




Instalación de Servidor PostgreSQL en Debian 12



Autor 

- Andrés Morales González
-  I.E.S. Gonzalo Nazareno  (Dos Hermanas, Sevilla).

Índice

- [Instalación de Servidor PostgreSQL en Debian 12](#)
 - [Autor](#) 
- [Índice](#)
- [Instalacion de servidor Postgres en Debian12](#)
- [Instalación y configuracion de PostgreSQL](#)
 - [1. Instalar PostgreSQL](#)
 - [2. Configuracion para el acceso remoto](#)
 - [Paso a:](#)
 - [Paso b:](#)
 - [Paso c:](#)
- [Creacion de un cliente en postgres](#)

Instalacion de servidor Postgres en Debian12

Para ello lo primero que haremos sera la creaci3n de una maquina debian, sin entorno gr3fico.

Una vez realizada procederemos a la instalaci3n por comandos de dicho servidor:

Instalaci3n y configuracion de PostgreSQL

1. Instalar PostgreSQL

Para ello tendremos que meter los siguientes comandos:

Este comando que meteremos a continuaci3n ser3 para actualizar lo que sera el sistema:

```
sudo apt update
```

Este comando sera para la instalaci3n de nuestro servidor:

```
sudo apt install postgresql postgresql-contrib -y
```

2. Configuracion para el acceso remoto

Paso a:

Modificamos el archivo de configuraci3n postgresql.conf para permitir conexiones desde la red local:

```
sudo nano /etc/postgresql/15/main/postgresql.conf
```

Tenemos que buscar la linea *listen_addresses* y le aadireremos la siguiente linea:

```
listen_addresses = '*'
```

```

GNU nano 7.2 /etc/postgresql/15/main/postgresql.conf *
external_pid_file = '/var/run/postgresql/15-main.pid'                                # write>
                                                    # (change requires restart)

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

# - Connection Settings -

#listen_addresses = 'localhost'               # what IP address(es) to listen on;
listen_addresses = '*'                         # comma-separated list >
                                                    # defaults to 'localhost'; use '*' for >
                                                    # (change requires restart)
port = 5432                                    # (change requires restart)
max_connections = 100                          # (change requires restart)
#superuser_reserved_connections = 3            # (change requires restart)
unix_socket_directories = '/var/run/postgresql' # comma-separated list of direc>
                                                    # (change requires restart)
#unix_socket_group = ''                       # (change requires restart)

^G Ayuda    ^O Guardar  ^W Buscar   ^K Cortar   ^T Ejecutar ^C Ubicación
^X Salir    ^R Leer fich. ^\ Reemplazar ^U Pegar    ^J Justificar ^/ Ir a línea

```

Paso b:

Ahora editamos el control de acceso *pg_hba.conf* para añadir permisos en nuestra red local:

```
sudo nano /etc/postgresql/15/main/pg_hba.conf
```

En este caso en el fichero si indagamos un poco hacia abajo, lo que podemos ver es la siguiente línea:

```

# IPv4 local connections:
host      all             all             127.0.0.1/32             scram-sha-
256

```

y yo en este caso voy a permitir la conexión desde cualquier red, ya que trabajamos tanto en clase, como en casa, así que pondremos lo siguiente:

```

# Permitir conexiones desde cualquier red
host      all             all             0.0.0.0/0                scram-sha-
256

```

Con lo que quedaría así:

```
GNU nano 7.2 /etc/postgresql/15/main/pg_hba.conf *
# Noninteractive access to all databases is required during automatic
# maintenance (custom daily cronjobs, replication, and similar tasks).
#
# Database administrative login by Unix domain socket
local    all             postgres                                peer

# TYPE      DATABASE        USER            ADDRESS              METHOD

# "local" is for Unix domain socket connections only
local    all             all                                peer
# IPv4 local connections:
host     all             all             127.0.0.1/32         scram-sha-256

#Permitir conexion desde cualquier red

host     all             all             0.0.0.0/0            scram-sha-256

# IPv6 local connections:
host     all             all             ::1/128              scram-sha-256
# Allow replication connections from localhost, by a user with the

^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar      ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich.^_ Reemplazar  ^U Pegar       ^J Justificar ^/ Ir a línea
```

Paso c:

Reiniciamos el servicio PostgreSQL:

```
sudo systemctl restart postgresql
```

y una vez reiniciado vemos su estado con el comando:

```
sudo systemctl status postgresql
```

```

Configurando postgresql (15+248) ...
Procesando disparadores para man-db (2.11.2-2) ...
Procesando disparadores para libc-bin (2.36-9+deb12u8) ...
andy@servidores:~$ sudo nano /etc/postgresql/XX/main/postgresql.conf
andy@servidores:~$ sudo nano /etc/postgresql/15/main/

conf.d/          pg_ctl.conf      pg_ident.conf    start.conf
environment      pg_hba.conf      postgresql.conf
andy@servidores:~$ sudo nano /etc/postgresql/15/main/postgresql.conf
andy@servidores:~$ sudo nano /etc/postgresql/15/main/pg_hba.conf
andy@servidores:~$ sudo systemctl restart postgresql
andy@servidores:~$ sudo systemctl sta
start    status
andy@servidores:~$ sudo systemctl status postgresql
• postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: e>
   Active: active (exited) since Tue 2024-10-08 16:04:52 CEST; 9s ago
   Process: 3027 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 3027 (code=exited, status=0/SUCCESS)
      CPU: 1ms

oct 08 16:04:52 servidores systemd[1]: Starting postgresql.service - PostgreSQL>
oct 08 16:04:52 servidores systemd[1]: Finished postgresql.service - PostgreSQL>
lines 1-9/9 (END)

```

Creacion de un cliente en postgres

Se creará un cliente que pueda entrar desde cualquier host:

```

andy@servidores:~$ sudo -u postgres psql
could not change directory to "/home/andy": Permiso denegado
psql (15.8 (Debian 15.8-0+deb12u1))
Type "help" for help.

postgres=# CREATE USER andy WITH PASSWORD 'andy';
CREATE ROLE
postgres=# ALTER USER andy CREATEDB;
ALTER ROLE
postgres=#

```

Pero este cliente si lo intentamos conectar nos dara este error :

```

andy@cliente-mariadb:~$ psql -U andy -h 192.168.1.159
Contraseña para usuario andy:
psql: error: falló la conexión al servidor en «192.168.1.159», puerto 5432:

```

```
FATAL: no existe la base de datos «andy»
```

ya que no hay ninguna bases de datos con ese nombre, ya que solo hemos creado el usuario, para ello nos iremos al servidor y crearemos una bases de datos, la cual llamaremos testeo, de la siguiente manera:

```
andy@servidores:~$ sudo -u postgres psql
could not change directory to "/home/andy": Permiso denegado
psql (15.8 (Debian 15.8-0+deb12u1))
Type "help" for help.

postgres=# \l
postgres=# CREATE DATABASE testeo;
CREATE DATABASE
postgres=#
```

Y ahora probamos a conectarnos:

```
andy@cliente-mariadb:~$ psql -U andy -h 192.168.1.159 -d testeo
Contraseña para usuario andy:
psql (15.8 (Debian 15.8-0+deb12u1))
Conexión SSL (protocolo: TLSv1.3, cifrado: TLS_AES_256_GCM_SHA384,
compresión: desactivado)
Digite «help» para obtener ayuda.

testeo=>
```

y como vemos estamos conectado y con todas la funcionalidades.