How to Install Ubuntu Server 22.04 LTS Step by Step

<https://www.linuxtechi.com/install-ubuntu-server-22-04-step-by-step/>

1. Download the Ubuntu server 22.04 iso file from its [official page](https://ubuntu.com/download/server),
2. On windows 11, use “Rufus” to create a bootable USB drive. <https://rufus.ie/en/>
3. Install the Server
4. Create a test user name winter/winter1

**Install PostgreSql**:

<https://computingforgeeks.com/how-to-install-postgresql-13-on-ubuntu/>

Check if PostgreSQL installed: $ postgres –version or $pg\_lsclusters command not found

## **Step 1: Update Ubuntu system**

We always work on a latest release of OS to make sure there are no old dependency issues. Login to your Ubuntu server and run the following commands to update all the packages installed.

sudo apt update && sudo apt -y full-upgrade

[ -f /var/run/reboot-required ] && sudo reboot -f

Once the system has been updated, I recommend you perform a reboot to get the new kernel running incase it was updated.

sudo reboot

## **Step 2: Add PostgreSQL 13 repository to Ubuntu 22.04|20.04|18.04**

Install required dependency packages

sudo apt update

sudo apt install curl gpg gnupg2 software-properties-common apt-transport-https lsb-release ca-certificates

Now that we have updated and rebooted our system, let’s add the APT repository required to pull the packages form the PostgreSQL repository.

curl -fsSL https://www.postgresql.org/media/keys/ACCC4CF8.asc|sudo gpg –dearmor -o /etc/apt/trusted.gpg.d/postgresql.gpg

After importing GPG key, add repository contents to your Ubuntu 22.04|20.04|18.04 system:

echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb\_release -cs`-pgdg main" |sudo tee /etc/apt/sources.list.d/pgdg.list

The repository added contains many different packages including third party addons. They include:

* postgresql-client
* postgresql
* libpq-dev
* postgresql-server-dev
* pgadmin packages

this one worked

wget -qO - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo gpg --dearmor -o /usr/share/keyrings/postgresql-keyring.gpg

echo "deb [arch=amd64 signed-by=/usr/share/keyrings/postgresql-keyring.gpg] http://apt.postgresql.org/pub/repos/apt $(lsb\_release -cs)-pgdg main" | sudo tee /etc/apt/sources.list.d/postgresql.list

## **Step 3: Install PostgreSQL 13 on Ubuntu 22.04|20.04|18.04**

With the repository added we can install the PostgreSQL 13 packages on our Ubuntu 22.04|20.04|18.04 Linux server. But first update the package index for the version to be available at the OS level.

sudo apt update

The run the commands below to install PostgreSQL 13 on Ubuntu 22.04|20.04|18.04 Linux system.

sudo apt install postgresql-13 postgresql-client-13

The PostgreSQL service is started and set to come up after every system reboot.

**sudo apt-get install libpq5=14.5-0ubuntu0.22.04.1**

**sudo apt-get install libpq-dev=14.5-0ubuntu0.22.04.1**

**pip3 install psycopg2**

sudo apt-get install libpq5=14.2-1ubuntu1

$ systemctl status postgresql@13-main.service

## **Step 4: Test PostgreSQL Connection**

During installation, a postgres user is created automatically. This user has full **superadmin** access to your entire PostgreSQL instance. Before you switch to this account, your logged in system user should have sudo privileges.

sudo su - postgres

Let’s reset this user password to a strong Password we can remember.

psql -c "alter user postgres with password 'StrongAdminP@ssw0rd'"

Start PostgreSQL prompt by using the command:

$ psql

Get connection details like below.

$ psql

psql (13.9 (Ubuntu 13.9-1.pgdg22.04+1))

Type "help" for help.

postgres=# \conninfo

You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql" at port "5432".

Let’s create a test database and user to see if it’s working.

postgres=# CREATE DATABASE winn\_dev\_6; /\* for 1\_django\_pg\_dev\_multi \*/

CREATE DATABASE

postgres=# CREATE DATABASE powerworks\_d1; /\* for D:\dev\1b\_gui\_app\pyqt\_chart \*/

postgres=# CREATE USER winter WITH PASSWORD 'winter';

CREATE ROLE

postgres=#GRANT ALL PRIVILEGES ON DATABASE winn\_dev\_6 to winter;

GRANT

postgres=#GRANT ALL PRIVILEGES ON DATABASE powerworks\_d1 to winter;

List created databases:

postgres=# \l

List of databases

Name | Owner | Encoding | Collate | Ctype | Access privileges

-----------+----------+----------+---------+---------+-------------------------

mytestdb | postgres | UTF8 | C.UTF-8 | C.UTF-8 | =Tc/postgres +

| | | | | postgres=CTc/postgres +

| | | | | mytestuser=CTc/postgres

postgres | postgres | UTF8 | C.UTF-8 | C.UTF-8 |

template0 | postgres | UTF8 | C.UTF-8 | C.UTF-8 | =c/postgres +

| | | | | postgres=CTc/postgres

template1 | postgres | UTF8 | C.UTF-8 | C.UTF-8 | =c/postgres +

| | | | | postgres=CTc/postgres

(4 rows)

Connect to database:

postgres-# \c mytestdb

You are now connected to database "mytestdb" as user "postgres".

Other PostgreSQL utilities installed such as **createuser** and **createdb** can be used to create database and users.

postgres@ubuntu:~$ createuser myuser --password

Password:

postgres@ubuntu:~$ createdb mydb -O myuser

postgres@ubuntu:~$psql -l

## **Step 5: Configure remote Connection (Optional)**

Installation of PostgreSQL 13 on Ubuntu only accepts connections from localhost. In ideal production environments, you’ll have a central database server and remote clients connecting to it – But of course within a **private network** (LAN).

To enable remote connections, edit PostgreSQL configuration file:

sudo nano /etc/postgresql/13/main/postgresql.conf

Uncomment line **59** and change the Listen address to accept connections within your networks.

# Listen on all interfaces

listen\_addresses = '\*'

# Listen on specified private IP address

listen\_addresses = '192.168.10.11'

Also set PostgreSQL to accept remote connections from allowed hosts.

$ sudo nano /etc/postgresql/13/main/pg\_hba.conf

# Accept from anywhere

host all all 0.0.0.0/0 md5

# Accept from trusted subnet

host all all 10.10.10.0/24 md5

After the change, restart postgresql service.

sudo systemctl restart postgresql

Confirm Listening addresses.

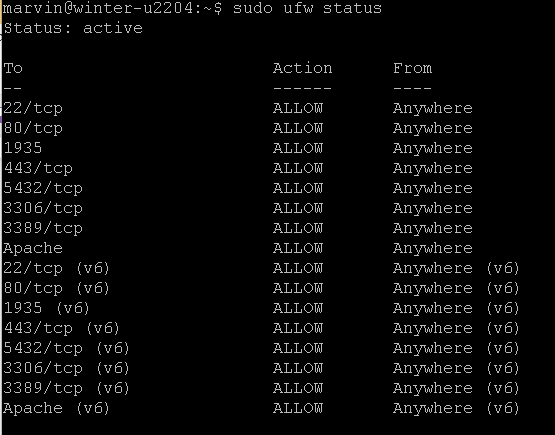
# netstat -tunelp | grep 5432

tcp 0 0 0.0.0.0:5432 0.0.0.0:\* LISTEN 111 112837 11143/postgres

tcp6 0 0 :::5432 :::\* LISTE

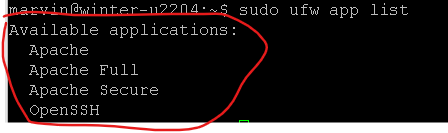
**Set up server firewall – ufw**

* sudo ufw status
* sudo ufw allow 22/tcp

****

**Setting up Apache2**

1. Check **apache2** version, install it if not installed already

* $apache2 -v
* Sudo apt update
* Sudo apt install apache2
* Sudo ufw app list
* 
  + Apache: This profile opens only port 80 (normal, unencrypted web traffic)
  + Apache Full: This profile opens both port 80 (normal, unencrypted web traffic) and port 443 (TLS/SSL encrypted traffic)
  + Apache Secure: This profile opens only port 443 (TLS/SSL encrypted traffic)
* Sudo ufw allow ‘Apache’

**Create user:**

1. Switch to root user: $sudo su –
2. $adduser winter