

How to Install PostgreSQL on FreeBSD 14.0

Learn how to install PostgreSQL on FreeBSD 14.0 with step-by-step instructions. This guide covers installation, configuration, and basic setup for beginners.

Contents

01	Introduction	3
02	Prerequisites	3
03	Install PostgreSQL	3
04	Manage the PostgreSQL System Service	4
05	Secure the PostgreSQL Database Server	6
06	Access the PostgreSQL Database Server	7
07	Conclusion	9

Introduction

PostgreSQL is an open-source advanced Relational Database Management System (RDBMS) designed to handle a wide range of data management tasks. It supports the Structured Query Language (SQL) to manage data in small and large enterprise applications such as analytical systems, Geographic Information Systems (GIS), healthcare applications, and dynamic web applications.

This article explains how to install PostgreSQL on a FreeBSD 14.0 server. You will enable the PostgreSQL database server and secure it for production use on your server.

Prerequisites

Before you begin:

- Deploy a [FreeBSD 14.0 server](#) instance on Vultr.
- Access the server [using SSH](#).
- Create a non-root user with sudo privileges and switch to the user.

Install PostgreSQL

PostgreSQL is available with multiple versions in the default `pkg` repository on FreeBSD. Follow the steps below to install the latest PostgreSQL packages on your server.

1. Update your server packages index.

```
CONSOLE
```

```
$ sudo pkg update
```

2. List all available PostgreSQL packages.

CONSOLE

```
$ pkg search postgresql
```

Output:

```
...
postgresql15-server-15.6      PostgreSQL is the most advanced open-source
database available anywhere
postgresql15-tds_fdw-2.0.3    PostgreSQL foreign data wrapper to connect to TDS
databases
postgresql15-zhparser-2.2     PostgreSQL extension for full-text search of
Chinese
postgresql16-client-16.2      PostgreSQL database (client)
postgresql16-contrib-16.2     The contrib utilities from the PostgreSQL
distribution
postgresql16-docs-16.2        The PostgreSQL documentation set
postgresql16-plperl-16.2      Write SQL functions for PostgreSQL using Perl5
postgresql16-plpython-16.2    Module for using Python to write SQL functions
postgresql16-pltcl-16.2       Module for using Tcl to write SQL functions
postgresql16-server-16.2      PostgreSQL is the most advanced open-source
database available anywhere
```

PostgreSQL 16 is the latest version available in the `pkg` repositories. Install the client and server packages to enable PostgreSQL on your server.

3. Install the PostgreSQL client and server packages on your server.

CONSOLE

```
$ sudo pkg install -y postgresql16-server postgresql16-client
```

Manage the PostgreSQL System Service

PostgreSQL uses the `postgresql` system service profile to run on your FreeBSD server. Follow the steps below to enable the PostgreSQL database server to

start automatically at system boot and verify the service status to manage the PostgreSQL processes on your server.

1. Enable the PostgreSQL service to automatically start at system boot.

CONSOLE

```
$ sudo sysrc postgresql_enable=yes
```

Output:

```
postgresql_enable: -> yes
```

2. Initialize the PostgreSQL database to enable the `postgres` user on your server.

CONSOLE

```
$ sudo service postgresql initdb
```

3. Start the PostgreSQL database server.

CONSOLE

```
$ sudo service postgresql start
```

4. View the PostgreSQL service status and verify that it's running.

CONSOLE

```
$ sudo service postgresql status
```

Output:

```
pg_ctl: server is running (PID: 2098)  
/usr/local/bin/postgres "-D" "/var/db/postgres/data16"
```

Secure the PostgreSQL Database Server

PostgreSQL runs with the default `postgres` privileged database user on your server by default when the initialization process is complete. Follow the steps below to enable password authentication and secure the PostgreSQL database server to grant permitted users access to specific databases.

1. Switch to the `postgres` user account and log in to the PostgreSQL database server console.

```
CONSOLE
```

```
$ sudo -u postgres psql
```

2. Modify the default `postgres` user to use a new encrypted password. Replace `strong_password` with your desired password.

```
SQL
```

```
postgres=# ALTER USER postgres WITH ENCRYPTED PASSWORD  
'strong_password';
```

3. Create a new sample database user `db_manager` with a strong encrypted password.

```
SQL
```

```
postgres=# CREATE USER db_manager ENCRYPTED PASSWORD  
'strong_password';
```

4. Exit the PostgreSQL console.

```
SQL
```

```
postgres=# \q
```

5. Run the following command to change the default `trust` value to `scram-sha-256` in the main PostgreSQL configuration file `pg_hba.conf` to enable password authentication on the PostgreSQL database server.

CONSOLE

```
$ sudo sed -i '' -E '/^(local|host)/s/trust/scram-sha-256/' /var/db/postgres/data16/pg_hba.conf
```

6. Restart the PostgreSQL database server to apply your configuration changes.

CONSOLE

```
$ sudo service postgresql restart
```

Access the PostgreSQL Database Server

You can access the PostgreSQL database console using the `psql` utility that's pre-installed with the PostgreSQL client package on your server. In addition, you can access the PostgreSQL database server using compatible tools that create a direct connection to the database console. In the following steps, use the `psql` utility to access your PostgreSQL database server console and create a new sample database to use on your server.

1. Create a new database `example-vultr` and grant ownership privileges to the `db_manager` user.

CONSOLE

```
$ sudo -u postgres createdb example-vultr -0 db_manager
```

When prompted, enter the `postgres` user password you created earlier.

2. Log in to the PostgreSQL database server using the sample user `db_manager` to test access to the `example-vultr` database.

CONSOLE

```
$ sudo -u postgres psql -U db_manager -d example-vultr
```

Enter the database user password when prompted and press Enter to access the database.

3. Create a new sample `doctors` table.

SQL

```
psql > CREATE TABLE doctors (  
    doctor_id SERIAL PRIMARY KEY,  
    first_name VARCHAR(50),  
    last_name VARCHAR(50),  
    appointment_date DATE  
);
```

The above SQL statement creates a new table in the `example-vultr` database with the following columns:

- `doctor_id` is a `PRIMARY KEY` that uniquely identifies each doctor in the `doctors` table.
- `first_name` and `last_name` store names in the `doctors` table.
- `appointment_date` stores a doctor's appointment date.
- `SERIAL` generates a new `doctor_id` for each record.

4. Insert sample data into the `doctors` table.

SQL

```
psql > INSERT INTO doctors  
    ( first_name, last_name, appointment_date)  
VALUES  
    ( 'Ben', 'Joe', '2024-11-15'),  
    ( 'Carson', 'Smith', '2024-02-28'),  
    ( 'Donald', 'James', '2024-04-10');
```

5. Query the `doctors` table to view all available records.

```
SQL
```

```
psql > SELECT * FROM doctors;
```

Output:

```
SQL
```

```
doctor_id | first_name | last_name | appointment_date
-----+-----+-----+-----
          1 | Ben       | Joe      | 2024-11-15
          2 | Carson   | Smith   | 2024-02-28
          3 | Donald   | James   | 2024-04-10
(3 rows)
```

6. Exit the PostgreSQL database console.

```
SQL
```

```
psql > \q
```

Conclusion

You have installed PostgreSQL on your FreeBSD 14.0 server and accessed the database server console using the `psql` utility to create sample databases and table records. You can further integrate the PostgreSQL database server with your existing applications to securely enable the creation and management of database records on your server. For more information and configuration options, please visit the official [PostgreSQL](#) documentation.



VULTR

