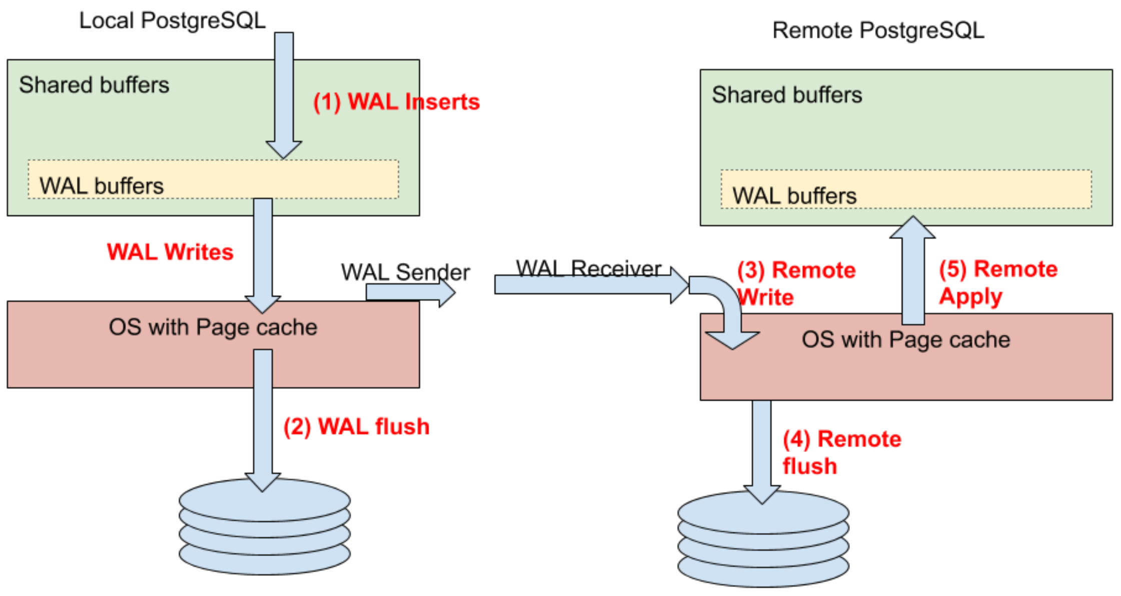
Replication

For *streaming replication*, servers will be either a primary/master or a replication/standby server. Masters can send data, while standbys are always receivers of replicated data. Parameters are mainly for sending and standby servers, though some parameters have meaning only on the master server.



Main Process:

```
Primary/Master Service
    ↓ use replication role, backup_replicator
    ↓ streaming replication + pg_basebackup
Replica/Standby Service
    ↓ use replication role, backup_user
```

Install environment

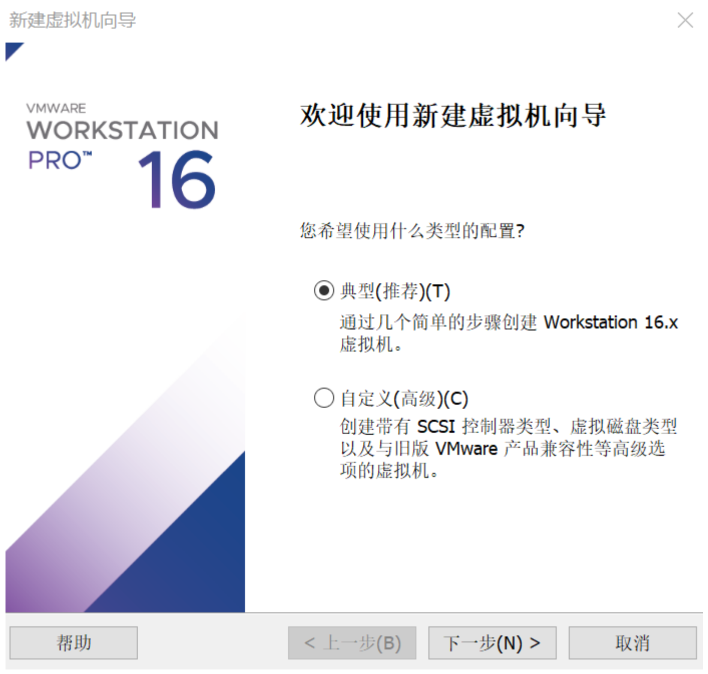
We need two different IPs to configure the master-standby database. For convenience, we will use virtual machines to achieve this. If you already have two devices that can be set with different IPs, please skip the steps for installing virtual machines.

Install virtual machine

1. Download virtual machine application **VMware workstation** refer to the blog . This application offer the free license to personal user. Install this app by yourself.
2. Download ubuntu, choose any version as you like depend on residual space of your computer. Here we use ubuntu 18.04.06 as example. This app will be installed in virtual machine.

3. New a virtual machine(totally, we need two, one is **master**, another is **standby**). Open software **VMware workstation**, click `File(F)`, and choose `Create a new virtual machine(N)...`

4. Follow the guide, complete the app installation. Here are some install steps. After the installation, open this virtual machine.



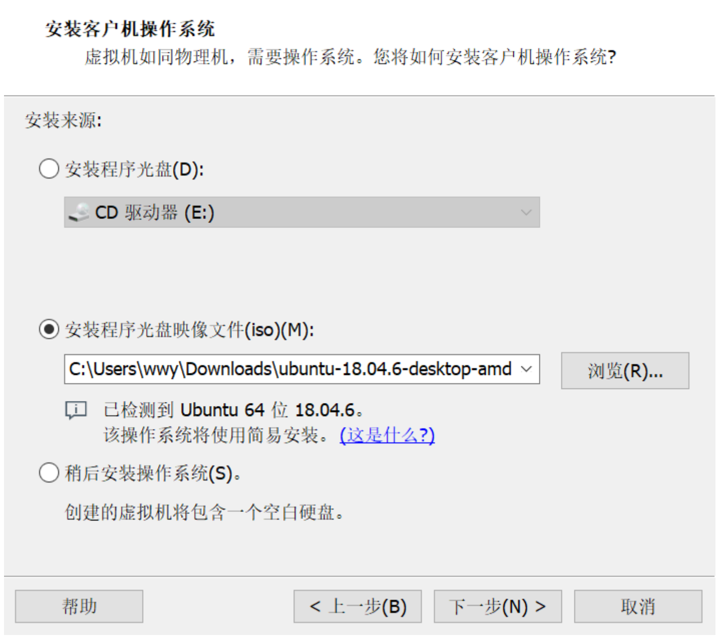
新建虚拟机向导

欢迎使用新建虚拟机向导

您希望使用什么类型的配置?

[✓ 典型(推荐)(T)]
通过几个简单的步骤创建 Workstation 16.x 虚拟机。

自定义(高级)(C)
创建带有 SCSI 控制器类型、虚拟磁盘类型以及与旧版 VMware 产品兼容性等高级选项的虚拟机。



安装客户机操作系统
虚拟机如同物理机，需要操作系统。您将如何安装客户机操作系统?

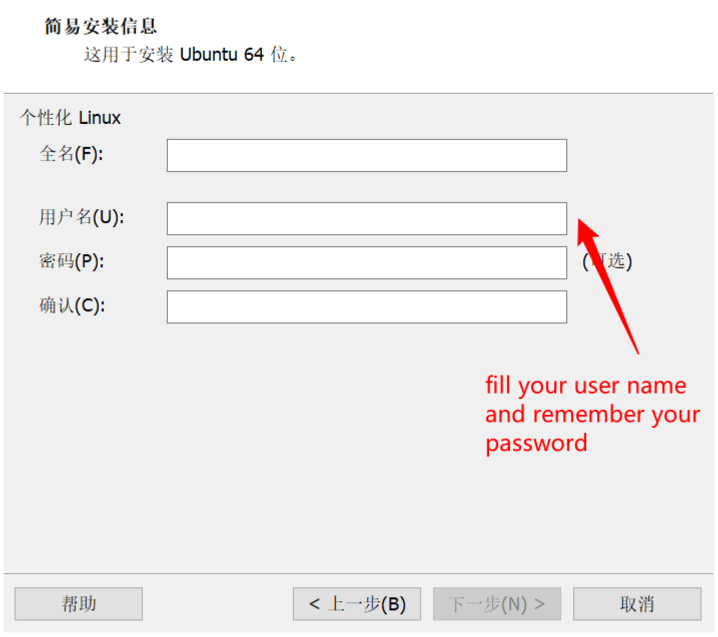
安装来源:

安装程序光盘(D):
CD 驱动器 (E:)

[✓ 安装程序光盘映像文件(iso)(M):]
C:\Users\wwy\Downloads\ubuntu-18.04.6-desktop-amd

已检测到 Ubuntu 64 位 18.04.6。
该操作系统将使用简易安装。(这是什么?)

稍后安装操作系统(S)。
创建的虚拟机将包含一个空白硬盘。



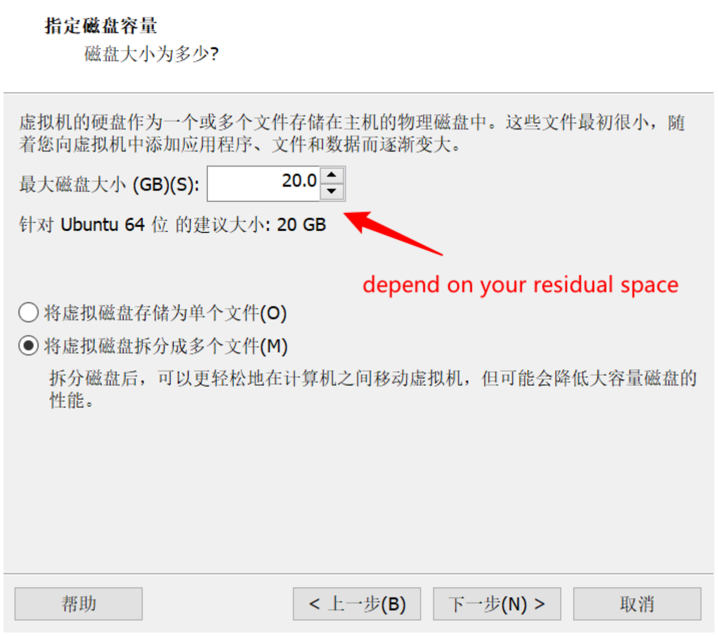
简易安装信息

这用于安装 Ubuntu 64 位。

个性化 Linux

全名(F):	
用户名(U):	
密码(P):	(可选)
确认(C):	

fill your user name and remember your password



指定磁盘容量

磁盘大小为多少?

虚拟机的硬盘作为一个或多个文件存储在主机的物理磁盘中。这些文件最初很小，随着您向虚拟机中添加应用程序、文件和数据而逐渐变大。

最大磁盘大小 (GB)(S): 20.0

针对 Ubuntu 64 位 的建议大小: 20 GB

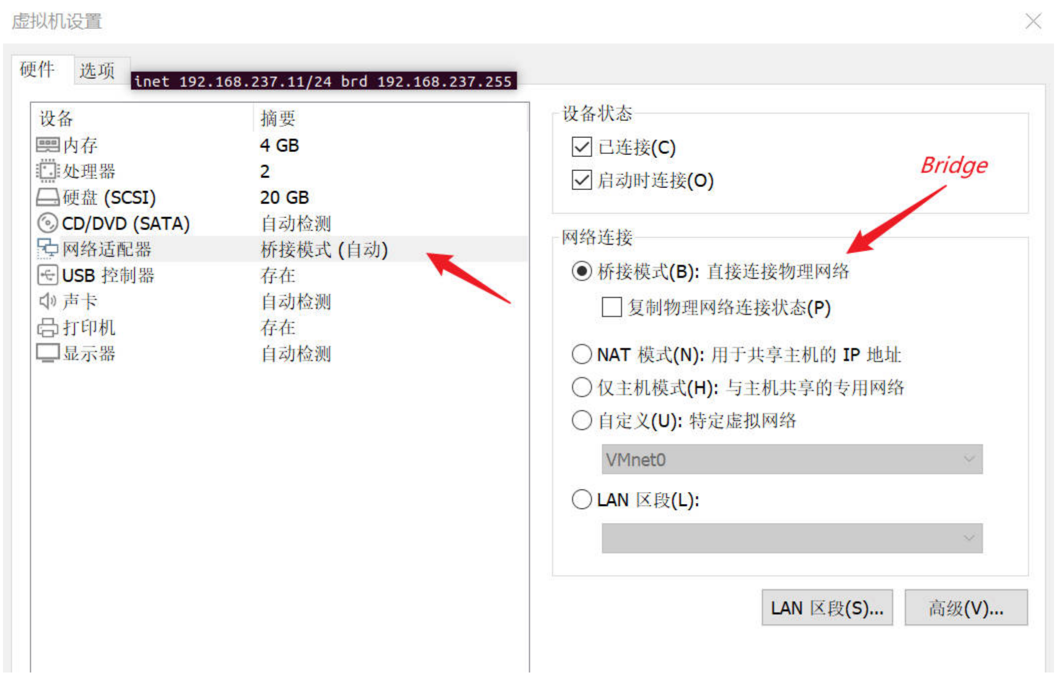
depend on your residual space

○ 将虚拟磁盘存储为单个文件(O)

● 将虚拟磁盘拆分成多个文件(M)

拆分磁盘后，可以更轻松地在计算机之间移动虚拟机，但可能会降低大容量磁盘的性能。

5. Open the **setting** of this two virtual machines, and make the following settings. Set the virtual machine to **bridge mode**, where the physical network card of the host machine is bridged with the virtual network card of the virtual machine. The virtual machine obtains an IP address from the LAN DHCP server (instead of a private IP address assigned by the host machine's NAT), enabling direct communication between the physical machine and the virtual machine.


虚拟机设置

硬件 选项

```
inet 192.168.237.11/24 brd 192.168.237.255
```

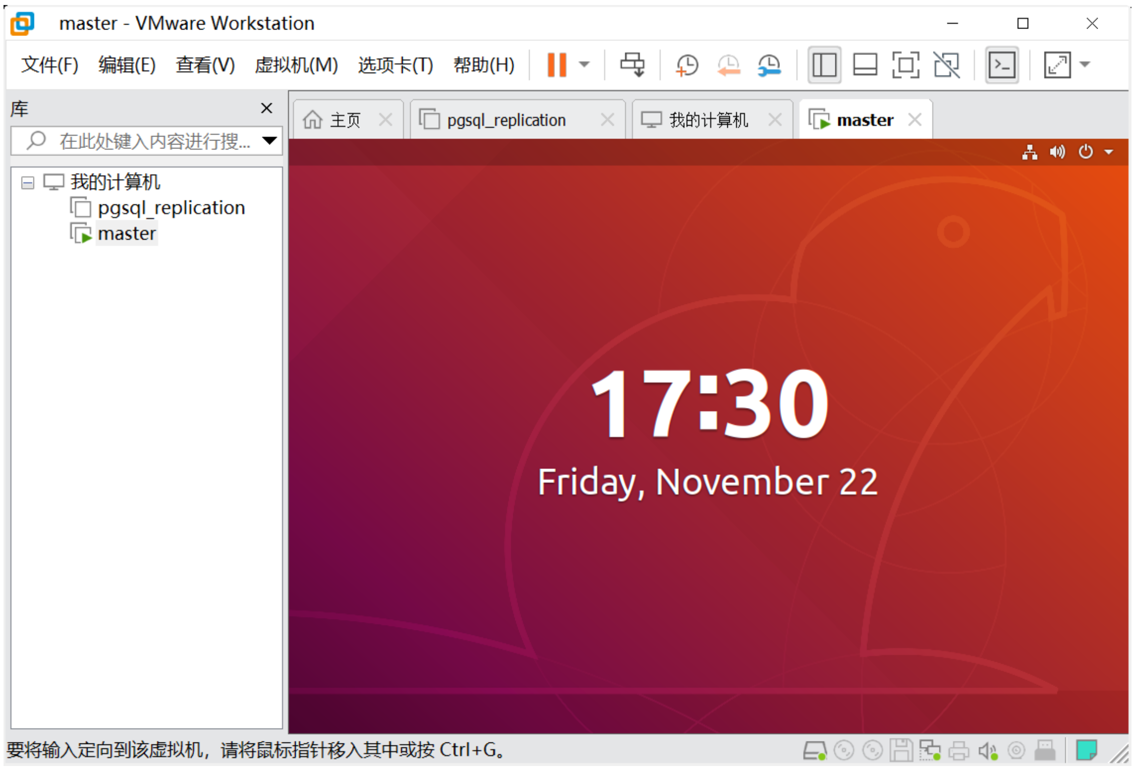
设备	摘要
内存	4 GB
处理器	2
硬盘 (SCSI)	20 GB
CD/DVD (SATA)	自动检测
网络适配器	桥接模式 (自动)
USB 控制器	存在
声卡	自动检测
打印机	存在
显示器	自动检测

设备状态
- ☑ 已连接(C)
- ☑ 启动时连接(O)

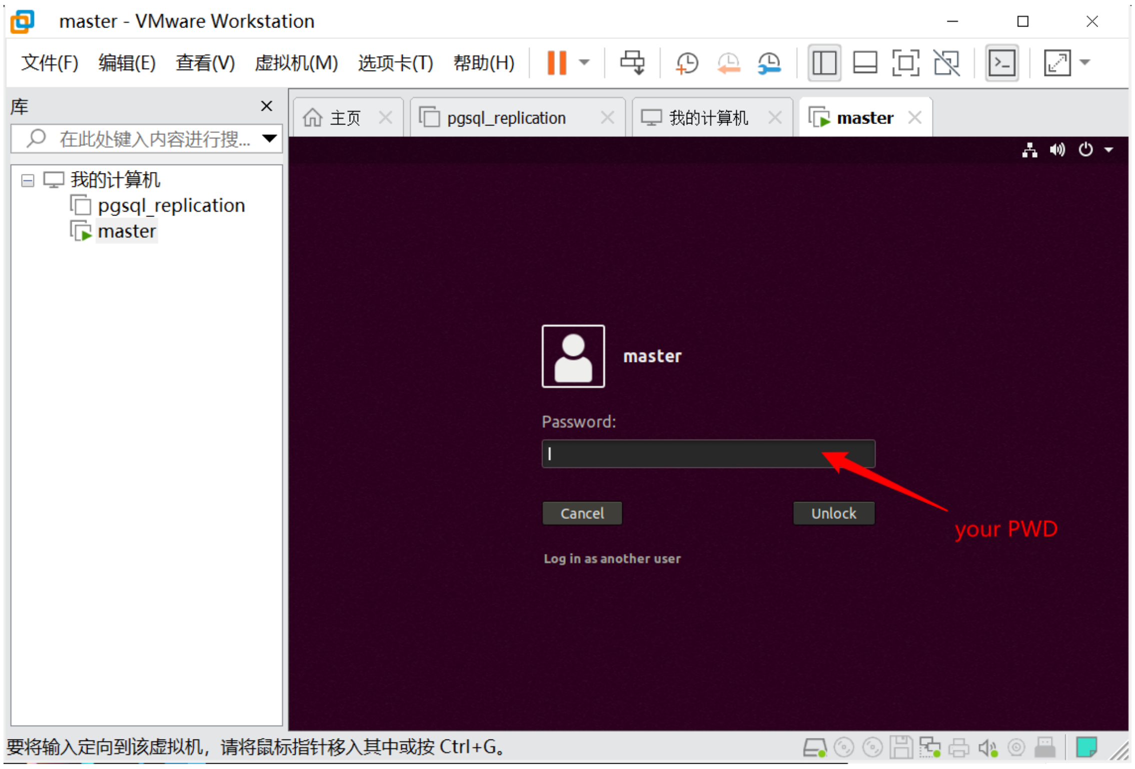
网络连接
- ● 桥接模式(B): 直接连接物理网络
  - ☐ 复制物理网络连接状态(P)
- ○ NAT 模式(N): 用于共享主机的 IP 地址
- ○ 仅主机模式(H): 与主机共享的专用网络
- ○ 自定义(U): 特定虚拟网络 VMnet0
- ○ LAN 区段(L):

Install postgres

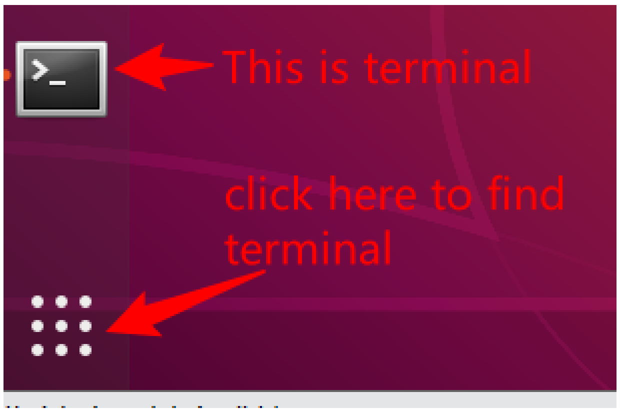
1. Open this virtual machine. Find terminal.



要将输入定向到该虚拟机，请将鼠标指针移入其中或按 Ctrl+G。

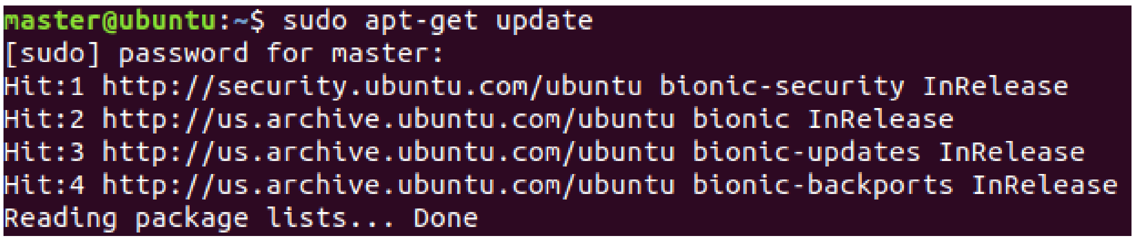


要将输入定向到该虚拟机，请将鼠标指针移入其中或按 Ctrl+G。



2. Update the `apt-get`. Here `sudo` means using the rights of administrator, what's more `sudo su` means log in with administrator. Then you need to type your password. It's no worry to see no sign when you type the password, because here is a word in ubuntu,*"No news is good news"*.

```
sudo apt-get update
```



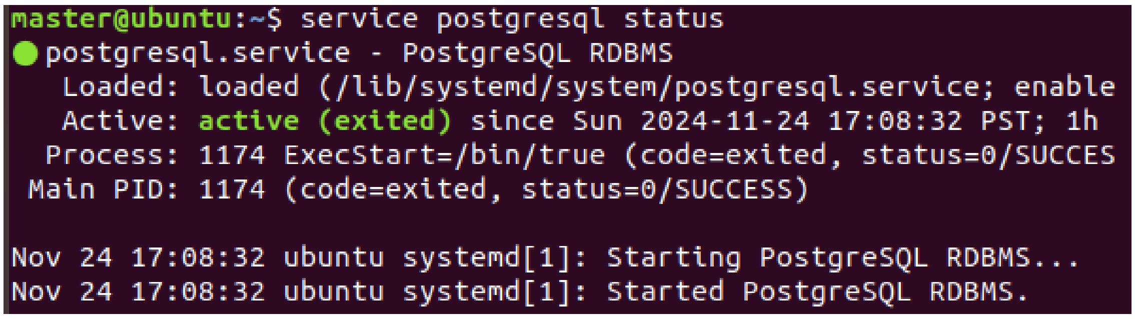
```
master@ubuntu:~$ sudo apt-get update
[sudo] password for master:
Hit:1 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
```

3. Install postgres. When the application ask you if execute this installation, remember to type 'Y'.

```
sudo apt-get install postgresql
```

4. Test if the installation is successful.

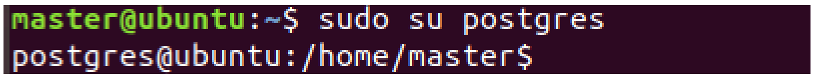
```
service postgresql status
```



```
master@ubuntu:~$ service postgresql status
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enable
   Active: active (exited) since Sun 2024-11-24 17:08:32 PST; 1h
  Process: 1174 ExecStart=/bin/true (code=exited, status=0/SUCCES
 Main PID: 1174 (code=exited, status=0/SUCCESS)

Nov 24 17:08:32 ubuntu systemd[1]: Starting PostgreSQL RDBMS...
Nov 24 17:08:32 ubuntu systemd[1]: Started PostgreSQL RDBMS.
```

5. There is a default role in postgresql: *postgres*. Log in with this role.

```
sudo su postgres
```



```
master@ubuntu:~$ sudo su postgres
postgres@ubuntu:/home/master$
```

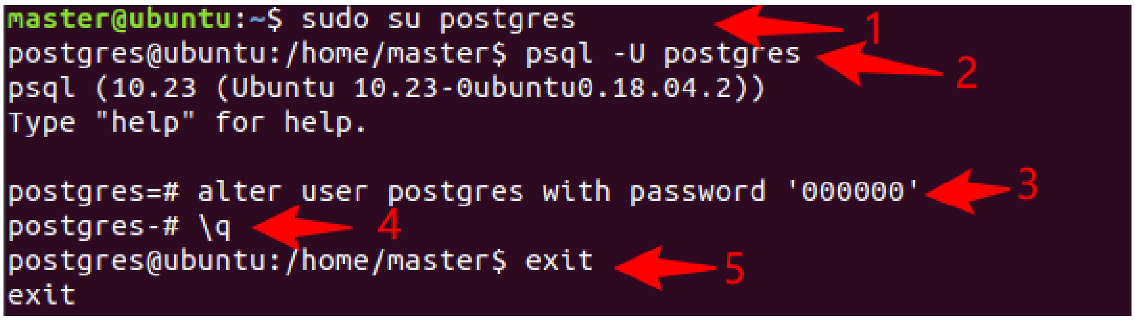
6. Modify the password of *postgres*. The default password of *postgres* is null. We log in *postgres* with null pwd.

```
psql -U postgres
```

Then modify the password

```
alter user postgres with password '000000'
```

`\q` and `exit` are all log out commond using in different situation. Here we use `\q` to log out psql



```
master@ubuntu:~$ sudo su postgres
postgres@ubuntu:/home/master$ psql -U postgres
psql (10.23 (Ubuntu 10.23-0ubuntu0.18.04.2))
Type "help" for help.

postgres=# alter user postgres with password '000000'
postgres-# \q
postgres@ubuntu:/home/master$ exit
exit
```

7. Add the path to system file. Find the path first using `locate pg_ctl` to get the path of postgres. Then open *~/.bash_profile* using `vi` command.

```
vi ~/.bash_profile
```

8. Search the usage of application **vim** by yourself. Here type `i` first for 'insert', then you can type follow sentences into *.bash_profile*. Here use your own path instead.

```
PATH=$PATH:$HOME/.local/bin:$HOME/bin:/usr/lib/postgresql/16/bin
export PATH
export PGDATA=/etc/postgresql/16/main
```

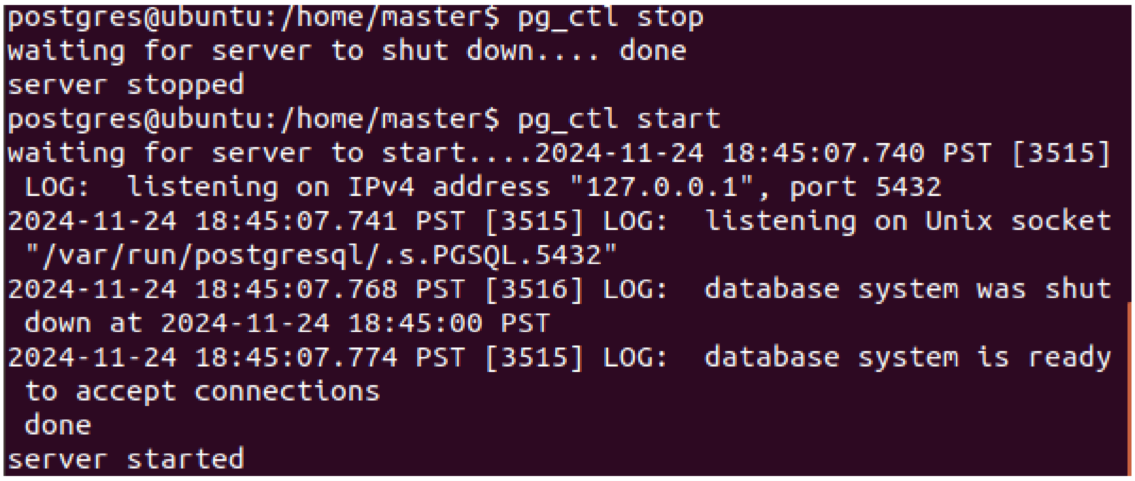
9. Press **Esc**, then type `:wq`, which means write and quit.

10. Activate system file.

```
source ~/.bash_profile
```

11. Test the following command, if there is no error, you get success.

```
pg_ctl stop
pg_ctl start
```



```
postgres@ubuntu:/home/master$ pg_ctl stop
waiting for server to shut down.... done
server stopped
postgres@ubuntu:/home/master$ pg_ctl start
waiting for server to start....2024-11-24 18:45:07.740 PST [3515]
 LOG:  listening on IPv4 address "127.0.0.1", port 5432
2024-11-24 18:45:07.741 PST [3515] LOG:  listening on Unix socket
 "/var/run/postgresql/.s.PGSQL.5432"
2024-11-24 18:45:07.768 PST [3516] LOG:  database system was shut
 down at 2024-11-24 18:45:00 PST
2024-11-24 18:45:07.774 PST [3515] LOG:  database system is ready
 to accept connections
 done
server started
```

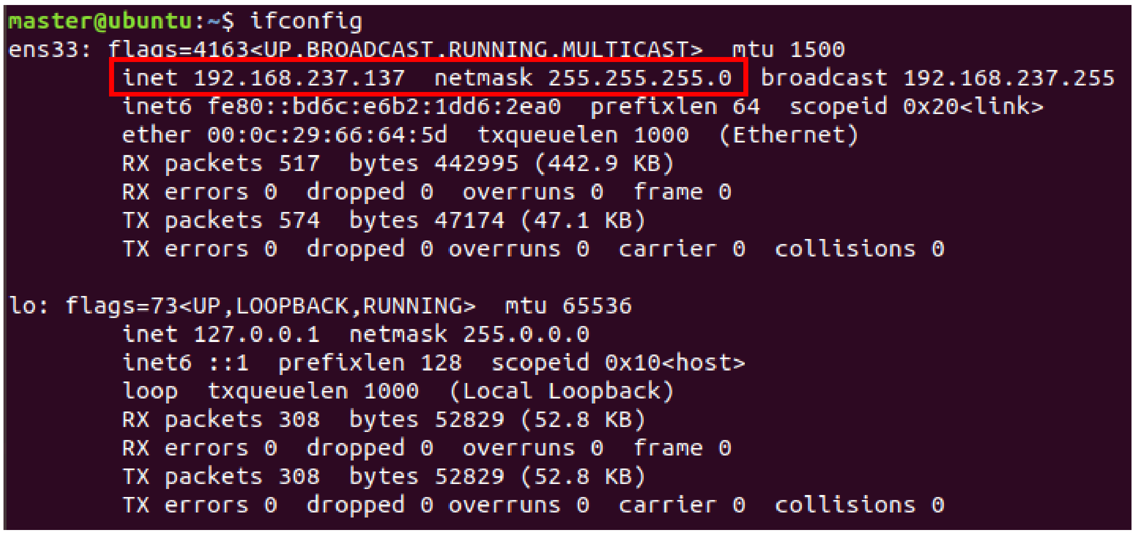
12. Create the second virtual machine using the same method as above.

Set IP connection

Here we want to make connection between two virtual machines with IP. **VMware workstation** offer two network adapter. We choose **VMnet8** to make this connection.

1. Check the VMnet8' IP of your own. If there is no net-tools, use command `sudo apt install net-tools` install one first.

```
ifconfig
```



```
master@ubuntu:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.237.137  netmask 255.255.255.0  broadcast 192.168.237.255
        inet6 fe80::bd6c:e6b2:1dd6:2ea0  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:66:64:5d  txqueuelen 1000  (Ethernet)
        RX packets 517  bytes 442995 (442.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 574  bytes 47174 (47.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 308  bytes 52829 (52.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 308  bytes 52829 (52.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

2. Open the network profile of **master**.

```
sudo vi /etc/network/interfaces
```

3. Add(`i`) the static IP to file, and save(`Esc`->`:wq`). Notice the IP added and **VMnet8** should be in the same network segment.

```
auto ens33
iface ens33 inet static
address 192.168.237.11 #same network segment with VMnet
gateway 192.168.237.1
netmask 255.255.255.0
```

4. Set the network of **replication** in the same way, with a different IP in the same network segment of **VMnet8**.

```
auto ens33
iface ens33 inet static
address 192.168.237.12 #same network segment with VMnet
gateway 192.168.237.1
netmask 255.255.255.0
```

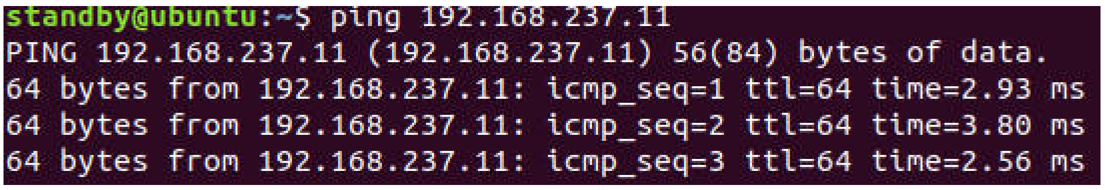
5. Ping each other. Change the detail IP into your own. Here in my computer

   For master(Blue):

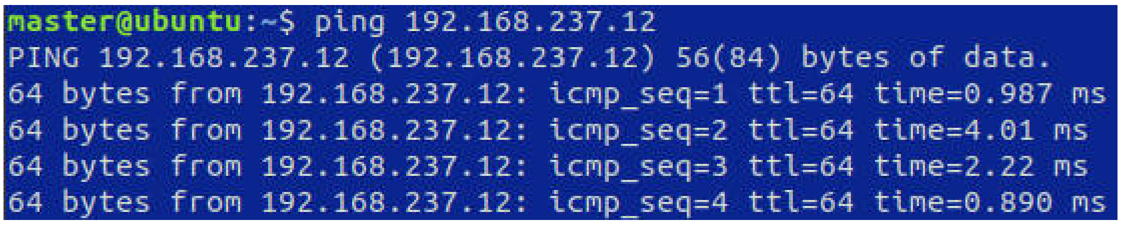
```
ping 192.168.237.11
```

   For standby(Black):

```
ping 192.168.237.12
```



```
standby@ubuntu:~$ ping 192.168.237.11
PING 192.168.237.11 (192.168.237.11) 56(84) bytes of data.
64 bytes from 192.168.237.11: icmp_seq=1 ttl=64 time=2.93 ms
64 bytes from 192.168.237.11: icmp_seq=2 ttl=64 time=3.80 ms
64 bytes from 192.168.237.11: icmp_seq=3 ttl=64 time=2.56 ms
```



```
master@ubuntu:~$ ping 192.168.237.12
PING 192.168.237.12 (192.168.237.12) 56(84) bytes of data.
64 bytes from 192.168.237.12: icmp_seq=1 ttl=64 time=0.987 ms
64 bytes from 192.168.237.12: icmp_seq=2 ttl=64 time=4.01 ms
64 bytes from 192.168.237.12: icmp_seq=3 ttl=64 time=2.22 ms
64 bytes from 192.168.237.12: icmp_seq=4 ttl=64 time=0.890 ms
```

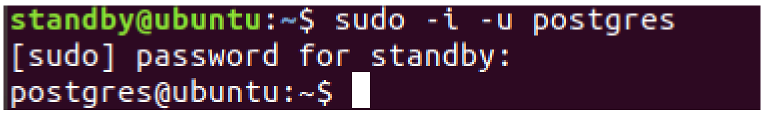
   * If you still can not connect two IPs, please search the internet for more help.

Create Replication Role

Step1 Enter PostgreSQL

For master and standby: Enter PostgreSQL use default user of postgresql.

```
sudo -i -u postgres
```

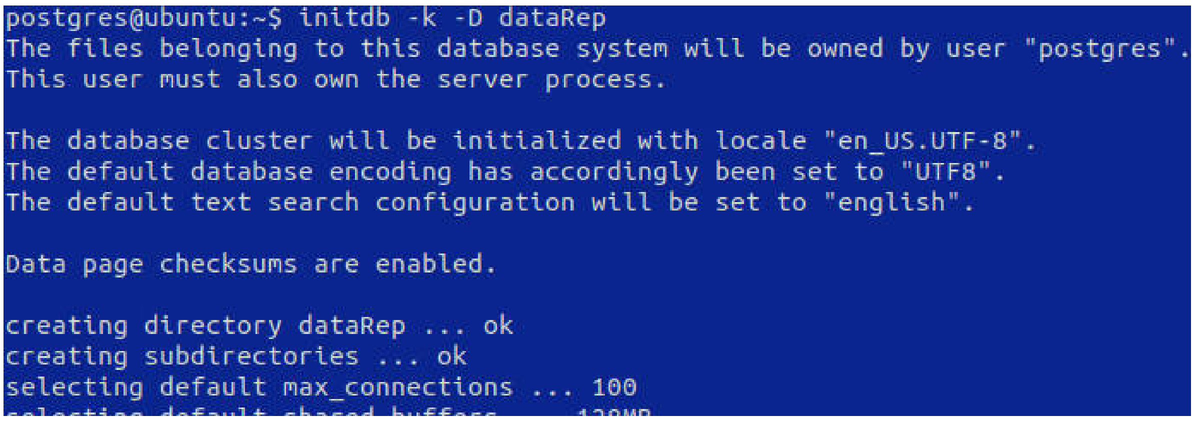


```
standby@ubuntu:~$ sudo -i -u postgres
[sudo] password for standby:
postgres@ubuntu:~$
```

Step2 Master create DB

For master: Create a new database for replication

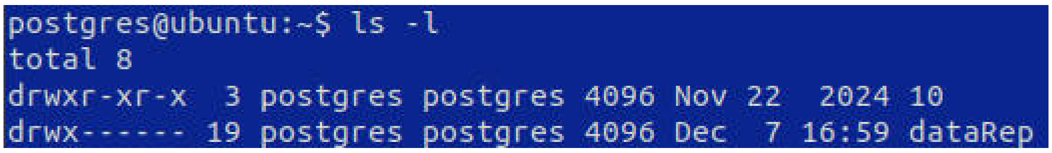
```
initdb -k -D dataRep
```



```
postgres@ubuntu:~$ initdb -k -D dataRep
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale "en_US.UTF-8".
The default database encoding has accordingly been set to "UTF8".
The default text search configuration will be set to "english".

Data page checksums are enabled.

creating directory dataRep ... ok
creating subdirectories ... ok
selecting default max_connections ... 100
```

For master: Check the database created in last step

```
ls -l
```

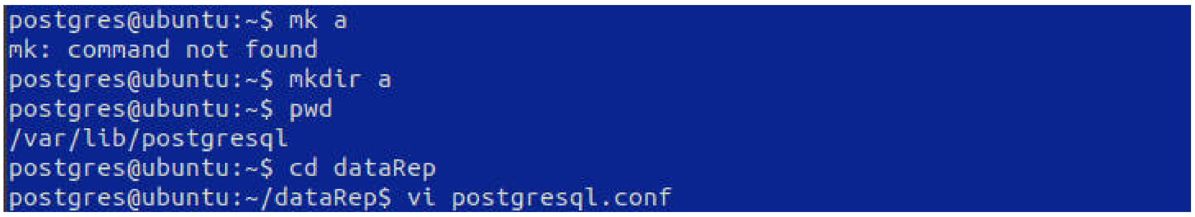


```
postgres@ubuntu:~$ ls -l
total 8
drwxr-xr-x  3 postgres postgres 4096 Nov 22  2024 10
drwx------ 19 postgres postgres 4096 Dec  7 16:59 dataRep
```

Step3 Change the setting of database

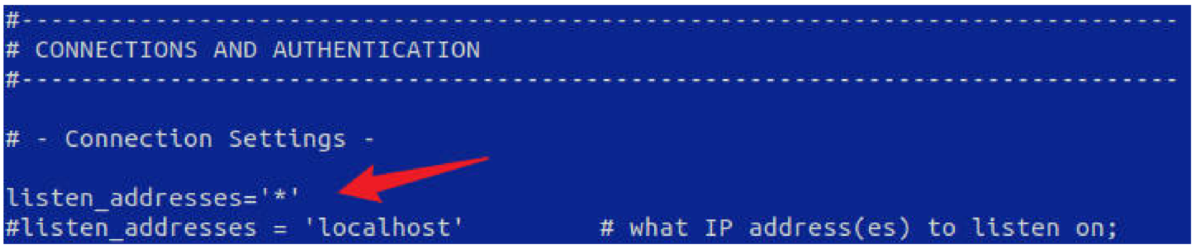
For master: create new folder `a`, and open file `postgresql.conf` via document tool `vim`. If you do not know the commands of `vim`, search on the internet.

```
mkdir a
pwd
cd dataRep
vi postgresql.conf
```

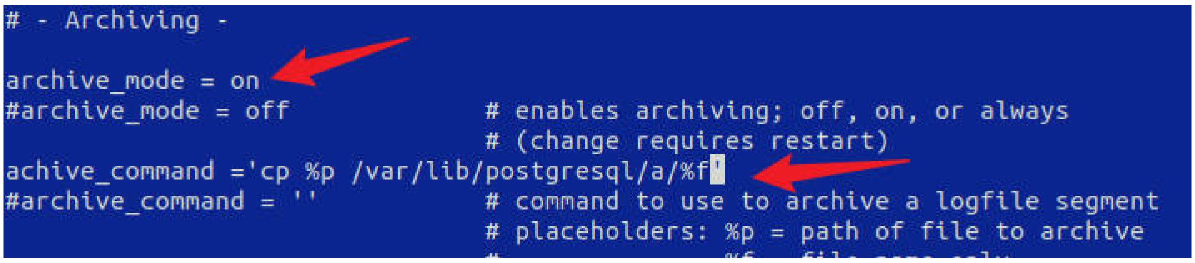


```
postgres@ubuntu:~$ mk a
mk: command not found
postgres@ubuntu:~$ mkdir a
postgres@ubuntu:~$ pwd
/var/lib/postgresql
postgres@ubuntu:~$ cd dataRep
postgres@ubuntu:~/dataRep$ vi postgresql.conf
```

For master: Add setting in `postgresql.conf`

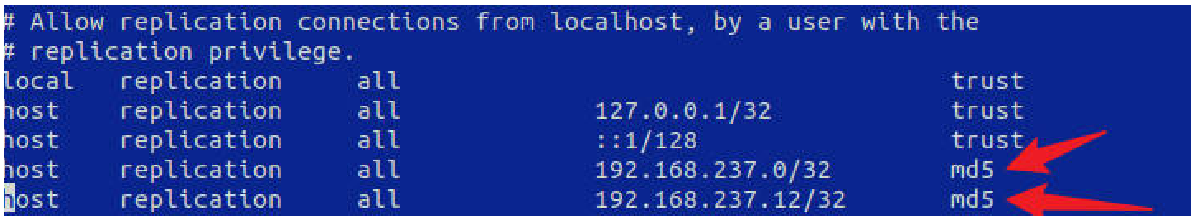


```
#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

listen_addresses='*'
#listen_addresses = 'localhost'         # what IP address(es) to listen on;
```



```
# - Archiving -

archive_mode = on
#archive_mode = off                # enables archiving; off, on, or always
                                   # (change requires restart)
achive_command ='cp %p /var/lib/postgresql/a/%f'
#archive_command = ''              # command to use to archive a logfile segment
                                   # placeholders: %p = path of file to archive
```

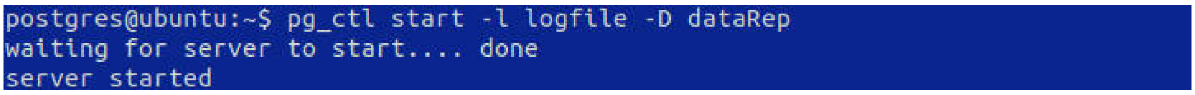
For master: Add settings in `pg_hba.conf`.



```
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     trust
host    replication     all             127.0.0.1/32            trust
host    replication     all             ::1/128                 trust
host    replication     all             192.168.237.0/32        md5
host    replication     all             192.168.237.12/32       md5
```

For master: Change the path and start the listening service

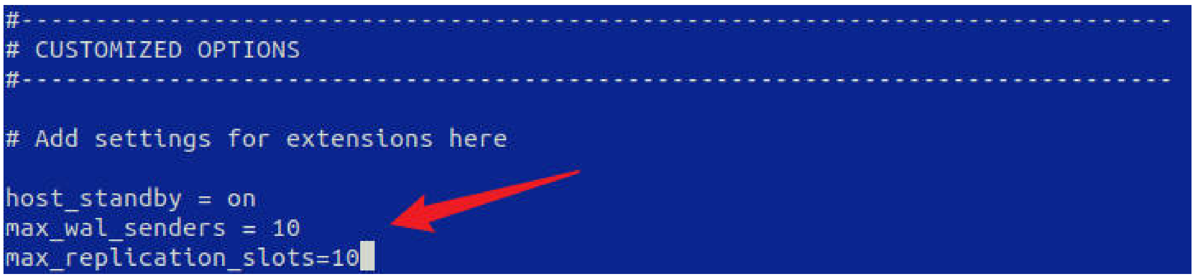
```
cd
pg_ctl -D dataRep -l logfile start
```



```
postgres@ubuntu:~$ pg_ctl start -l logfile -D dataRep
waiting for server to start.... done
server started
```

For master: Stop the listening service, add settings in `postgresql.conf`, then start the listening service again.

```
pg_ctl stop -D dataRep
vi dataRep/postgresql.conf
```



```
#------------------------------------------------------------------------------
# CUSTOMIZED OPTIONS
#------------------------------------------------------------------------------

# Add settings for extensions here

host_standby = on
max_wal_senders = 10
max_replication_slots=10
```

```
cd
pg_ctl -D dataRep -l logfile start
```

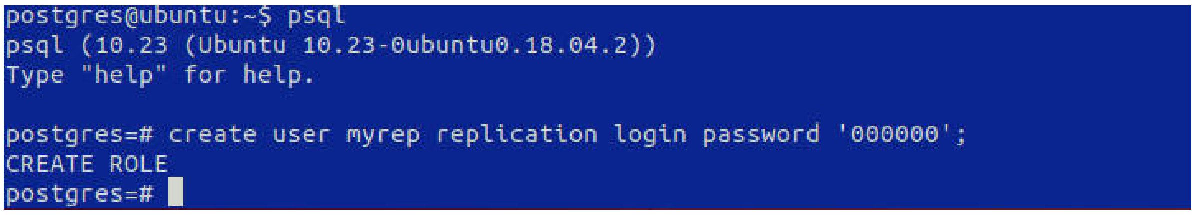
Step 4 Create Replication user in Master

For master: Enter in postgres coding mode.

```
psql
```

For master: Create Replication user with login privilege.

```
create user myrep replication login password '000000';
```



```
postgres@ubuntu:~$ psql
psql (10.23 (Ubuntu 10.23-0ubuntu0.18.04.2))
Type "help" for help.

postgres=# create user myrep replication login password '000000';
CREATE ROLE
postgres=#
```

Step 5 Login the Replication user in Standby

For standby: Login with replication user

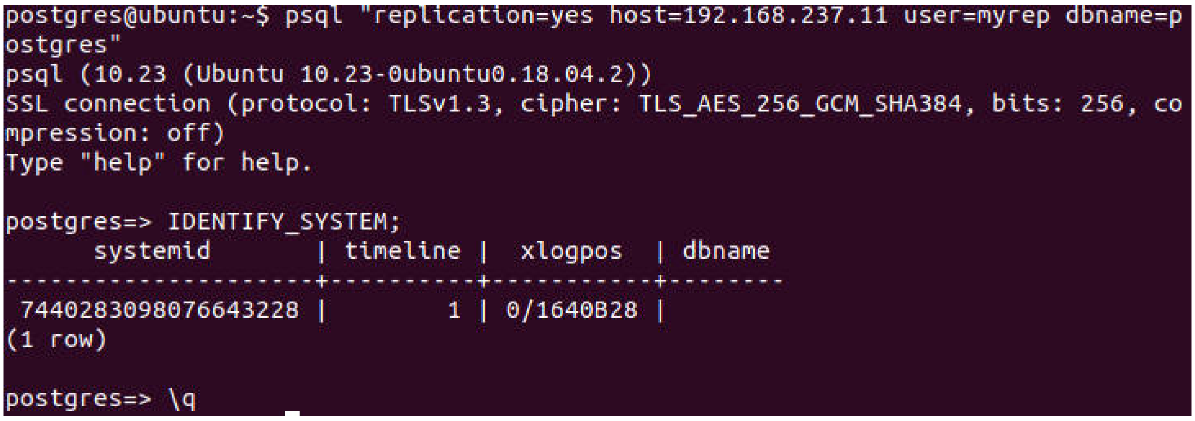
```
psql "replication=yes host=192.168.237.11 user=myrep dbname=postgres"
```

For standby: Test replication mode/pg_basebackup.

```
IDENTIFY_SYSTEM;
```

For standby: Quit postgres coding mode.

```
\q
```



```
postgres@ubuntu:~$ psql "replication=yes host=192.168.237.11 user=myrep dbname=p
ostgres"
psql (10.23 (Ubuntu 10.23-0ubuntu0.18.04.2))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, co
mpression: off)
Type "help" for help.

postgres=> IDENTIFY_SYSTEM;
      systemid       | timeline |  xlogpos  | dbname
---------------------+----------+-----------+--------
 7440283098076643228 |        1 | 0/1640B28 |
(1 row)

postgres=> \q
```

Step 6 Recover replication database in standby

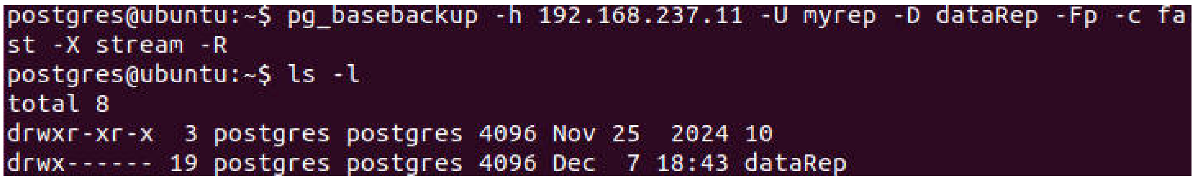
For standby: Use `pg_basebackup` command to create a replication database from master (192.168.237.11)

```
pg_basebackup -h 192.168.237.11 -U myrep -D dataRep -Fp -c fast -X stream -R
```

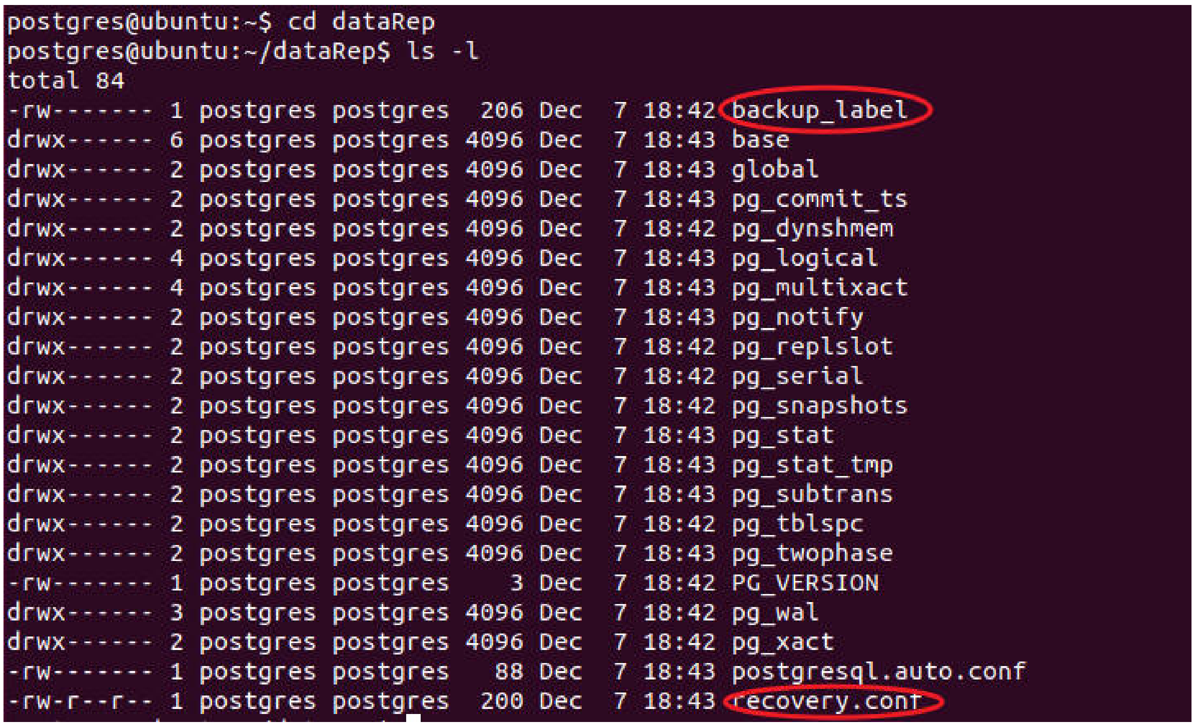
- `-h 192.168.237.11`: replicate from master with IP 192.168.237.11
- `-U myrep`: use user myrep to get connection
- `-D dataRep`: replicate to local direction dataRep
- `-Fp`: use plain form (Common file system format)
- `-c fast`: use fast check point
- `-X stream`: pass WAL log stream during replication
- `-R`: recover automatically (recovery.conf or standby.signal)

For standby: Check the replication database's status.

```
ls -l
```


```
postgres@ubuntu:~$ pg_basebackup -h 192.168.237.11 -U myrep -D dataRep -Fp -c fa
st -X stream -R
postgres@ubuntu:~$ ls -l
total 8
drwxr-xr-x  3 postgres postgres 4096 Nov 25  2024 10
drwx------ 19 postgres postgres 4096 Dec  7 18:43 dataRep
```

```
cd dataRep
ls -l
```


```
postgres@ubuntu:~$ cd dataRep
postgres@ubuntu:~/dataRep$ ls -l
total 84
-rw------- 1 postgres postgres  206 Dec  7 18:42 backup_label
drwx------ 6 postgres postgres 4096 Dec  7 18:43 base
drwx------ 2 postgres postgres 4096 Dec  7 18:43 global
drwx------ 2 postgres postgres 4096 Dec  7 18:43 pg_commit_ts
drwx------ 2 postgres postgres 4096 Dec  7 18:42 pg_dynshmem
drwx------ 4 postgres postgres 4096 Dec  7 18:43 pg_logical
drwx------ 4 postgres postgres 4096 Dec  7 18:43 pg_multixact
drwx------ 2 postgres postgres 4096 Dec  7 18:43 pg_notify
drwx------ 2 postgres postgres 4096 Dec  7 18:42 pg_replslot
drwx------ 2 postgres postgres 4096 Dec  7 18:42 pg_serial
drwx------ 2 postgres postgres 4096 Dec  7 18:42 pg_snapshots
drwx------ 2 postgres postgres 4096 Dec  7 18:43 pg_stat
drwx------ 2 postgres postgres 4096 Dec  7 18:43 pg_stat_tmp
drwx------ 2 postgres postgres 4096 Dec  7 18:43 pg_subtrans
drwx------ 2 postgres postgres 4096 Dec  7 18:42 pg_tblspc
drwx------ 2 postgres postgres 4096 Dec  7 18:43 pg_twophase
-rw------- 1 postgres postgres    3 Dec  7 18:42 PG_VERSION
drwx------ 3 postgres postgres 4096 Dec  7 18:42 pg_wal
drwx------ 2 postgres postgres 4096 Dec  7 18:42 pg_xact
-rw------- 1 postgres postgres   88 Dec  7 18:43 postgresql.auto.conf
-rw-r--r-- 1 postgres postgres  200 Dec  7 18:43 recovery.conf
```

For standby: Check the information of file `recovery.conf`, it will like the follow format, you can replace the detail information by your own.

```
standby_mode = 'on'
primary_conninfo = 'host=主库IP port=5432 user=replicator password=密码
application_name=standby1'
trigger_file = '/tmp/promote_standby'
recovery_target_timeline = 'latest'
restore_command = 'cp /var/lib/postgresql/wal_archive/%f %p'
```

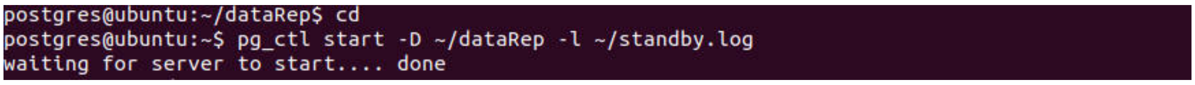
Step 7 Check the data stream

For standby: Start the `standby.log` service.

```
pg_ctl start -D ~/dataRep -l ~/standby.log
```

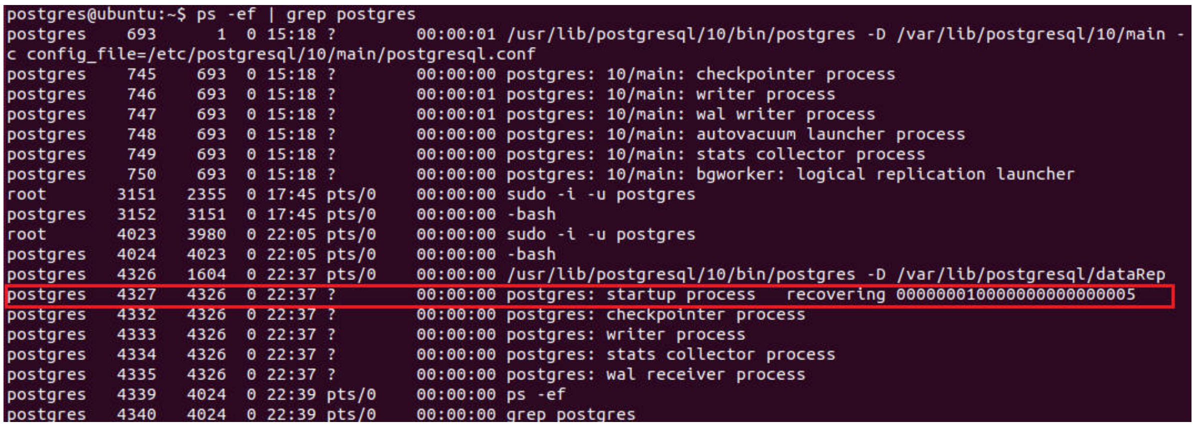
For standby: Check if the `standby.log` works.

```
tail -f ~/standby.log
```

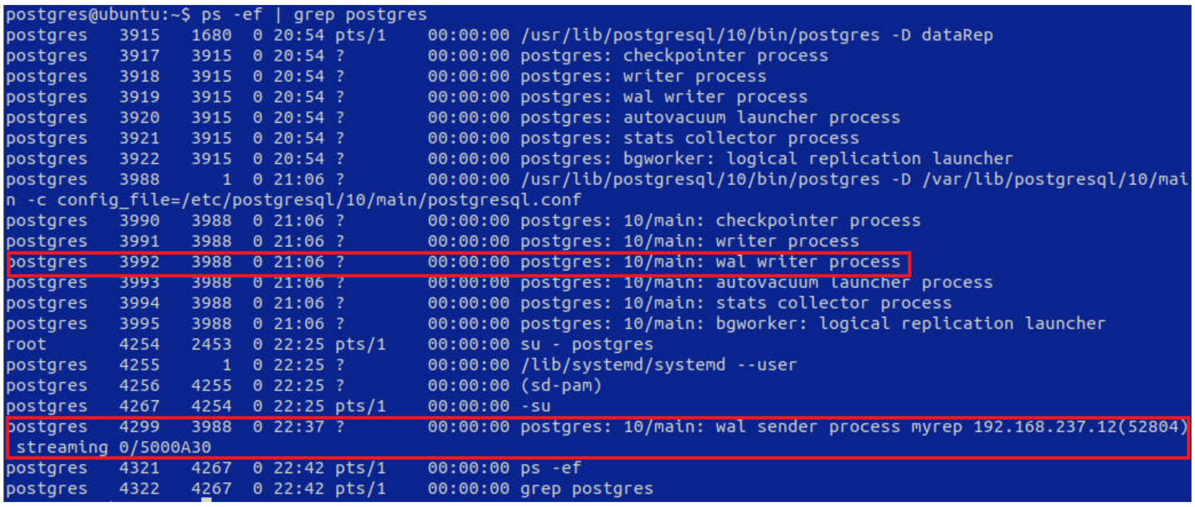

```
postgres@ubuntu:~/dataRep$ cd
postgres@ubuntu:~$ pg_ctl start -D ~/dataRep -l ~/standby.log
waiting for server to start.... done
```

For master and standby: grep the postgres process information.

```
ps -ef | grep postgres
```



```
postgres@ubuntu:~$ ps -ef | grep postgres
postgres    693      1  0 15:18 ?        00:00:01 /usr/lib/postgresql/10/bin/postgres -D /var/lib/postgresql/10/main -
c config_file=/etc/postgresql/10/main/postgresql.conf
postgres    745    693  0 15:18 ?        00:00:00 postgres: 10/main: checkpointer process
postgres    746    693  0 15:18 ?        00:00:01 postgres: 10/main: writer process
postgres    747    693  0 15:18 ?        00:00:01 postgres: 10/main: wal writer process
postgres    748    693  0 15:18 ?        00:00:00 postgres: 10/main: autovacuum launcher process
postgres    749    693  0 15:18 ?        00:00:00 postgres: 10/main: stats collector process
postgres    750    693  0 15:18 ?        00:00:00 postgres: 10/main: bgworker: logical replication launcher
root       3151   2355  0 17:45 pts/0    00:00:00 sudo -i -u postgres
postgres   3152   3151  0 17:45 pts/0    00:00:00 -bash
root       4023   3980  0 22:05 pts/0    00:00:00 sudo -i -u postgres
postgres   4024   4023  0 22:05 pts/0    00:00:00 -bash
postgres   4326   1604  0 22:37 pts/0    00:00:00 /usr/lib/postgresql/10/bin/postgres -D /var/lib/postgresql/dataRep
postgres   4327   4326  0 22:37 ?        00:00:00 postgres: startup process   recovering 000000010000000000000005
postgres   4332   4326  0 22:37 ?        00:00:00 postgres: checkpointer process
postgres   4333   4326  0 22:37 ?        00:00:00 postgres: writer process
postgres   4334   4326  0 22:37 ?        00:00:00 postgres: stats collector process
postgres   4335   4326  0 22:37 ?        00:00:00 postgres: wal receiver process
postgres   4339   4024  0 22:39 pts/0    00:00:00 ps -ef
postgres   4340   4024  0 22:39 pts/0    00:00:00 grep postgres
```



```
postgres@ubuntu:~$ ps -ef | grep postgres
postgres   3915   1680  0 20:54 pts/1    00:00:00 /usr/lib/postgresql/10/bin/postgres -D dataRep
postgres   3917   3915  0 20:54 ?        00:00:00 postgres: checkpointer process
postgres   3918   3915  0 20:54 ?        00:00:00 postgres: writer process
postgres   3919   3915  0 20:54 ?        00:00:00 postgres: wal writer process
postgres   3920   3915  0 20:54 ?        00:00:00 postgres: autovacuum launcher process
postgres   3921   3915  0 20:54 ?        00:00:00 postgres: stats collector process
postgres   3922   3915  0 20:54 ?        00:00:00 postgres: bgworker: logical replication launcher
postgres   3988      1  0 21:06 ?        00:00:00 /usr/lib/postgresql/10/bin/postgres -D /var/lib/postgresql/10/mai
n -c config_file=/etc/postgresql/10/main/postgresql.conf
postgres   3990   3988  0 21:06 ?        00:00:00 postgres: 10/main: checkpointer process
postgres   3991   3988  0 21:06 ?        00:00:00 postgres: 10/main: writer process
postgres   3992   3988  0 21:06 ?        00:00:00 postgres: 10/main: wal writer process
postgres   3993   3988  0 21:06 ?        00:00:00 postgres: 10/main: autovacuum launcher process
postgres   3994   3988  0 21:06 ?        00:00:00 postgres: 10/main: stats collector process
postgres   3995   3988  0 21:06 ?        00:00:00 postgres: 10/main: bgworker: logical replication launcher
root       4254   2453  0 22:25 pts/1    00:00:00 su - postgres
postgres   4255      1  0 22:25 ?        00:00:00 /lib/systemd/systemd --user
postgres   4256   4255  0 22:25 ?        00:00:00 (sd-pam)
postgres   4267   4254  0 22:25 pts/1    00:00:00 -su
postgres   4299   3988  0 22:37 ?        00:00:00 postgres: 10/main: wal sender process myrep 192.168.237.12(52804)
 streaming 0/5000A30
postgres   4321   4267  0 22:42 pts/1    00:00:00 ps -ef
postgres   4322   4267  0 22:42 pts/1    00:00:00 grep postgres
```

Test replication

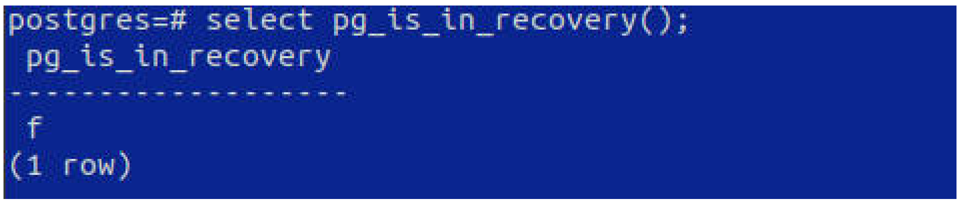
Step1 Check the streaming status

For master and standby: Check the streaming status by querying the value of `pg_is_in_recovery()`. `f` for false, stand for sender; `t` for true, stand for receiver.

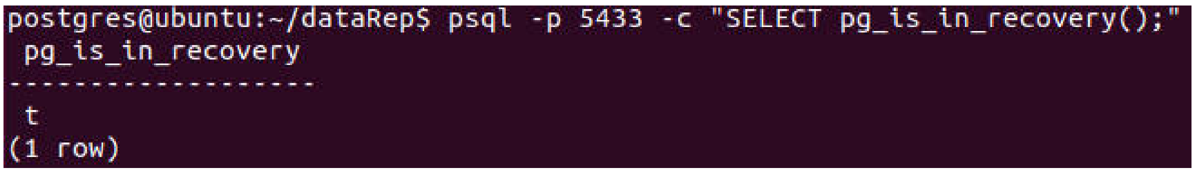
```
psql
select pg_is_in_recovery();
```

or(here the port I used is 5433 which you can replace by your own)

```
psql -p 5433 -c "SELECT pg_is_in_recovery();"
```



```
postgres=# select pg_is_in_recovery();
 pg_is_in_recovery
-------------------
 f
(1 row)
```



```
postgres@ubuntu:~/dataRep$ psql -p 5433 -c "SELECT pg_is_in_recovery();"
 pg_is_in_recovery
-------------------
 t
(1 row)
```

Step 2 An replication example

For master: Check the original databases in master(in psql coding mode)

```
\l
```

For standby: Check the original databases in standby

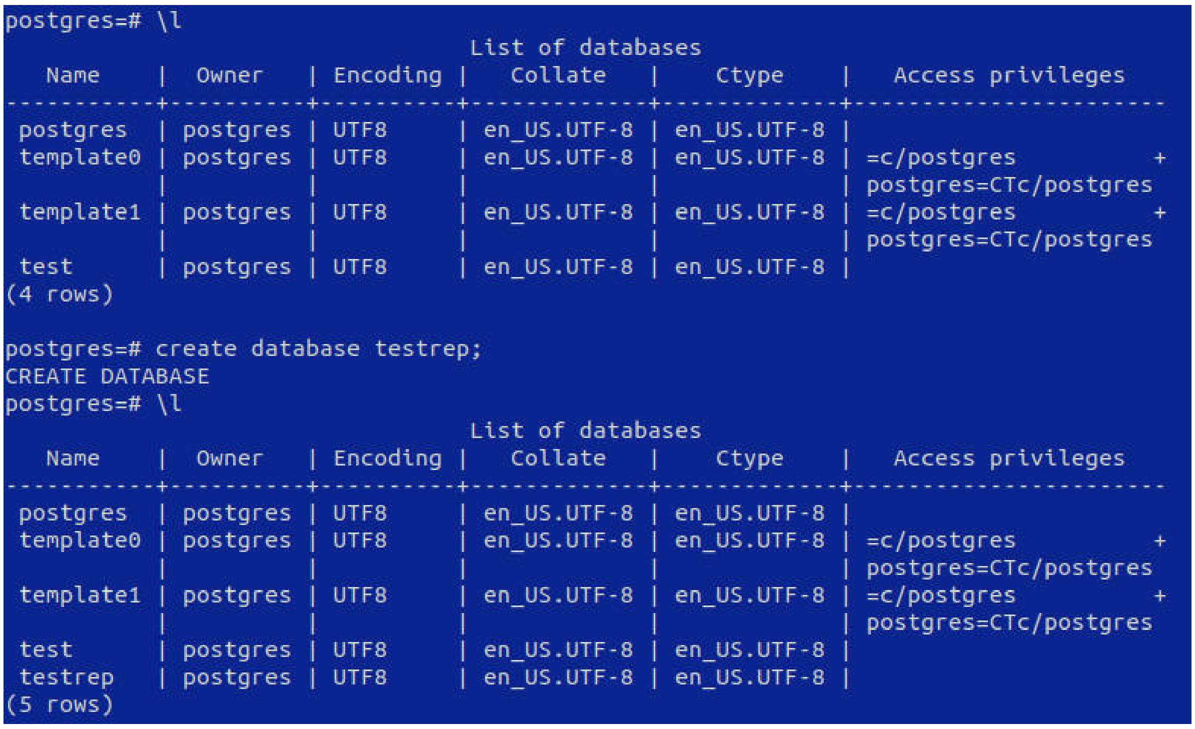
```
psql -p 5433 -c "\l"
```

For master: Create a database `testrep`

```
create database testrep;
```

For master: Check the updated databases in master

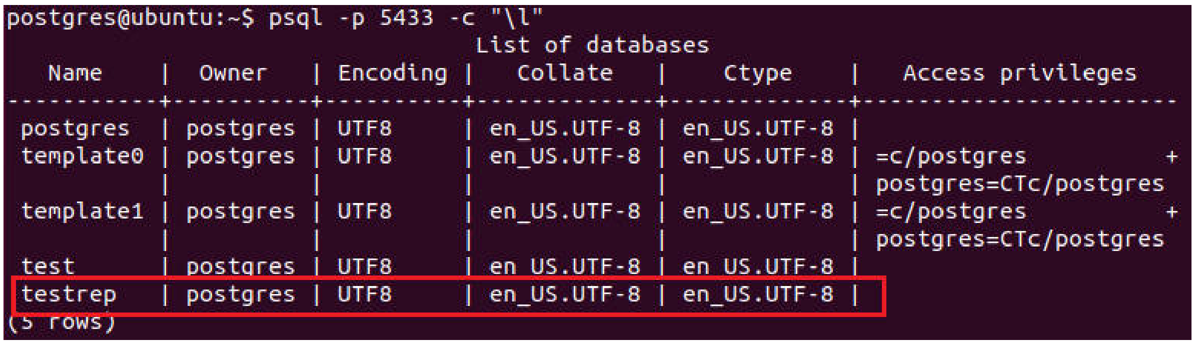
```
\l
```



```
postgres=# \l
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 test      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
(4 rows)

postgres=# create database testrep;
CREATE DATABASE
postgres=# \l
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 test      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 testrep   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
(5 rows)
```

For standby: Check the Updated databases in standby

```
psql -p 5433 -c "\l"
```



```
postgres@ubuntu:~$ psql -p 5433 -c "\l"
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
 postgres  | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 test      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 testrep   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
(5 rows)
```

Now you can make a replication from master to standby!

Reference

1. https://cloud.tencent.com/developer/article/2315296
2. https://ubuntu.com/server/docs/install-and-configure-postgresql
3. https://blog.csdn.net/weixin_42366065/article/details/123883942
4. http://postgres.cn/docs/15/warm-standby.html
5. http://postgres.cn/docs/15/runtime-config-replication.html
6. https://www.postgresql.org/docs/16/warm-standby.html#STREAMING-REPLICATION
7. http://postgres.cn/docs/16/runtime-config-replication.html
8. https://www.cnblogs.com/abclife/p/16403174.html