

How to Install PostgreSQL On Ubuntu 22.04 Step-by-Step

Last Updated: March 20, 2024 by [Narendra K](#)

In this post, we will show you how to install PostgreSQL on Ubuntu 22.04 step-by-step.

PostgreSQL is a powerful, open-source object-relational Database Management System (DBMS). It's been battle-tested for over 35 years which has earned it a strong reputation for reliability and performance. This feature-rich database is used by many tech giants, such as Apple, IMDB, Instagram, and so on.

[PostgreSQL](#) supports large number of the SQL standard and is constructed to be extensible by users in many aspects. Some of the salient features include ACID transactions, foreign keys, subqueries, triggers, user-defined types, functions, etc.

Table of Contents

Prerequisites

- 1) Add PostgreSQL Package Repository
- 2) Install PostgreSQL on Ubuntu 22.04
- 3) Update PostgreSQL Admin User Password
- 4) Configure PostgreSQL to Allow Remote Connections

Verifying Remote Connection

Prerequisites

Before installing the PostgreSQL database server, we must ensure that the system meets the following installation requirements:

- Pre-Installed Ubuntu 22.04
- A regular user with sudo rights
- An active internet connection
- At least 2 GB of RAM with an additional 512 MB of disk space. Please note that [^] is a minimal requirement for the demo environment. The actual hardware

configuration will vary with data volume.

Without any further delay, let's deep dive into PostgreSQL installation steps.

AD

...

1) Add PostgreSQL Package Repository

PostgreSQL 15 package is not available in the default package repository, so add its official package repository using following commands.

```
$ sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release  
$ wget -qO- https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo tee /et
```

To begin, let's fetch the latest versions of the packages. We can achieve this using the apt update command as shown below:

```
$ sudo apt update
```

The above command will take a few seconds to complete.

^

AD

● ● ●

2) Install PostgreSQL on Ubuntu 22.04

The postgresql package installs the default version of the PostgreSQL database server whereas the PostgreSQL-client package installs the client utility.

Let's install the PostgreSQL client and server using the below apt command:

```
$ sudo apt install postgresql postgresql-client -y
```

```
linuxtechi@ubuntu-jammy:~$ sudo apt install postgresql postgresql-client -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcommon-sense-perl libjson-perl libjson-xs-perl libpq5 libtypes-serialiser-perl postgresql-16 postgresql-client-16 postgresql-client-common
  postgresql-common sysstat
Suggested packages:
  postgresql-doc postgresql-doc-16 isag
The following NEW packages will be installed:
  libcommon-sense-perl libjson-perl libjson-xs-perl libpq5 libtypes-serialiser-perl postgresql postgresql-16 postgresql-client postgresql-client-16
  postgresql-client-common postgresql-common sysstat
0 upgraded, 12 newly installed, 0 to remove and 16 not upgraded.
Need to get 21.3 MB of archives.
After this operation, 72.6 MB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libjson-perl all 4.04000-1 [81.8 kB]
Get:2 http://apt.postgresql.org/pub/repos/apt jammy-pgdg/main amd64 postgresql-client-common all 256.pgdg22.04+1 [94.0 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libcommon-sense-perl amd64 3.75-2build1 [21.1 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libtypes-serialiser-perl all 1.01-1 [11.6 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libjson-xs-perl amd64 4.030-1build3 [87.2 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 sysstat amd64 12.5.2-2ubuntu0.2 [487 kB]
Get:7 http://apt.postgresql.org/pub/repos/apt jammy-pgdg/main amd64 postgresql-common all 256.pgdg22.04+1 [238 kB]
Get:8 http://apt.postgresql.org/pub/repos/apt jammy-pgdg/main amd64 libpq5 amd64 16.1-1.pgdg22.04+1 [213 kB]
Get:9 http://apt.postgresql.org/pub/repos/apt jammy-pgdg/main amd64 postgresql-client-16 amd64 16.1-1.pgdg22.04+1 [1,889 kB]
Get:10 http://apt.postgresql.org/pub/repos/apt jammy-pgdg/main amd64 postgresql-16 amd64 16.1-1.pgdg22.04+1 [18.1 MB]
Get:11 http://apt.postgresql.org/pub/repos/apt jammy-pgdg/main amd64 postgresql all 16+256.pgdg22.04+1 [68.9 kB]
Get:12 http://apt.postgresql.org/pub/repos/apt jammy-pgdg/main amd64 postgresql-client all 16+256.pgdg22.04+1 [68.9 kB]
Fetched 21.3 MB in 4min 30s (78.9 kB/s)
```

^

AD

...

Next, let's verify that the PostgreSQL service is up and running:

```
$ sudo systemctl status postgresql
```

```
linuxtechi@jammy-jellyfish:~$ sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor preset: enabled)
   Active: active (exited) since Mon 2022-10-24 06:50:11 BST; 49s ago
     Main PID: 37398 (code=exited, status=0/SUCCESS)
        CPU: 2ms

Oct 24 06:50:11 jammy-jellyfish systemd[1]: Starting PostgreSQL RDBMS ...
Oct 24 06:50:11 jammy-jellyfish systemd[1]: Finished PostgreSQL RDBMS.
linuxtechi@jammy-jellyfish:~$
```

Finally, check the PostgreSQL version using the psql command line utility:

^

...

```
$ psql --version
```

Here, we can see that the version of PostgreSQL is 16.

```
linuxtechi@jammy-jellyfish:~$  
linuxtechi@jammy-jellyfish:~$ psql --version  
psql (PostgreSQL) 16.1 (Ubuntu 16.1-1.pgdg22.04+1)  
linuxtechi@jammy-jellyfish:~$  
linuxtechi@jammy-jellyfish:~$
```

3) Update PostgreSQL Admin User Password

By default, we can connect to the PostgreSQL server without using any password. Let's see this in action using the psql utility:

```
$ sudo -u postgres psql  
postgres=#
```

In the above output, the `postgres=#` prompt indicated the active connection with the PostgreSQL server.

In this example, we have used the `postgres` user. This is an admin user of PostgreSQL and it gets created during the installation process.

Allowing administrative access to the database without any password isn't a good idea. So, let's set the password for the postgres user:

AD





```
postgres=# ALTER USER postgres PASSWORD 'demoPassword';
```

The above SQL query sets the user password to [demoPassword](#). Please note that, we have used a very simple password because this is a demo environment. However, the same is not recommended in the production environment.

Let's verify that the password has been set successfully. So first, terminate the current session with the server using the [\q command](#).

```
postgres=# \q
```

Output of above commands,

AD



```
linuxtechi@jammy-jellyfish:~$ sudo -u postgres psql
could not change directory to "/home/linuxtechi": Permission denied
psql (15.0 (Ubuntu 15.0-1.pgdg22.04+1))
Type "help" for help.

postgres=# ALTER USER postgres PASSWORD 'demoPassword';
ALTER ROLE
postgres=# \q
linuxtechi@jammy-jellyfish:~$
linuxtechi@jammy-jellyfish:~$
```

Now, let's connect to the database server again:

```
$ psql -h localhost -U postgres
```

Let's enter the demoPassword string as a password and now we are connected to the database.

...

...

```
linuxtechi@jammy-jellyfish:~$ psql -h localhost -U postgres
Password for user postgres:
psql (15.0 (Ubuntu 15.0-1.pgdg22.04+1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

postgres=# \q
linuxtechi@jammy-jellyfish:~$
```

4) Configure PostgreSQL to Allow Remote Connections

^

By default, PostgreSQL accepts connections from the localhost only. However, we can easily modify the configuration to allow connection from remote clients.

PostgreSQL reads its configuration from the [postgresql.conf](#) file which is located in the [/etc/postgresql/<version>/main/](#) directory. Here, the version indicates the major version of PostgreSQL.

For example, in our case the full path of the file is [/etc/postgresql/16/main/postgresql.conf](#)

Now, open the postgresql.conf file in a text editor, uncomment the line that starts with the [listen_addresses](#), and replace 'localhost' with '*'.

This setting is located under the [CONNECTIONS AND AUTHENTICATION](#) section. After modification the file will look like this:

AD

• • •




```
#-----  
# CONNECTIONS AND AUTHENTICATION  
#-----  
  
# - Connection Settings -  
  
listen_addresses = '*'          # what IP address(es) to listen on;  
                                # comma-separated list of addresses;  
                                # defaults to 'localhost'; use '*' for all  
                                # (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 directories  
                                # (change requires restart)  
#unix_socket_group = ''         # (change requires restart)  
#unix_socket_permissions = 0777 # begin with 0 to use octal notation  
                                # (change requires restart)  
#bonjour = off                  # advertise server via Bonjour  
                                # (change requires restart)
```

Save and close the file.

Next, edit the IPv4 local connections section of the [pg_hba.conf](#) file to allow IPv4 connections from all clients. Please note that this file is also located in [/etc/postgresql/16/main/](#) directory.

AD

• • •

```
$ sudo vi /etc/postgresql/16/main/pg_hba.conf
```

After modification the file will look like this:

^

```
# DO NOT DISABLE!  
# If you change this first entry you will need to make sure that the  
# database superuser can access the database using some other method.  
# 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             192.168.1.0/24         scram-sha-256  
# IPv6 local connections:  
host     all             all             ::1/128                scram-sha-256  
# Allow replication connections from localhost, by a user with the  
# replication privilege.  
local    replication    all             peer  
host     replication    all             127.0.0.1/32           scram-sha-256  
host     replication    all             ::1/128                scram-sha-256
```

In the above configuration indicates to allow connection from the network 192.168.1.0/24

AD

...

In case, Ubuntu firewall is running on your system then allow PostgreSQL 5432 port using following command,

```
$ sudo ufw allow 5432/tcp
```

Verifying Remote Connection

^

Finally, restart the service and verify it's up and running:

```
$ sudo systemctl restart postgresql  
$ sudo systemctl status postgresql
```

Now, let's try to access DB from remote client.

```
$ psql -h 192.168.1.192 -U postgres
```

In this example, 192.168.1.192 is the IP address of the PostgreSQL database server.

```
linuxtechi@jammy-jellyfish:~$ psql -h 192.168.1.192 -U postgres  
Password for user postgres:  
psql (15.0 (Ubuntu 15.0-1.pgdg22.04+1))  
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)  
Type "help" for help.  
  
postgres=#  
postgres=#  
postgres=#  
postgres=#
```

Here we can see that we are able to access DB from the remote client.

That's all from this article. Please do post your queries and feedback in the below comments section.

Read Also: [How to Set Static IP Address on Ubuntu Server 22.04](#)

2 thoughts on “How to Install PostgreSQL On Ubuntu 22.04 Step-by-Step”



Ced Hood

September 7, 2023 at 11:47 pm

Great article! Well written and laid out.

[Reply](#)



**Vitaliy**

September 8, 2023 at 11:37 am

Awesome article! It would be even better if you add steps to install pgadmin + adding connection to the deployed database.

[Reply](#)

Leave a Comment

Join the newsletter!

Subscribe

Recent Posts

[How to Install Docker on Ubuntu 24.04 Step-by-Step](#)

[How to Install Ubuntu Server 24.04 Step-by-Step](#)

[How to Configure Static IP Address on Ubuntu 24.04 \(Desktop\)](#)

[Top 11 Things to Do After Installing Ubuntu 24.04](#)

[How to Install Ubuntu 24.04 LTS Desktop Step-by-Step](#)

[How to Upgrade Ubuntu 22.04 to Ubuntu 24.04 LTS](#)

[How to Upgrade to Fedora 40 From Fedora 39 \(GUI & CLI\)](#)

[How to Install Fedora 40 Workstation Step by Step](#)

[How to Use Free Red Hat Developer Subscription on RHEL 9/8](#)

[How to Install Minikube on Linux Mint 21 Step-by-Step](#)



AD



















[HTML Sitemap](#) [Privacy Policy](#) [Contact Us](#) [About Us](#) [Write For LinuxTechi](#)

© 2024 LinuxTechi



