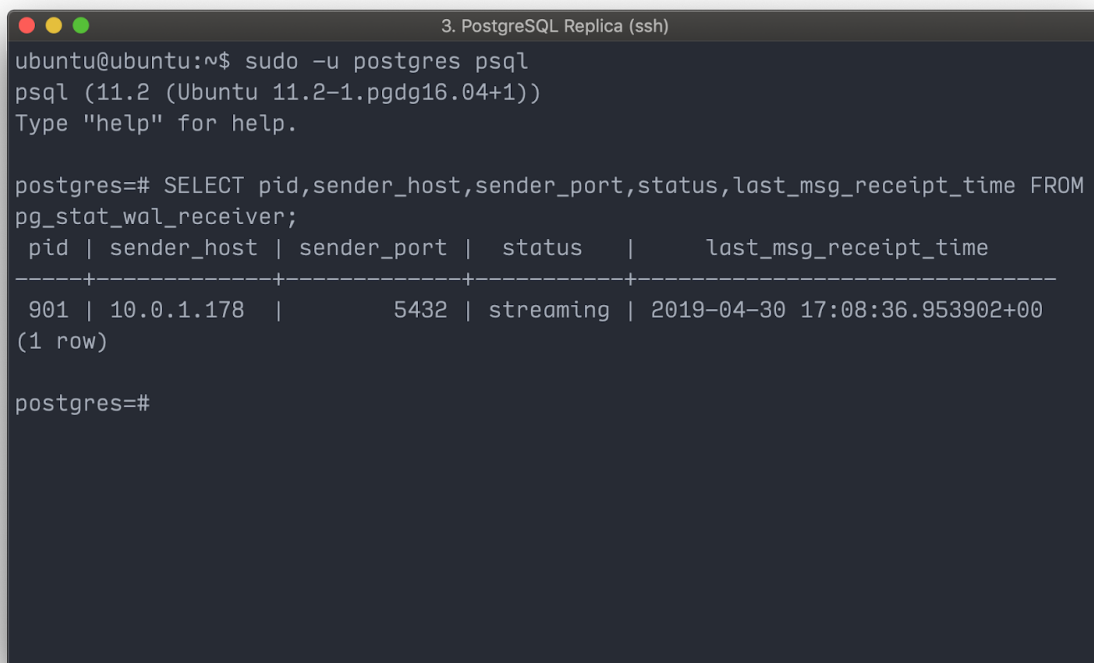
postgres=#
```

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;
```

A terminal window titled "3. PostgreSQL Replica (ssh)" showing the execution of a SQL query to check replication status. The user runs 'sudo -u postgres psql' and then 'SELECT pid,sender_host,sender_port,status,last_msg_receipt_time FROM pg_stat_wal_receiver;'. The output shows a single row with pid 901, sender_host 10.0.1.178, sender_port 5432, status streaming, and last_msg_receipt_time 2019-04-30 17:08:36.953902+00.

```
ubuntu@ubuntu:~$ sudo -u postgres psql
psql (11.2 (Ubuntu 11.2-1.pgdg16.04+1))
Type "help" for help.

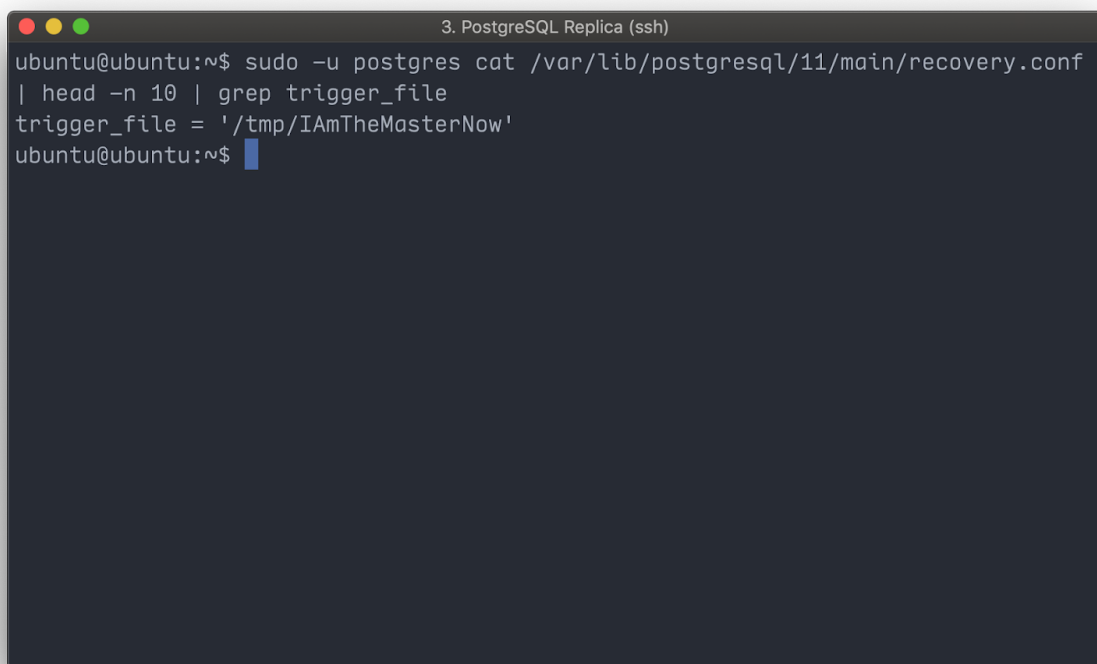
postgres=# SELECT pid,sender_host,sender_port,status,last_msg_receipt_time FROM
pg_stat_wal_receiver;
 pid | sender_host | sender_port | status  | last_msg_receipt_time
-----+-----+-----+-----+-----
  901 | 10.0.1.178  |         5432 | streaming | 2019-04-30 17:08:36.953902+00
(1 row)

postgres=#
```

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
```



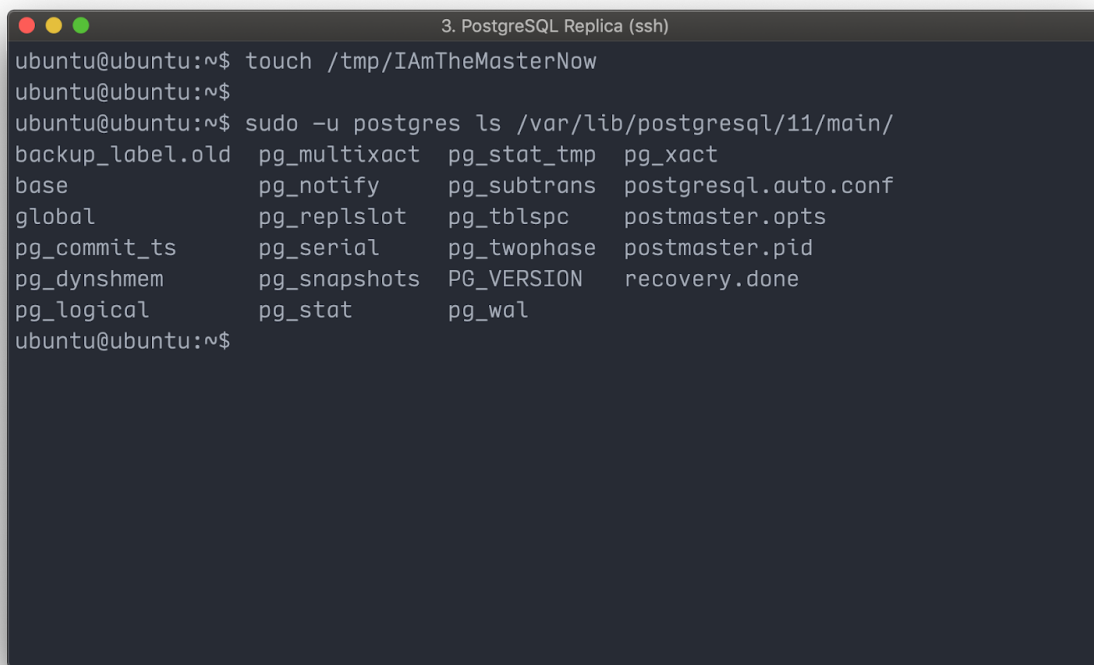
```
3. PostgreSQL Replica (ssh)
ubuntu@ubuntu:~$ sudo -u postgres cat /var/lib/postgresql/11/main/recovery.conf
| head -n 10 | grep trigger_file
trigger_file = '/tmp/IAmTheMasterNow'
ubuntu@ubuntu:~$
```

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/
```

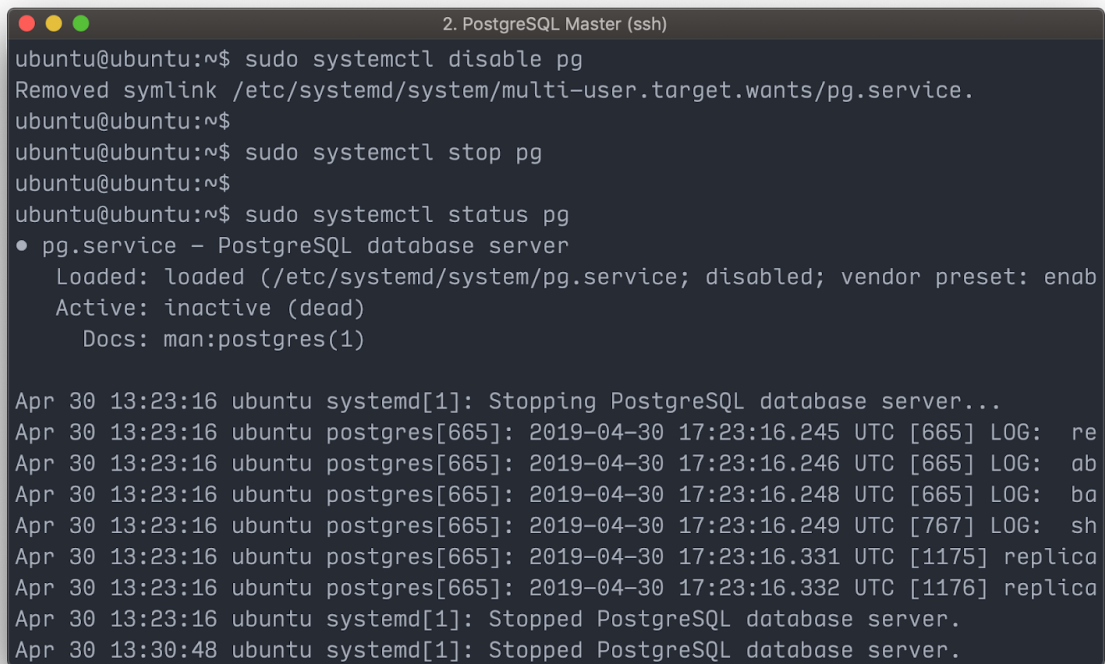
A terminal window titled "3. PostgreSQL Replica (ssh)" showing the execution of two commands. The first command, "touch /tmp/IAmTheMasterNow", is executed successfully. The second command, "sudo -u postgres ls /var/lib/postgresql/11/main/", lists the contents of the PostgreSQL data directory. The output shows a list of files and directories including backup_label.old, base, global, pg_commit_ts, pg_dynshmem, pg_logical, pg_multixact, pg_notify, pg_replslot, pg_serial, pg_stat, pg_stat_tmp, pg_subtrans, pg_tblspc, pg_twophase, pg_wal, postgresql.auto.conf, postmaster.opts, postmaster.pid, PG_VERSION, and recovery.done.

```
ubuntu@ubuntu:~$ touch /tmp/IAmTheMasterNow
ubuntu@ubuntu:~$
ubuntu@ubuntu:~$ sudo -u postgres ls /var/lib/postgresql/11/main/
backup_label.old  pg_multixact  pg_stat_tmp  pg_xact
base              pg_notify    pg_subtrans  postgresql.auto.conf
global           pg_replslot  pg_tblspc    postmaster.opts
pg_commit_ts     pg_serial    pg_twophase  postmaster.pid
pg_dynshmem      pg_snapshots PG_VERSION   recovery.done
pg_logical       pg_stat      pg_wal
ubuntu@ubuntu:~$
```

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 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

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
```



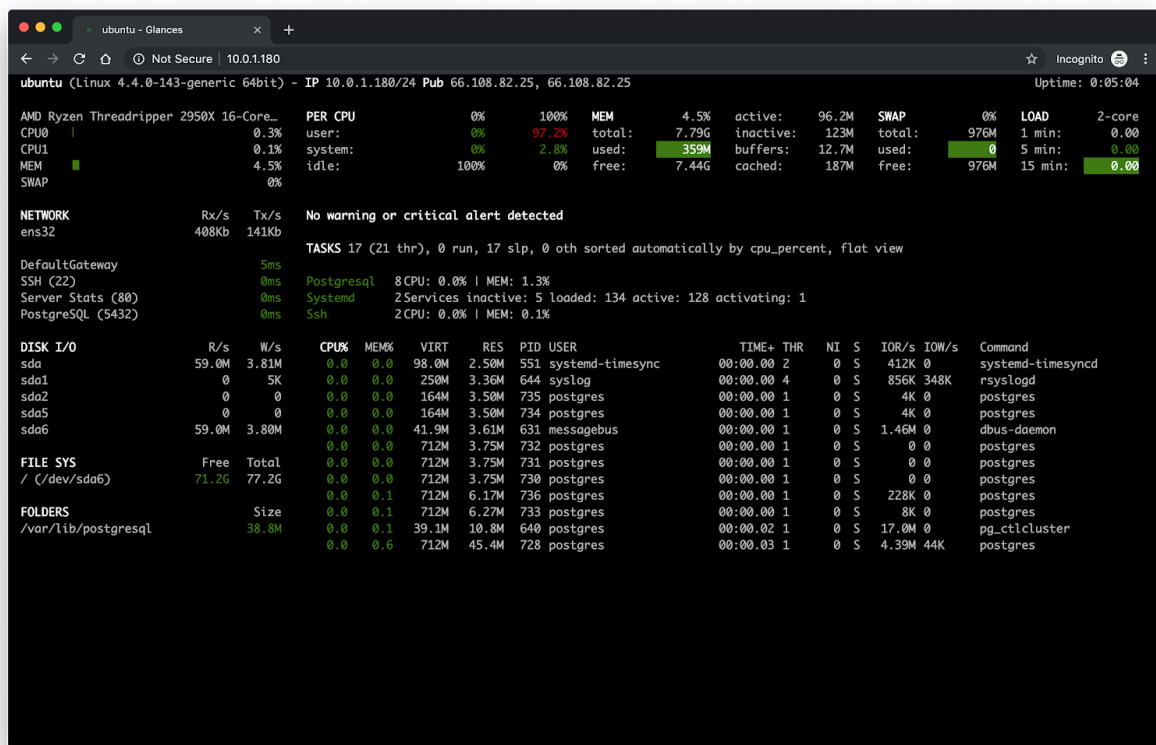
```
3. PostgreSQL Replica (ssh)
ubuntu@ubuntu:~$ sudo -u postgres rm /var/lib/postgresql/11/main/recovery.done
ubuntu@ubuntu:~$
ubuntu@ubuntu:~$ sudo -u postgres ls /var/lib/postgresql/11/main/
backup_label.old  pg_multixact  pg_stat_tmp  pg_xact
base              pg_notify    pg_subtrans  postgresql.auto.conf
global           pg_replslot  pg_tblspc    postmaster.opts
pg_commit_ts     pg_serial    pg_twophase  postmaster.pid
pg_dynshmem      pg_snapshots PG_VERSION
pg_logical       pg_stat      pg_wal
ubuntu@ubuntu:~$
```


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
```



```
3. PostgreSQL Master (ssh)
ubuntu@ubuntu:~$ sudo systemctl stop pg
ubuntu@ubuntu:~$
ubuntu@ubuntu:~$ sudo vim /etc/postgresql/11/main/postgresql.conf
```

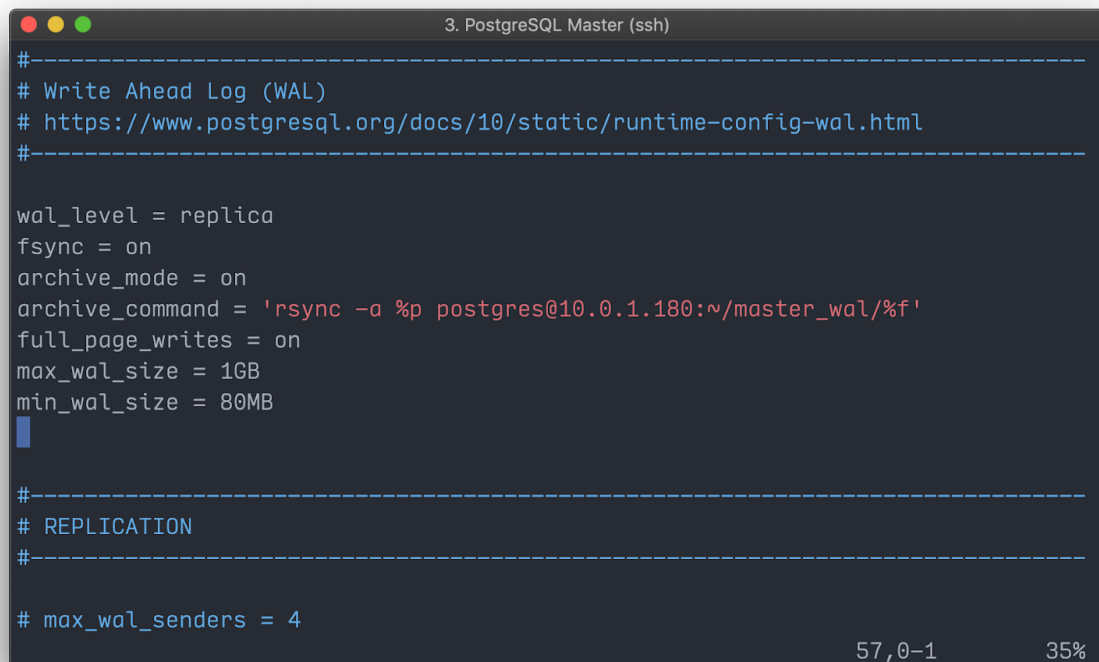
Enable the Replication and Write Ahead Log settings that are disabled in **Disable Replication Mode** step.

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
```



```
3. PostgreSQL Master (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@10.0.1.180:~/master_wal/%f'
full_page_writes = on
max_wal_size = 1GB
min_wal_size = 80MB
█

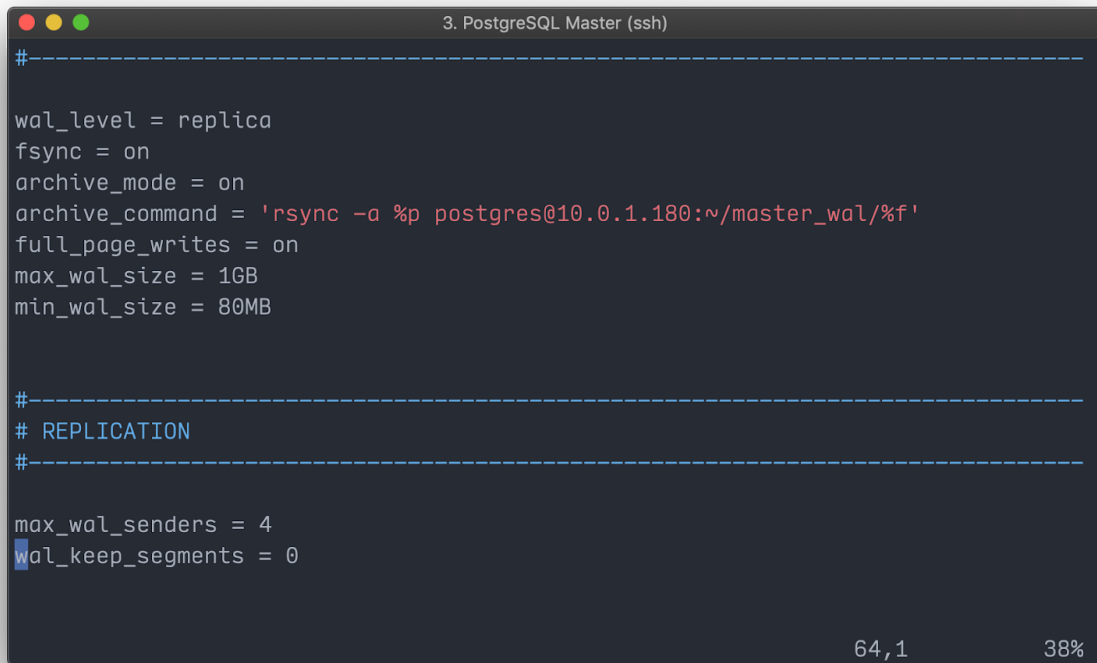
#-----
# REPLICATION
#-----

# max_wal_senders = 4

57,0-1 35%
```

Enable Replication settings from **line 63**

```
max_wal_senders = 4  
wal_keep_segments = 0
```



A terminal window titled "3. PostgreSQL Master (ssh)" with a dark background and light blue text. The window displays PostgreSQL configuration settings. It starts with a comment line "# Enable Replication settings from line 63" followed by a dashed line. Then, it lists several settings: wal_level = replica, fsync = on, archive_mode = on, archive_command = 'rsync -a %p postgres@10.0.1.180:~/master_wal/%f' (with 'rsync' in red), full_page_writes = on, max_wal_size = 1GB, and min_wal_size = 80MB. This is followed by another dashed line, a comment line "# REPLICATION", and a third dashed line. Finally, it shows max_wal_senders = 4 and wal_keep_segments = 0, with the latter being highlighted by a blue cursor. In the bottom right corner, the text "64,1" and "38%" are visible.

```
#-----  
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  
  
#-----  
# REPLICATION  
#-----  
  
max_wal_senders = 4  
wal_keep_segments = 0  
  
64,1 38%
```

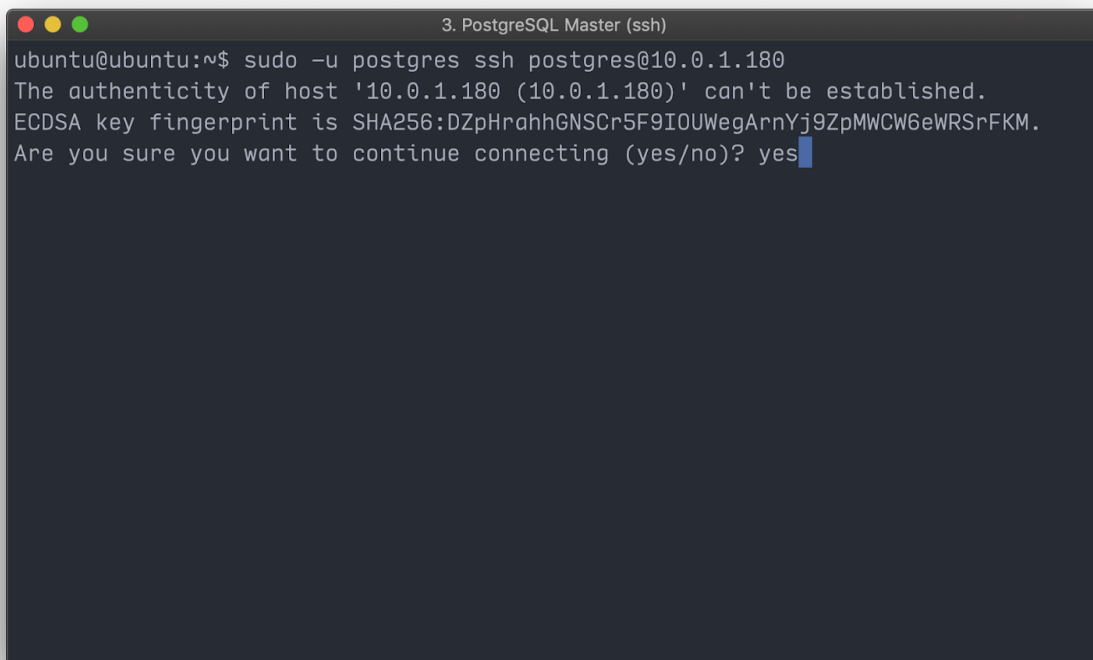
Update known_hosts

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
```

A terminal window titled "3. PostgreSQL Master (ssh)" showing the execution of the command "sudo -u postgres ssh postgres@10.0.1.180". The terminal output displays a warning about the authenticity of the host, showing the ECDSA key fingerprint "SHA256:DZpHrahhGNSCr5F9IOUWegArnYj9ZpMWCW6eWRSrFKM". It then prompts the user "Are you sure you want to continue connecting (yes/no)?", with "yes" entered and a cursor at the end.

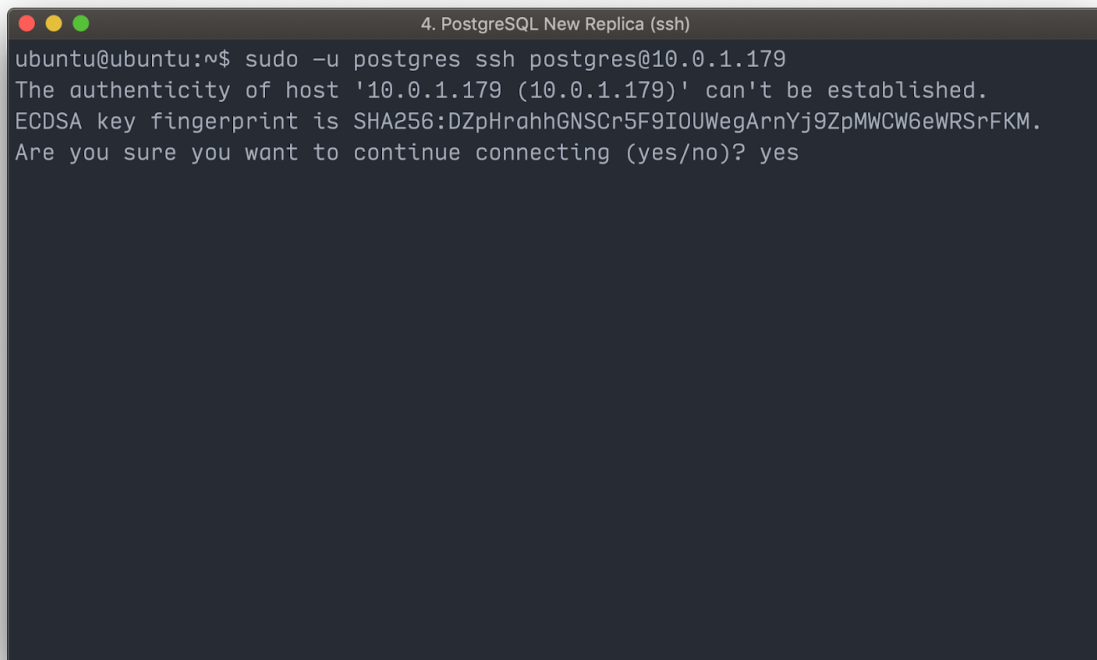
```
3. PostgreSQL Master (ssh)  
ubuntu@ubuntu:~$ sudo -u postgres ssh postgres@10.0.1.180  
The authenticity of host '10.0.1.180 (10.0.1.180)' can't be established.  
ECDSA key fingerprint is SHA256:DZpHrahhGNSCr5F9IOUWegArnYj9ZpMWCW6eWRSrFKM.  
Are you sure you want to continue connecting (yes/no)? yes
```

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
```

A terminal window titled "4. PostgreSQL New Replica (ssh)" is shown. The prompt is "ubuntu@ubuntu:~\$". The command entered is "sudo -u postgres ssh postgres@10.0.1.179". The output shows a warning about the authenticity of the host '10.0.1.179 (10.0.1.179)' and an ECDSA key fingerprint. The user responds with "yes".

```
4. PostgreSQL New Replica (ssh)  
ubuntu@ubuntu:~$ sudo -u postgres ssh postgres@10.0.1.179  
The authenticity of host '10.0.1.179 (10.0.1.179)' can't be established.  
ECDSA key fingerprint is SHA256:DZpHrahhGNsCr5F9IOUWegArnYj9ZpMWCW6eWRSrFKM.  
Are you sure you want to continue connecting (yes/no)? yes
```

Reboot the primary server after enabling replication settings and known_hosts.

```
# Reboot  
sudo reboot
```

Verify Replication User

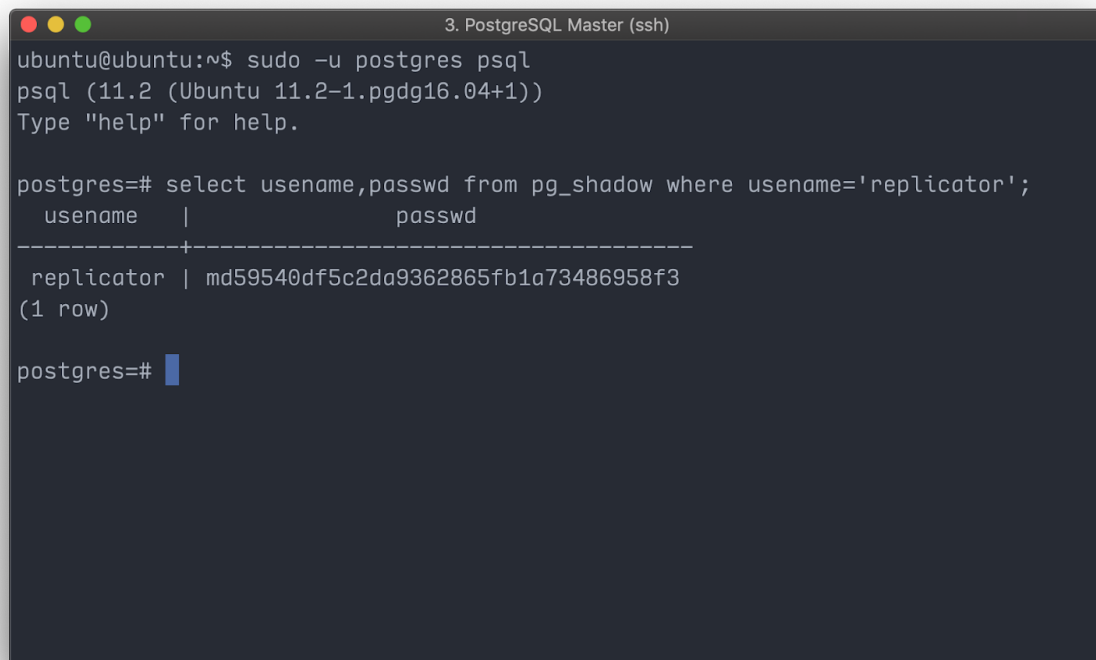
Verify the replication user on the **primary** server exists with password configured during initial setup.

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 username,passwd from pg_shadow where username='replicator';

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



The screenshot shows a terminal window titled "3. PostgreSQL Master (ssh)". The user runs the command `sudo -u postgres psql` to open the PostgreSQL shell. The prompt changes to `postgres=#`. The user then runs the SQL query `select username,passwd from pg_shadow where username='replicator';`. The output shows a single row with the username `replicator` and the password `md59540df5c2da9362865fb1a73486958f3`. The prompt returns to `postgres=#`.

```
ubuntu@ubuntu:~$ sudo -u postgres psql
psql (11.2 (Ubuntu 11.2-1.pgdg16.04+1))
Type "help" for help.

postgres=# select username,passwd from pg_shadow where username='replicator';
 username | passwd
-----+-----
 replicator | md59540df5c2da9362865fb1a73486958f3
(1 row)

postgres=#
```


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.

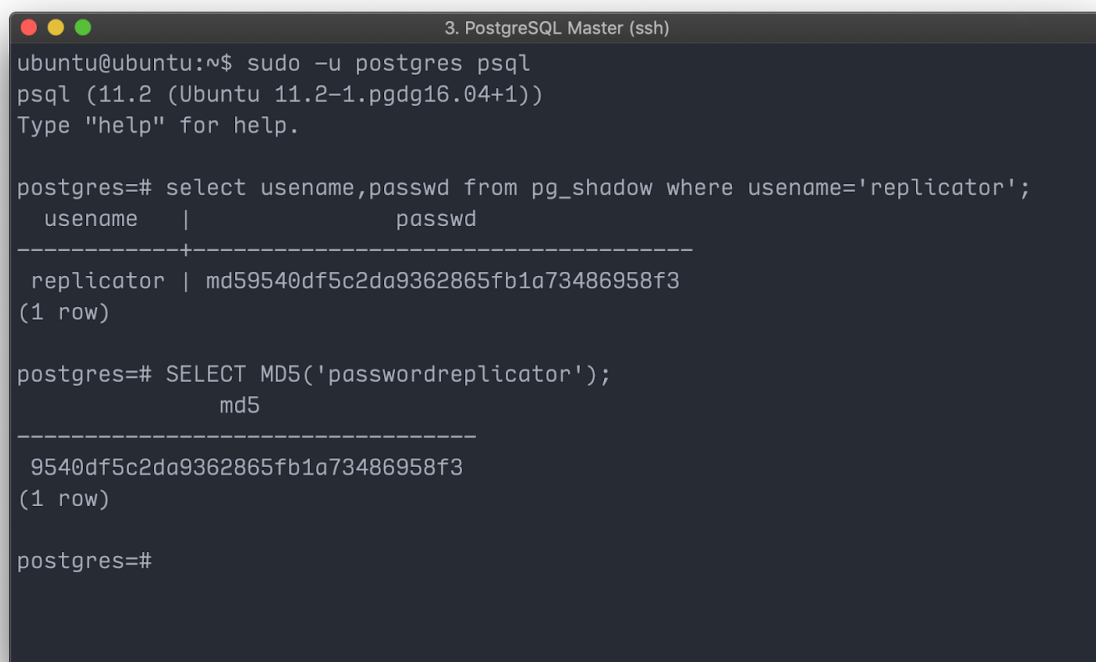
For example if the replication user's username is **replicator** and password is **password**,

```
# 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
```



```
3. PostgreSQL Master (ssh)
ubuntu@ubuntu:~$ sudo -u postgres psql
psql (11.2 (Ubuntu 11.2-1.pgdg16.04+1))
Type "help" for help.

postgres=# select username,passwd from pg_shadow where username='replicator';
 username |          passwd
-----+-----
 replicator | md59540df5c2da9362865fb1a73486958f3
(1 row)

postgres=# SELECT MD5('passwordreplicator');
          md5
-----
 9540df5c2da9362865fb1a73486958f3
(1 row)

postgres=#
```

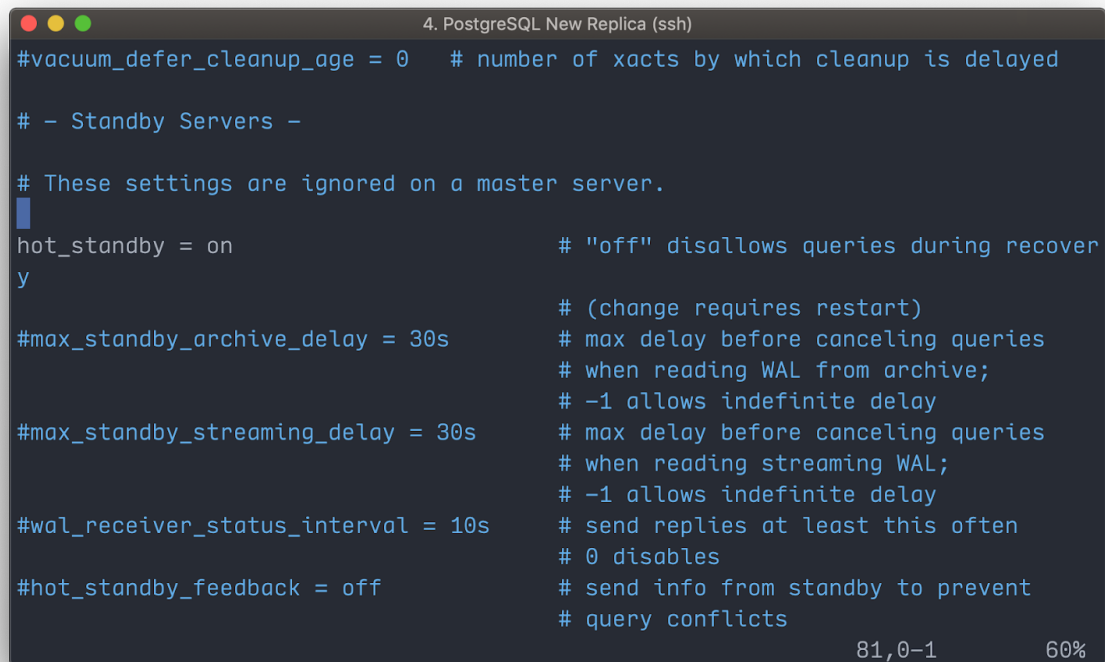
Setup Replica Server

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
```



A screenshot of a terminal window titled "4. PostgreSQL New Replica (ssh)". The terminal displays the contents of the `postgresql.conf` file, specifically the section for standby servers. The configuration includes settings for `vacuum_defer_cleanup_age`, `hot_standby` (set to `on`), `max_standby_archive_delay` (set to `30s`), `max_standby_streaming_delay` (set to `30s`), `wal_receiver_status_interval` (set to `10s`), and `hot_standby_feedback` (set to `off`). Each setting is followed by a comment explaining its purpose. The terminal also shows a progress bar at the bottom right indicating 81,0-1 and 60% completion.

```
4. PostgreSQL New Replica (ssh)
#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 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

81,0-1 60%
```

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/

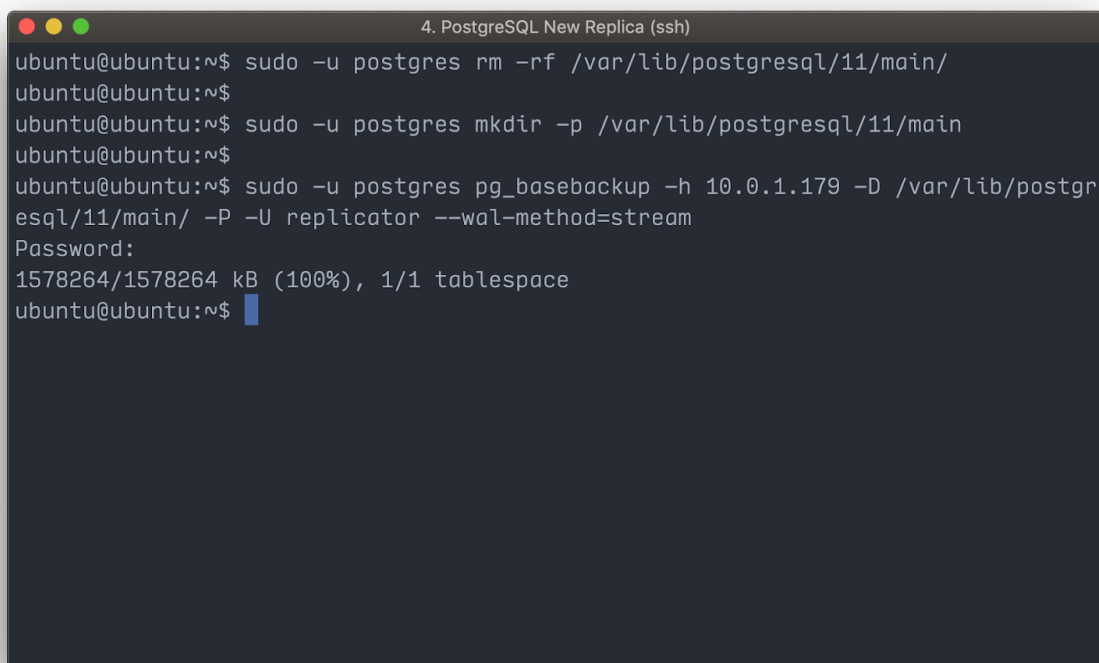
# Create main directory
sudo -u postgres mkdir -p /var/lib/postgresql/11/main

# 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

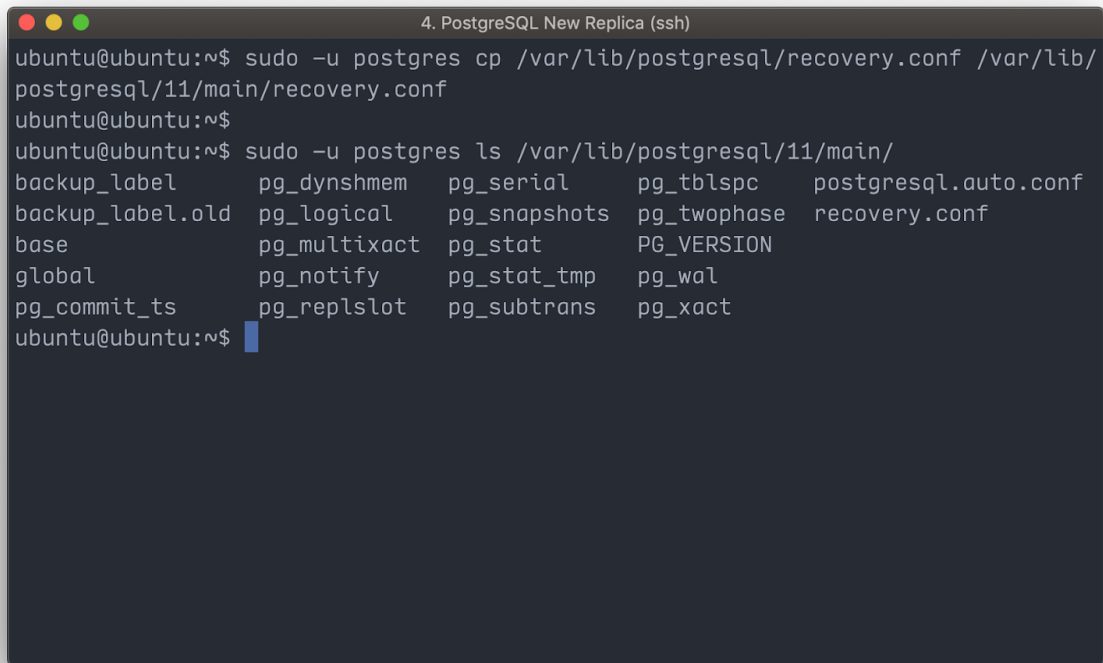
```
sudo -u postgres pg_basebackup -h 10.0.1.179 -D
/var/lib/postgresql/11/main/ -P -U replicator --wal-method=stream
```

A terminal window titled "4. PostgreSQL New Replica (ssh)" showing the execution of the pg_basebackup command. The user runs the command with the host 10.0.1.179 and the destination directory /var/lib/postgresql/11/main/. The command prompts for a password, which is entered. The output shows the backup progress: 1578264/1578264 kB (100%), 1/1 tablespace. The prompt returns to the user.

```
ubuntu@ubuntu:~$ sudo -u postgres rm -rf /var/lib/postgresql/11/main/
ubuntu@ubuntu:~$ sudo -u postgres mkdir -p /var/lib/postgresql/11/main
ubuntu@ubuntu:~$ sudo -u postgres pg_basebackup -h 10.0.1.179 -D /var/lib/postgr
esql/11/main/ -P -U replicator --wal-method=stream
Password:
1578264/1578264 kB (100%), 1/1 tablespace
ubuntu@ubuntu:~$
```

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
```



A terminal window titled "4. PostgreSQL New Replica (ssh)" showing the execution of two commands. The first command copies the recovery.conf file from /var/lib/postgresql to /var/lib/postgresql/11/main/. The second command lists the contents of the /var/lib/postgresql/11/main/ directory, displaying a list of files and subdirectories including backup_label, pg_dynshmem, pg_serial, pg_tblspc, postgresql.auto.conf, backup_label.old, pg_logical, pg_snapshots, pg_twophase, recovery.conf, base, pg_multixact, pg_stat, PG_VERSION, global, pg_notify, pg_stat_tmp, pg_wal, pg_commit_ts, pg_replslot, pg_subtrans, and pg_xact.

```
ubuntu@ubuntu:~$ sudo -u postgres cp /var/lib/postgresql/recovery.conf /var/lib/  
postgresql/11/main/recovery.conf  
ubuntu@ubuntu:~$  
ubuntu@ubuntu:~$ sudo -u postgres ls /var/lib/postgresql/11/main/  
backup_label      pg_dynshmem      pg_serial        pg_tblspc      postgresql.auto.conf  
backup_label.old  pg_logical       pg_snapshots     pg_twophase     recovery.conf  
base              pg_multixact     pg_stat          PG_VERSION  
global            pg_notify        pg_stat_tmp      pg_wal  
pg_commit_ts      pg_replslot      pg_subtrans      pg_xact  
ubuntu@ubuntu:~$
```

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'
```

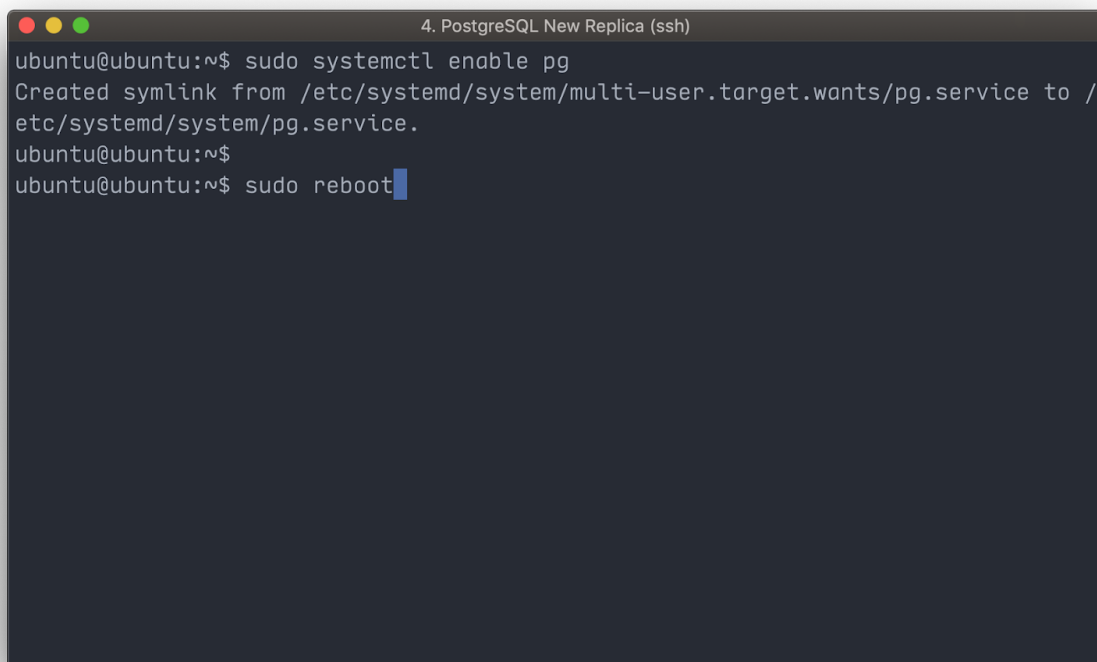
10,0-1 Top

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.

A terminal window titled "4. PostgreSQL New Replica (ssh)" is shown. The prompt is "ubuntu@ubuntu:~\$". The first command entered is "sudo systemctl enable pg", which results in the output "Created symlink from /etc/systemd/system/multi-user.target.wants/pg.service to /etc/systemd/system/pg.service." followed by a new prompt "ubuntu@ubuntu:~\$". The second command entered is "sudo reboot", followed by a new prompt "ubuntu@ubuntu:~\$".

```
4. PostgreSQL New Replica (ssh)  
ubuntu@ubuntu:~$ sudo systemctl enable pg  
Created symlink from /etc/systemd/system/multi-user.target.wants/pg.service to /  
etc/systemd/system/pg.service.  
ubuntu@ubuntu:~$  
ubuntu@ubuntu:~$ sudo reboot
```

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,

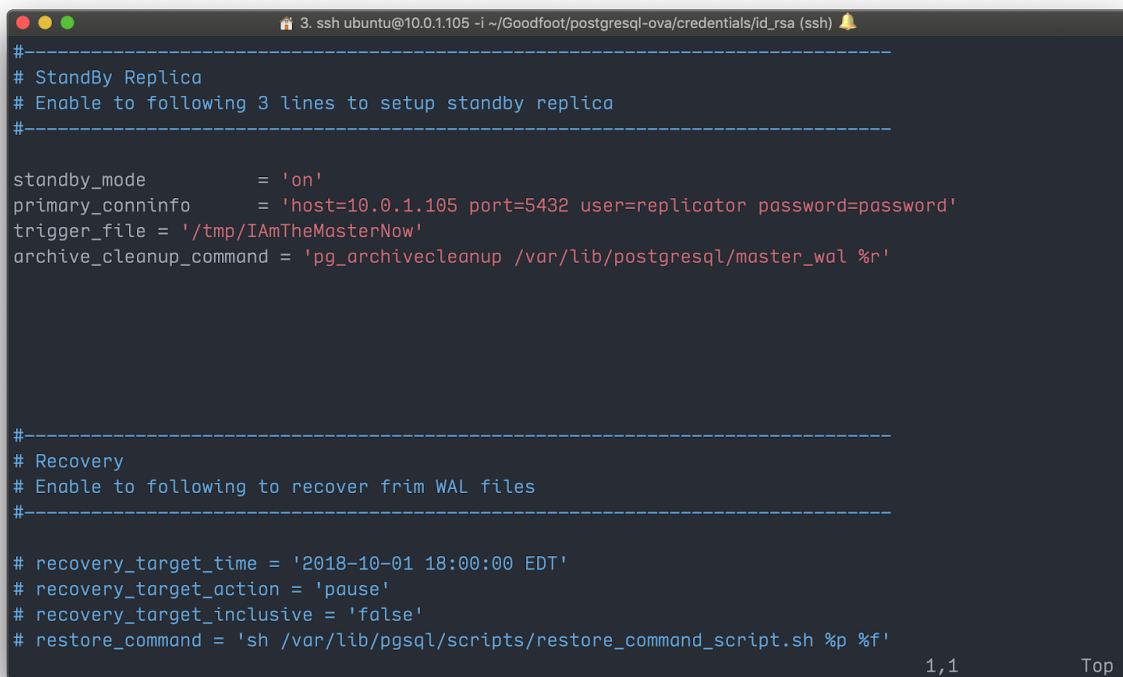
```
# Stop postgres service on Primary Server  
sudo systemctl stop pg
```

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
```

A screenshot of a terminal window with a dark background. The window title bar shows '3. ssh ubuntu@10.0.1.105 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)' and a notification icon. The terminal displays the configuration for a PostgreSQL standby replica and recovery. It includes comments and settings for standby_mode, primary_conninfo, trigger_file, archive_cleanup_command, recovery_target_time, recovery_target_action, recovery_target_inclusive, and restore_command. The bottom right corner of the terminal shows '1,1' and a 'Top' link.

```
#-----
# 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 from WAL files
#-----

# recovery_target_time = '2018-10-01 18:00:00 EDT'
# recovery_target_action = 'pause'
# recovery_target_inclusive = 'false'
# restore_command = 'sh /var/lib/pgsql/scripts/restore_command_script.sh %p %f'

1,1 Top
```

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/pgsqql/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

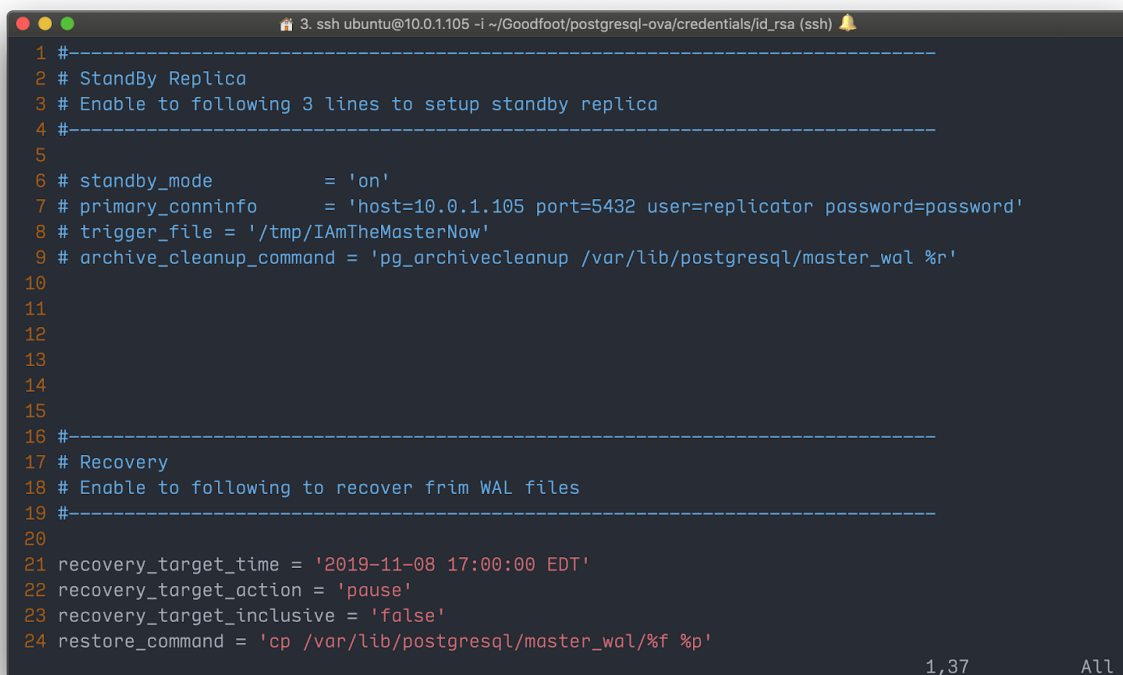
Set to **pause** the database after completing the recovery process.

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/**.



```
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 from WAL files
19 #-----
20
21 recovery_target_time = '2019-11-08 17:00:00 EDT'
22 recovery_target_action = 'pause'
23 recovery_target_inclusive = 'false'
24 restore_command = 'cp /var/lib/postgresql/master_wal/%f %p'
```

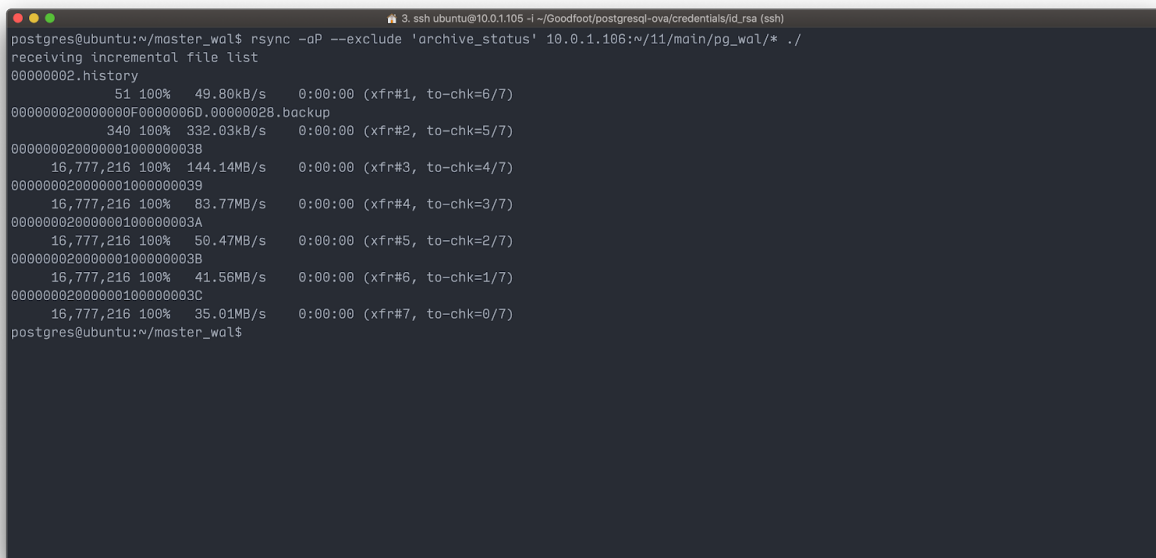
1,37 All

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/* ./
```

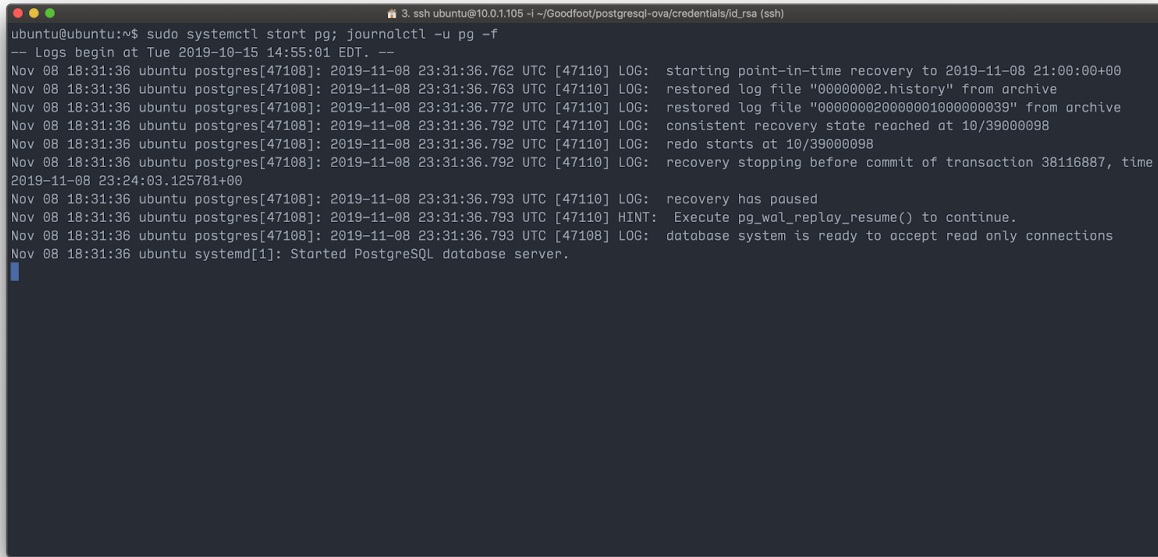


```
3. ssh ubuntu@10.0.1.105 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)  
postgres@ubuntu:~/master_wal$ rsync -aP --exclude 'archive_status' 10.0.1.106:~/11/main/pg_wal/* ./  
receiving incremental file list  
00000002.history  
 51 100% 49.80kB/s 0:00:00 (xfr#1, to-chk=6/7)  
00000020000000F00000006D.00000028.backup  
 340 100% 332.03kB/s 0:00:00 (xfr#2, to-chk=5/7)  
0000000200000010000000038  
16,777,216 100% 144.14MB/s 0:00:00 (xfr#3, to-chk=4/7)  
0000000200000010000000039  
16,777,216 100% 83.77MB/s 0:00:00 (xfr#4, to-chk=3/7)  
000000020000001000000003A  
16,777,216 100% 50.47MB/s 0:00:00 (xfr#5, to-chk=2/7)  
000000020000001000000003B  
16,777,216 100% 41.56MB/s 0:00:00 (xfr#6, to-chk=1/7)  
000000020000001000000003C  
16,777,216 100% 35.01MB/s 0:00:00 (xfr#7, to-chk=0/7)  
postgres@ubuntu:~/master_wal$
```

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

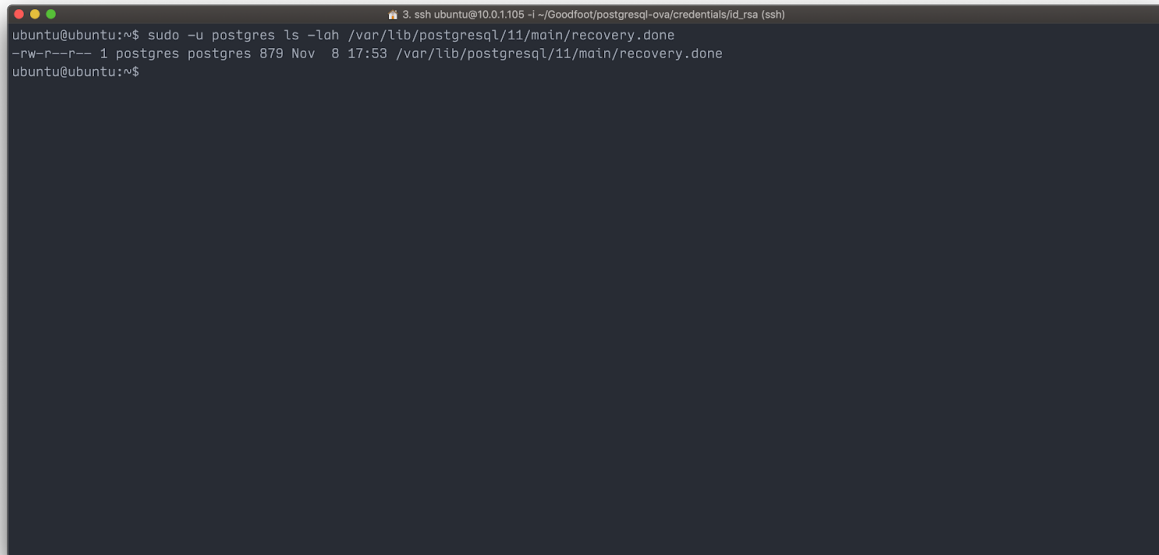
```
# Start pg service
sudo systemctl start pg
```

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

A terminal window with a dark background and light text. The title bar shows a red, yellow, and green window control icon, followed by the text '3. ssh ubuntu@10.0.1.105 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)'. The terminal content shows the command 'journalctl -u pg -f' being executed. The output displays a series of log messages from PostgreSQL, including the start of the point-in-time recovery process, the restoration of log files from the archive, the reaching of a consistent recovery state, and the start of the redo process. The logs also show the recovery pausing and then resuming, and finally, the database system becoming ready to accept read-only connections. The last line shows the systemd service starting the PostgreSQL database server.

```
ubuntu@ubuntu:~$ sudo systemctl start pg; journalctl -u pg -f
-- Logs begin at Tue 2019-10-15 14:55:01 EDT. --
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.762 UTC [47110] LOG:  starting point-in-time recovery to 2019-11-08 21:00:00+00
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.763 UTC [47110] LOG:  restored log file "00000002.history" from archive
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.772 UTC [47110] LOG:  restored log file "0000000200000001000000039" from archive
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.792 UTC [47110] LOG:  consistent recovery state reached at 10/39000098
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.792 UTC [47110] LOG:  redo starts at 10/39000098
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.792 UTC [47110] LOG:  recovery stopping before commit of transaction 38116887, time
2019-11-08 23:24:03.125781+00
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.793 UTC [47110] LOG:  recovery has paused
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.793 UTC [47110] HINT:  Execute pg_wal_replay_resume() to continue.
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.793 UTC [47108] LOG:  database system is ready to accept read only connections
Nov 08 18:31:36 ubuntu systemd[1]: Started PostgreSQL database server.
```

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

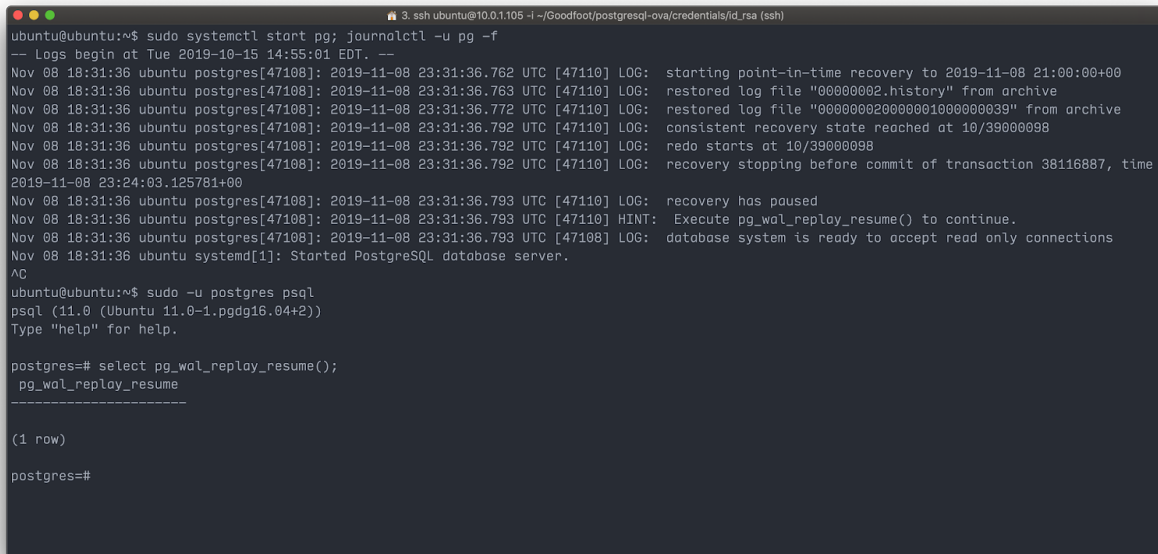
A terminal window with a dark background and light text. The window title bar shows a red, yellow, and green icon on the left, and the text '3. ssh ubuntu@10.0.1.105 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)' on the right. The terminal content shows a user prompt 'ubuntu@ubuntu:~\$' followed by the command 'sudo -u postgres ls -lah /var/lib/postgresql/11/main/recovery.done'. The output line shows file permissions '-rw-r--r--', owner '1 postgres', group 'postgres', size '879', date 'Nov 8 17:53', and path '/var/lib/postgresql/11/main/recovery.done'. The prompt 'ubuntu@ubuntu:~\$' appears again on the next line.

```
ubuntu@ubuntu:~$ sudo -u postgres ls -lah /var/lib/postgresql/11/main/recovery.done
-rw-r--r-- 1 postgres postgres 879 Nov  8 17:53 /var/lib/postgresql/11/main/recovery.done
ubuntu@ubuntu:~$
```

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();
```

A terminal window screenshot showing the process of starting and resuming a PostgreSQL database. The terminal title is '3. ssh ubuntu@10.0.1.105 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)'. The user runs 'sudo systemctl start pg; journalctl -u pg -f'. The logs show a point-in-time recovery starting at 2019-11-08 21:00:00+00, restoring log files, reaching a consistent recovery state, and pausing before a commit. The user then runs 'sudo -u postgres psql'. Inside the psql shell, they run 'select pg_wal_replay_resume();', which returns a single row. The terminal output is as follows:

```
ubuntu@ubuntu:~$ sudo systemctl start pg; journalctl -u pg -f
-- Logs begin at Tue 2019-10-15 14:55:01 EDT. --
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.762 UTC [47110] LOG:  starting point-in-time recovery to 2019-11-08 21:00:00+00
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.763 UTC [47110] LOG:  restored log file "00000002.history" from archive
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.772 UTC [47110] LOG:  restored log file "0000000200000001000000039" from archive
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.792 UTC [47110] LOG:  consistent recovery state reached at 10/39000098
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.792 UTC [47110] LOG:  redo starts at 10/39000098
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.792 UTC [47110] LOG:  recovery stopping before commit of transaction 38116887, time
2019-11-08 23:24:03.125781+00
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.793 UTC [47110] LOG:  recovery has paused
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.793 UTC [47110] HINT:  Execute pg_wal_replay_resume() to continue.
Nov 08 18:31:36 ubuntu postgres[47108]: 2019-11-08 23:31:36.793 UTC [47108] LOG:  database system is ready to accept read only connections
Nov 08 18:31:36 ubuntu systemd[1]: Started PostgreSQL database server.
^C
ubuntu@ubuntu:~$ sudo -u postgres psql
psql (11.0 (Ubuntu 11.0-1.pgdg16.04+2))
Type "help" for help.

postgres=# select pg_wal_replay_resume();
 pg_wal_replay_resume 
-----
(1 row)

postgres=#
```

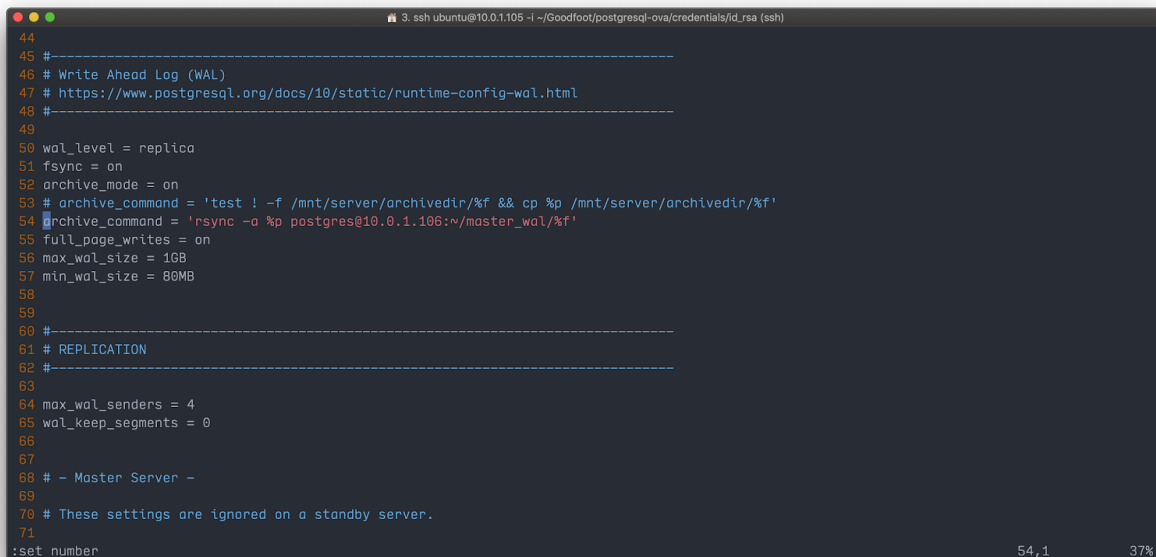
After recovery is completed follow the steps to convert Replica server to Primary server and Primary server to Replica server.

Convert Replica to Primary

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

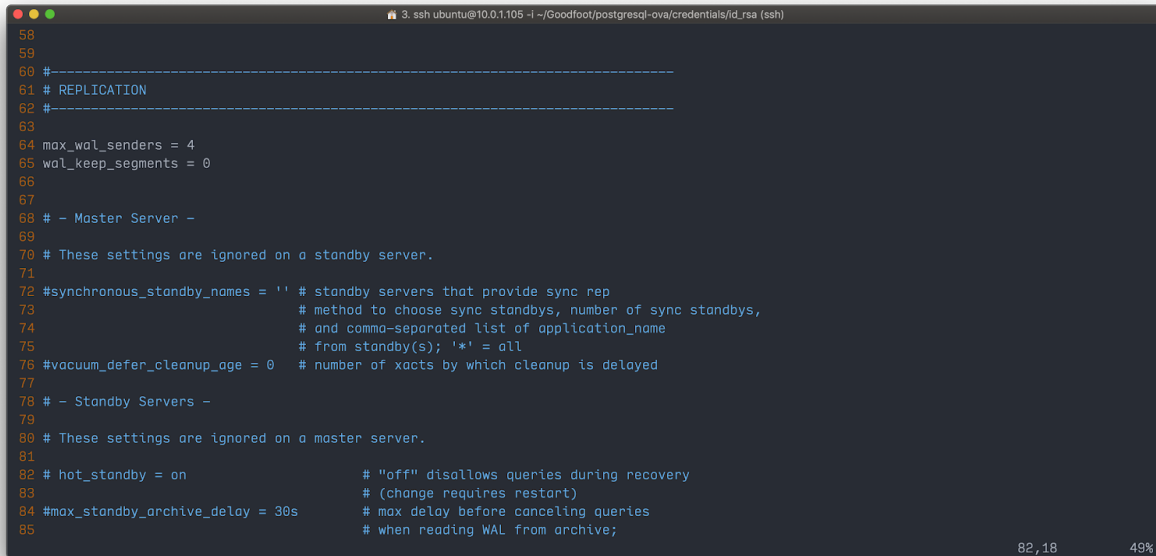
For example if the **Primary** server IP is **10.0.1.106** and **Replica** server IP is **10.0.1.105**.
Un-comment **archive_command** on line 54, to promote the Replica server as the new Primary server

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



```
3. ssh ubuntu@10.0.1.105 -i ~/Goodfoot/postgresql-ova/credentials/id_rsa (ssh)
44 #
45 #-----
46 # Write Ahead Log (WAL)
47 # https://www.postgresql.org/docs/10/static/runtime-config-wal.html
48 #-----
49
50 wal_level = replica
51 fsync = on
52 archive_mode = on
53 # archive_command = 'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/server/archivedir/%f'
54 archive_command = 'rsync -a %p postgres@10.0.1.106:~/master_wal/%f'
55 full_page_writes = on
56 max_wal_size = 1GB
57 min_wal_size = 80MB
58
59 #-----
60 # REPLICATION
61 #-----
62 #
63
64 max_wal_senders = 4
65 wal_keep_segments = 0
66
67
68 # - Master Server -
69
70 # These settings are ignored on a standby server.
71
:set number                                     54,1                                     37%
```


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



```
58
59
60 #-----
61 # REPLICATION
62 #-----
63
64 max_wal_senders = 4
65 wal_keep_segments = 0
66
67
68 # - Master Server -
69
70 # These settings are ignored on a standby server.
71
72 #synchronous_standby_names = '' # standby servers that provide sync rep
73                                # method to choose sync standbys, number of sync standbys,
74                                # and comma-separated list of application_name
75                                # from standby(s); '*' = all
76 #vacuum_defer_cleanup_age = 0  # number of xacts by which cleanup is delayed
77
78 # - Standby Servers -
79
80 # These settings are ignored on a master server.
81
82 # hot_standby = on              # "off" disallows queries during recovery
83                                # (change requires restart)
84 #max_standby_archive_delay = 30s # max delay before canceling queries
85                                # when reading WAL from archive;
```

82,18 49%

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
```

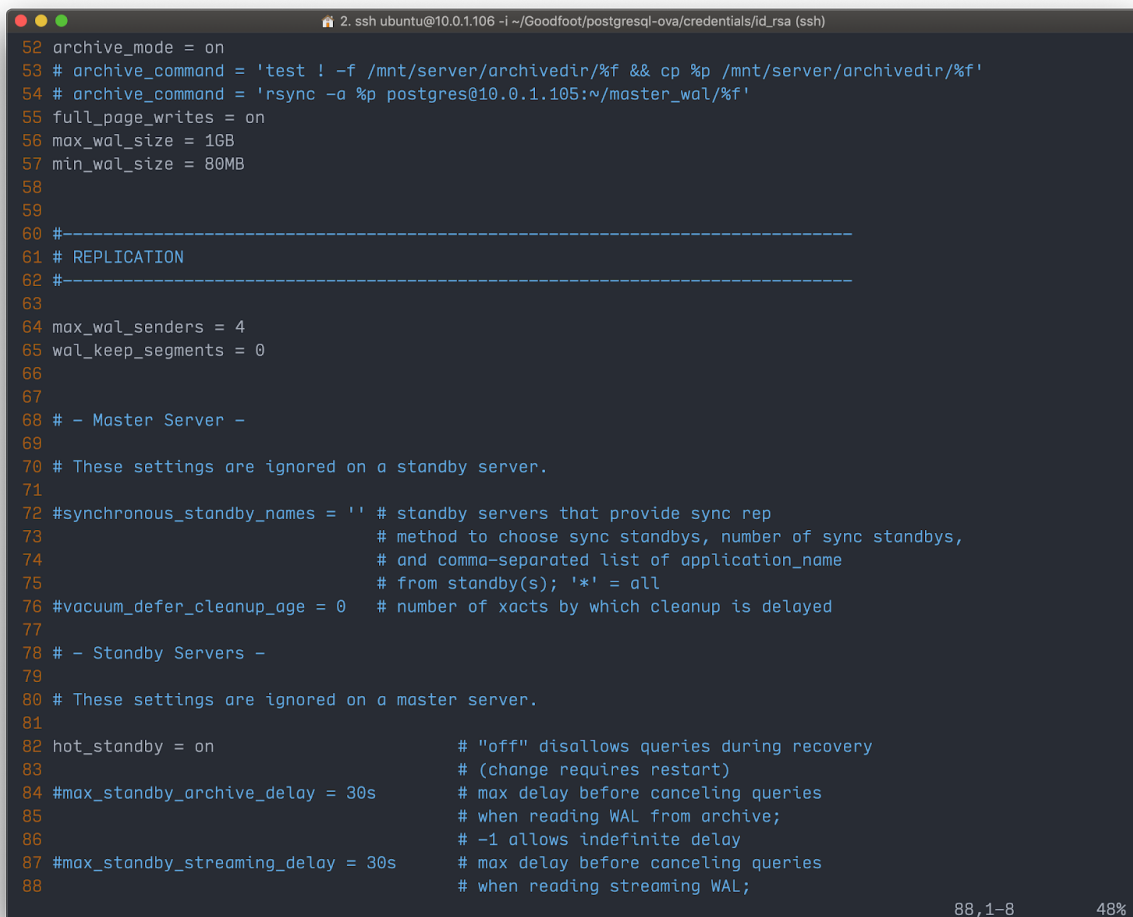
Convert Primary to Replica

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'
```

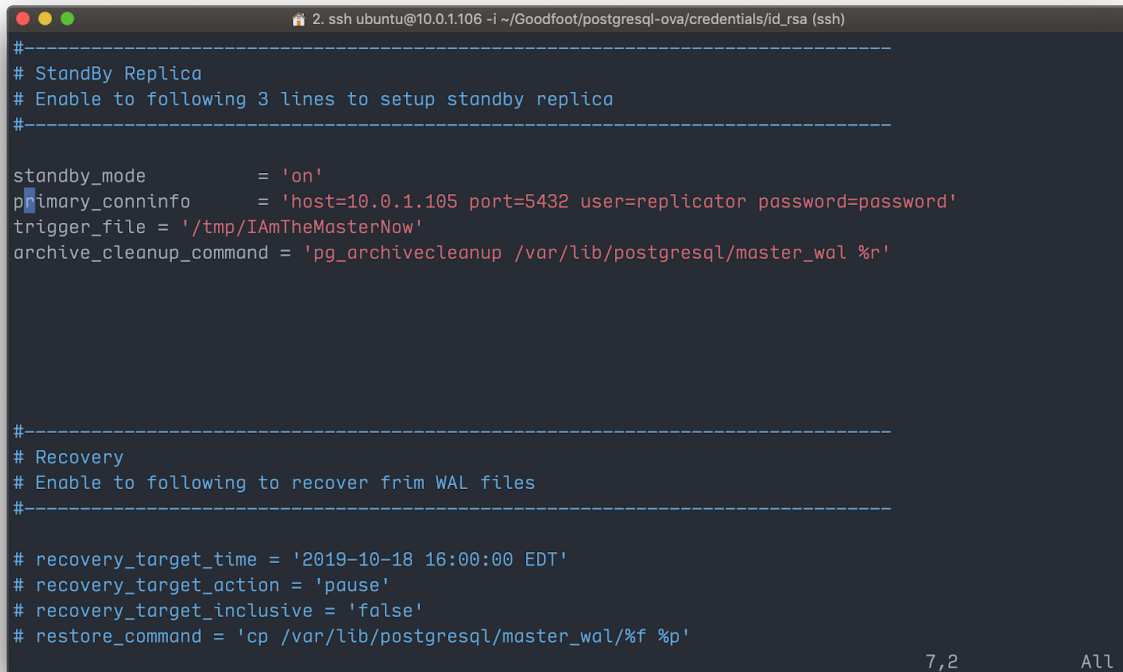
Un-comment hot_standby on line 82



```
52 archive_mode = on
53 # archive_command = 'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/server/archivedir/%f'
54 # archive_command = 'rsync -a %p postgres@10.0.1.105:~/master_wal/%f'
55 full_page_writes = on
56 max_wal_size = 1GB
57 min_wal_size = 80MB
58
59
60 #-----
61 # REPLICATION
62 #-----
63
64 max_wal_senders = 4
65 wal_keep_segments = 0
66
67
68 # - Master Server -
69
70 # These settings are ignored on a standby server.
71
72 #synchronous_standby_names = '' # standby servers that provide sync rep
73                                # method to choose sync standbys, number of sync standbys,
74                                # and comma-separated list of application_name
75                                # from standby(s); '*' = all
76 #vacuum_defer_cleanup_age = 0 # number of xacts by which cleanup is delayed
77
78 # - Standby Servers -
79
80 # These settings are ignored on a master server.
81
82 hot_standby = on                # "off" disallows queries during recovery
83                                # (change requires restart)
84 #max_standby_archive_delay = 30s # max delay before canceling queries
85                                # when reading WAL from archive;
86                                # -1 allows indefinite delay
87 #max_standby_streaming_delay = 30s # max delay before canceling queries
88                                # when reading streaming WAL;
```

88,1-8 48%

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**.

A terminal window with a dark background and light blue text. The title bar shows a home icon, the user '2', the host 'ssh ubuntu@10.0.1.106', and the path '~ / Goodfoot/postgresql-ova/credentials/id_rsa (ssh)'. The terminal content shows the generation of a PostgreSQL recovery.conf file. It starts with a separator line, followed by a comment '# StandBy Replica' and instructions to enable the following 3 lines. Then, four configuration lines are shown: 'standby_mode = \'on\'', 'primary_conninfo = \'host=10.0.1.105 port=5432 user=replicator password=password\'', 'trigger_file = \'/tmp/IAMTheMasterNow\'', and 'archive_cleanup_command = \'pg_archivecleanup /var/lib/postgresql/master_wal %r\''. Another separator line follows, then a comment '# Recovery' and instructions to enable the following lines to recover from WAL files. Finally, four more configuration lines are shown: 'recovery_target_time = \'2019-10-18 16:00:00 EDT\'', 'recovery_target_action = \'pause\'', 'recovery_target_inclusive = \'false\'', and 'restore_command = \'cp /var/lib/postgresql/master_wal/%f %p\''. At the bottom right, '7,2' and 'All' are visible.

```
#-----  
# 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 from 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'
```

Restart the now Primary and Replica servers and following the instructions in [Replication Status](#) section to verify everything is working correctly.