

# Basic Tools

## Server Installation and Management



### Copyright

© Postgres Professional, 2017–2025

Authors: Egor Rogov, Pavel Luzanov, Ilya Bashtanov, Alexey Beresnev

Translated by: Liudmila Mantrova, Alexander Meleshko, Elena Sharafutdinova

Photo: Oleg Bartunov (Phu Monastery and Bhrikuti Peak, Nepal)

### Use of Course Materials

Non-commercial use of course materials (presentations, demonstrations) is allowed without restrictions. Commercial use is possible only with the written permission of Postgres Professional. It is prohibited to make changes to the course materials.

### Feedback

Please send your feedback, comments and suggestions to:

[edu@postgrespro.ru](mailto:edu@postgrespro.ru)

### Disclaimer

Postgres Professional assumes no responsibility for any damages and losses, including loss of income, caused by direct or indirect, intentional or accidental use of course materials. Postgres Professional company specifically disclaims any warranties on course materials. Course materials are provided “as is”, and Postgres Professional company has no obligations to provide maintenance, support, updates, enhancements, or modifications.

Basic Concepts

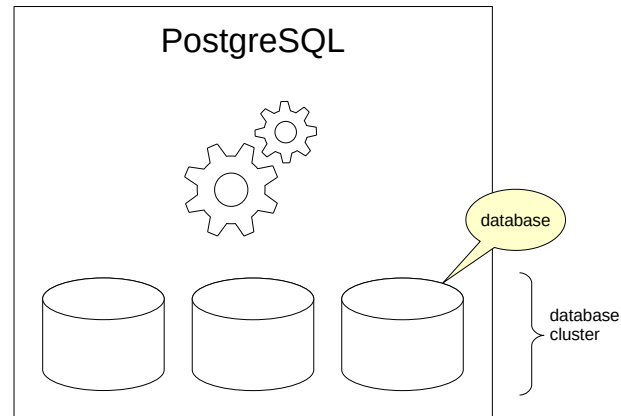
Installation from Source Code

Server Management

Package Installation

Server Management in Ubuntu

Cloud Solutions



Before talking about installation, let's look at some basic concepts.

PostgreSQL is a program that belongs to the class of *database management systems*.

When this program is running, we call it a PostgreSQL *server* or a *server instance*. So far, the server seems to be a "black box" for us, but gradually we will get acquainted with how it works.

The data managed by PostgreSQL is stored in *databases*. One instance of PostgreSQL simultaneously manages several databases. This set of databases is called a *database cluster*. We will talk more about databases in the lesson Data Organization. Databases and Schemas.

So, a database cluster is data in files. A server or a server instance is a program that manages a database cluster.

## Installation from source code

- stable server version
- you can use non-standard parameters
- or build the server for a non-standard architecture

## Installation from git

- current server version
- used primarily by kernel developers; requires a broader set of tools

You should at least have an idea of how to install the program from source code, so that you could build PostgreSQL with non-standard parameters or on a non-standard architecture, if necessary.

You can find the source code in 2 formats: gzip and bzip2 at <https://www.postgresql.org/ftp/source>, as well as in the VM in the student user's home directory.

You can also get the source code straight from the project's git repository: <https://git.postgresql.org/> (there are other mirrors, including github).

This way you can build not just the stable version, but also a version at any specific commit, including the most recent one.

When installing from the git repository, a wider set of tools is required. For example, the lexer and parser are made using Flex and Bison, and git stores the source codes for these tools. They generate C files when built, which are then compiled. The source code archives include the pre-built C files already.

# Required Software



## Hard requirements

tar, gzip/bzip2, GNU make, C compiler (C89)

## Used but can be disabled

GNU Readline, zlib, lz4, and zstd libraries

## Extras

Perl, Python, and Tcl programming languages  
for using PL/Perl, PL/Python, PL/Tcl

Kerberos, OpenSSL, OpenLDAP, PAM for authentication and encryption

ICU library for cross-platform UNICODE support

Separate tools when building from the git repository

5

The configuration and installation process is traditionally based on GNU Make and Autoconf:

<https://postgrespro.com/docs/postgresql/16/install-make>

This course covers the traditional build method.

Starting with version 16, PostgreSQL also supports the Meson build system as an alternative.

<https://postgrespro.com/docs/postgresql/16/install-meson>

A number of programs and utilities are required when building from source.

The readline library allows you to edit the command line, use the command history and auto-completion. In a server installation, it may not be necessary if you are never going to run psql via the console.

The libraries zlib, lz4, and zstd are used for compression.

## Unpacking the Archive

The source code package is located in the student user's home directory. Let's unpack and open it:

```
student$ cd

student$ tar xzf /home/student/postgresql-16.0.tar.gz

student$ ls -l /home/student/postgresql-16.0

total 876
-rw-r--r--  1 student student   365 Sep 11  2023 aclocal.m4
drwxrwxr-x  2 student student  4096 Sep 11  2023 config
-rwxr-xr-x  1 student student 584200 Sep 11  2023 configure
-rw-r--r--  1 student student 87156 Sep 11  2023 configure.ac
drwxrwxr-x 61 student student  4096 Sep 11  2023 contrib
-rw-r--r--  1 student student  1192 Sep 11  2023 COPYRIGHT
drwxrwxr-x  3 student student  4096 Sep 11  2023 doc
-rw-r--r--  1 student student  4288 Sep 11  2023 GNUmakefile.in
-rw-r--r--  1 student student   277 Sep 11  2023 HISTORY
-rw-r--r--  1 student student 64592 Sep 11  2023 INSTALL
-rw-r--r--  1 student student  1875 Sep 11  2023 Makefile
-rw-r--r--  1 student student 101920 Sep 11  2023 meson.build
-rw-r--r--  1 student student  6266 Sep 11  2023 meson_options.txt
-rw-r--r--  1 student student  1213 Sep 11  2023 README
drwxrwxr-x 16 student student  4096 Sep 11  2023 src
```

---

## Configuration

We'll set up the configuration process in the directory containing the unpacked PostgreSQL source code using the configure script.

```
student$ cd /home/student/postgresql-16.0
```

Important Note! If rebuilding (e.g., with different configuration parameters), you must first remove files from previous configuration to avoid conflicts:

```
student$ make distclean
```

The configure command accepts a number of parameters. Example:

- `--prefix` — installation directory, default `/usr/local/pgsql`
- `--enable-debug` — to enable debug information

See the documentation for the full list of parameters.

The command also makes use of environment variables. For example, `CC` and `CFLAGS` configure the C compiler.

By default, the configure script prepares compilation and installation, placing all executable files, libraries, and other PostgreSQL files in the directory: `/usr/local/pgsql`. Typically, the installation directory is accessible for writing only to the OS superuser.

For demonstration purposes, we use the `--prefix` option to specify the server software installation directory in the `pgsql16` subdirectory of the student user's home directory. We are also using port 5555 instead of the default 5432:

```
student$ ./configure --prefix=/home/student/pgsql16 --with-pgport=5555
```

The output of this command is extremely verbose, so we are skipping it entirely.

## PostgreSQL Compilation and Installation

Options:

- `make` — build only the server
- `make world` — build the server, all extensions and documentation

Let's compile the server. Depending on your computer performance, this may take minutes or even tens of minutes.

```
student$ make
```

Now, install the server.

```
student$ make install
```

To compile the extensions, repeat the `make` and `make install` commands from within the `contrib` directory.

```
student$ cd contrib
```

```
student$ make
```

```
student$ make install
```

If you build the server together with the extensions and docs (make world), you can install everything with just the make install-world command.



## Tool

initdb

## Features

a new database cluster cannot belong to the OS superuser

PGDATA is a convenient variable that represents the cluster directory

configuration files are created in the PGDATA directory

🚧 enabling checksum calculation for data pages is a good idea

After installing the server, you need to create a database cluster. The `initdb` tool does that for you.

For security reasons, the directory in which the cluster is initialized cannot belong to the OS superuser. Usually, the cluster ownership is assigned to the `postgres` user.

The cluster owner can define the `PGDATA` environment variable pointing to the cluster directory. This variable is used by some server tools when they need to find out the location of the cluster. Such tools include `initdb`, as well as `pg_ctl`, the main server management tool, which we will discuss in a bit.

During cluster initialization, `initdb` creates configuration files in the `PGDATA` directory. We will talk more about configuration files in another lesson.

`initdb` has many keys that affect its operation. One important key is `-k` or `--data-checksums`. It enables or disables calculating checksums for data pages. The checksum check is performed when accessing any data page in the cluster. This slightly reduces performance, but allows you to quickly detect data corruption.

For more details, see <https://postgrespro.com/docs/postgresql/16/app-initdb>



## Creating a Cluster

Now, let's create a data directory. In production, it's a good practice to have a separate OS user as the owner of the data directory. Here, we will just use the student user and the /home/student/pgsql16 directory.

```
student$ mkdir /home/student/pgsql16/data
```

This directory is often referred to as PGDATA. PGDATA is the environment variable pointing to it, and it's a convenient shortcut.

```
student$ export PGDATA=/home/student/pgsql16/data
```

All server tools and utilities are located in the bin directory. We recommend you to add it into the PATH variable:

```
student$ export PATH=/home/student/pgsql16/bin:$PATH
```

initdb is a tool used to initialize a database cluster.

- In the -U option, you can specify the name of the DBMS superuser. If you do not specify it, then the OS username is used, in our case student. Usually this user is postgres, so let's set this name explicitly.
- The -k option includes page checksum calculation, which allows timely detection of data corruption.
- If PGDATA is not set, specify a directory for the data in the -D option. In our case, we do not actually need to specify this option.

```
student$ initdb -U postgres -k -D /home/student/pgsql16/data
```



## Tool

pg\_ctl

## Main operations

starting, stopping and checking the status of the server

reloading configuration parameters

switching to a standby

The main server management operations include starting and stopping the server, getting the current status of the server, reloading the configuration, and some others.

pg\_ctl, a stock PostgreSQL tool, is designed to perform these actions.

pg\_ctl should be run as the owner of the database cluster.

For more details about server management for database administrators, see:

<https://postgrespro.com/docs/postgresql/16/app-pg-ctl>

<https://postgrespro.com/docs/postgresql/16/runtime>

## Server Management

We can start the server now.

- In the -l option, specify the server message log file.
- The -D option is omitted because the PGDATA variable is set.

```
student$ pg_ctl start -l /home/student/logfile
```

Let's verify that the server is running: use the psql tool to connect to the server and return current time:

```
student$ psql -U postgres -p 5555 -c 'SELECT now();'
```

```
      now
-----
2025-09-24 16:56:04.020244+03
(1 row)
```

---

The following command stops the server:

```
student$ pg_ctl stop
```

Package installation is usually preferable

Linux (ubuntu<sup>®</sup>, Debian, Red Hat, SUSE and others)

included in the OS distribution

repository (yum, apt) or RPM/DEB packages

FreeBSD, OpenBSD

packages from the Ports and Packages Collection

Mac OS X

Windows

The preferred option is to use ready-made packages, since in this case a clear, supported and easily updated installation is obtained.

Packages exist for most popular systems. Each of them may have its own features that you should get to know before installing.

Package repositories of some systems already include PostgreSQL, but usually not the latest version. The latest version is always available in the PostgreSQL Global Development Group (PGDG) repository:

<https://www.postgresql.org/download/>

We will be working with the Ubuntu PostgreSQL package.



# Creating a Cluster



## Tool

pg\_createcluster  initdb

## Features

initialization is performed when installing the package, creates a cluster named “main”



checksum calculation is not enabled by default

defines the location of server configuration and log files

sets up automatic server startup at OS startup

12

pg\_createcluster is a wrapper for the initdb tool designed to initialize a cluster in Ubuntu.

The help for the pg\_createcluster command can be obtained using man:

```
$ man pg_createcluster
```

pg\_createcluster is executed automatically when the package is installed and creates a database cluster named “main”.

Note that cluster initialization is done with checksum calculation for data pages disabled.

Executables, configuration files and the server log are placed in accordance with the Ubuntu practices.

In addition, the installation sets up the PostgreSQL server to start and stop when Ubuntu starts and stops.

To delete a cluster, the pg\_dropcluster tool is used.

Both pg\_createcluster and pg\_dropcluster are specific to Ubuntu.

In other systems, you need to explicitly initialize a cluster using initdb and use the appropriate operating system tools.

## Creating a Cluster in Ubuntu

In the provided VM, PostgreSQL is installed from the distribution package:

```
student$ sudo apt install -y postgresql-16
```

PostgreSQL directory:

```
student$ sudo ls -l /usr/lib/postgresql/16
```

```
total 8
drwxr-xr-x 2 root root 4096 Sep 11 08:52 bin
drwxr-xr-x 4 root root 4096 Sep 11 08:53 lib
```

The owner of the server software is the root user.

---

pg\_config shows the parameters the server was compiled with:

```
student$ sudo /usr/lib/postgresql/16/bin/pg_config --configure
```

```
'--build=x86_64-linux-gnu' '--prefix=/usr' '--includedir=${prefix}/include'
'--mandir=${prefix}/share/man' '--infodir=${prefix}/share/info' '--sysconfdir=/etc'
'--localstatedir=/var' '--disable-option-checking' '--disable-silent-rules'
'--libdir=${prefix}/lib/x86_64-linux-gnu' '--runstatedir=/run'
'--disable-maintainer-mode' '--disable-dependency-tracking' '--with-tcl' '--with-perl'
'--with-python' '--with-pam' '--with-openssl' '--with-libxml' '--with-libxslt'
'--mandir=/usr/share/postgresql/16/man' '--docdir=/usr/share/doc/postgresql-doc-16'
'--sysconfdir=/etc/postgresql-common' '--datarootdir=/usr/share/'
'--datadir=/usr/share/postgresql/16' '--bindir=/usr/lib/postgresql/16/bin'
'--libdir=/usr/lib/x86_64-linux-gnu/' '--libexecdir=/usr/lib/postgresql/'
'--includedir=/usr/include/postgresql/' '--with-extra-version= (Ubuntu
16.10-1.pgdg24.04+1)' '--enable-nls' '--enable-thread-safety' '--enable-debug'
'--disable-rpath' '--with-uuid=e2fs' '--with-gnu-ld' '--with-gssapi' '--with-ldap'
'--with-pgport=5432' '--with-system-tzdata=/usr/share/zoneinfo' 'AWK=mawk'
'MKDIR_P=/bin/mkdir -p' 'PROVE=/usr/bin/prove' 'PYTHON=/usr/bin/python3' 'TAR=/bin/tar'
'XSLTPROC=xsltproc --nonet' 'CFLAGS=-g -O2 -fno-omit-frame-pointer
-mno-omit-leaf-frame-pointer -flto=auto -ffat-lto-objects -fstack-protector-strong
-fstack-clash-protection -Wformat -Werror=format-security -fcf-protection
-fno-omit-frame-pointer' 'LDFLAGS=-Wl,-Bsymbolic-functions -flto=auto -ffat-lto-objects
-Wl,-z,relro -Wl,-z,now' '--enable-tap-tests' '--with-icu' '--with-llvm'
'LLVM_CONFIG=/usr/bin/llvm-config-19' 'CLANG=/usr/bin/clang-19' '--with-lz4'
'--with-zstd' '--with-systemd' '--with-selinux' '--enable-dtrace'
'build_alias=x86_64-linux-gnu' 'CPPFLAGS=-Wdate-time -D_FORTIFY_SOURCE=3' 'CXXFLAGS=-g
-O2 -fno-omit-frame-pointer -mno-omit-leaf-frame-pointer -flto=auto -ffat-lto-objects
-fstack-protector-strong -fstack-clash-protection -Wformat -Werror=format-security
-fcf-protection'
```

A database cluster called main is already initialized and is located in /var/lib/postgresql/16/main.

The owner of the directory is postgres user. Here's what the directory contains:

```
student$ sudo ls -l /var/lib/postgresql/16/main
```


```
total 92
drwx----- 1 postgres postgres 4096 Sep 20 19:23 base
drwx----- 1 postgres postgres 4096 Sep 20 16:55 global
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_commit_ts
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_dynshmem
drwx----- 4 postgres postgres 4096 Sep 20 19:24 pg_logical
drwx----- 4 postgres postgres 4096 Sep 20 19:23 pg_multixact
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_notify
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_replslot
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_serial
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_snapshots
drwx----- 1 postgres postgres 4096 Sep 24 16:55 pg_stat
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_stat_tmp
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_subtrans
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_tblspc
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_twophase
-rw----- 1 postgres postgres 3 Sep 20 19:23 PG_VERSION
drwx----- 3 postgres postgres 4096 Sep 20 19:23 pg_wal
drwx----- 2 postgres postgres 4096 Sep 20 19:23 pg_xact
-rw----- 1 postgres postgres 88 Sep 20 19:23 postgresql.auto.conf
-rw----- 1 postgres postgres 130 Sep 24 16:55 postmaster.opts
-rw----- 1 postgres postgres 108 Sep 24 16:55 postmaster.pid
```





## Tool

pg\_ctlcluster

 pg\_ctl

## Main operations

starting, stopping and checking the status of the server

reloading configuration parameters

switching to a standby

In the Ubuntu PostgreSQL package, pg\_ctl is not run directly, but through a special wrapper pg\_ctlcluster. The help for pg\_ctlcluster can be obtained using man:

```
$ man pg_ctlcluster
```

In other systems, pg\_ctl can be used directly.



## Server Management in Ubuntu

With the package installation, PostgreSQL is automatically added to the OS startup list. So, there is no need to start it manually.

You can explicitly control the server using the `pg_ctlcluster` tool, which must be run either as the OS postgres user or via `sudo`.

The `pg_ctlcluster` tool can manage multiple servers of different versions. It has the following parameters:

- 16 — server version number
- main — server name
- Command

Stop:

```
student$ sudo pg_ctlcluster 16 main stop
```

```
student$ pg_lsclusters 16 main
```

Ver	Cluster	Port	Status	Owner	Data directory	Log file
16	main	5432	down	postgres	/var/lib/postgresql/16/main	/var/log/postgresql/postgresql-16-main.log

Start:

```
student$ sudo pg_ctlcluster 16 main start
```

Restart:

```
student$ sudo pg_ctlcluster 16 main restart
```

Get server status:

```
student$ sudo pg_ctlcluster 16 main status
```

```
pg_ctl: server is running (PID: 3518)
/usr/lib/postgresql/16/bin/postgres "-D" "/var/lib/postgresql/16/main" "-c"
"config_file=/etc/postgresql/16/main/postgresql.conf"
```

Reload configuration:

```
student$ sudo pg_ctlcluster 16 main reload
```

---

For a package installation, the server log file is located at:

```
student$ ls -l /var/log/postgresql/postgresql-16-main.log
```

```
-rw-r----- 1 postgres adm 7377 Sep 24 16:56 /var/log/postgresql/postgresql-16-main.log
```

Let's look at the last messages:

```
student$ tail -n 10 /var/log/postgresql/postgresql-16-main.log
```

```
2025-09-24 16:56:08.894 MSK [3465] LOG:  shutting down
2025-09-24 16:56:08.909 MSK [3465] LOG:  checkpoint starting: shutdown immediate
2025-09-24 16:56:09.010 MSK [3465] LOG:  checkpoint complete: wrote 3 buffers (0.0%); 0
WAL file(s) added, 0 removed, 0 recycled; write=0.030 s, sync=0.015 s, total=0.116 s;
sync files=2, longest=0.008 s, average=0.008 s; distance=0 kB, estimate=0 kB;
lsn=0/1921788, redo lsn=0/1921788
2025-09-24 16:56:09.017 MSK [3464] LOG:  database system is shut down
2025-09-24 16:56:09.256 MSK [3518] LOG:  starting PostgreSQL 16.10 (Ubuntu
16.10-1.pgdg24.04+1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu
13.3.0-6ubuntu2~24.04) 13.3.0, 64-bit
2025-09-24 16:56:09.256 MSK [3518] LOG:  listening on IPv4 address "127.0.0.1", port 5432
2025-09-24 16:56:09.270 MSK [3518] LOG:  listening on Unix socket
"/var/run/postgresql/.s.PGSQL.5432"
2025-09-24 16:56:09.306 MSK [3521] LOG:  database system was shut down at 2025-09-24
16:56:08 MSK
2025-09-24 16:56:09.334 MSK [3518] LOG:  database system is ready to accept connections
2025-09-24 16:56:11.946 MSK [3518] LOG:  received SIGHUP, reloading configuration files
```

## PostgreSQL in a virtual environment

- virtualization overhead
- the administrator has full access to the instance

## PostgreSQL as a Service

- offered by many cloud providers
- the provider takes over the administration routine
- limited management, backup and monitoring tools

Virtualization solutions can be used to run databases, including PostgreSQL. The trade off for the convenience of this approach is considerable overhead costs. For high-load systems, any additional intermediate layers between the DBMS and the hardware are undesirable.

In addition, many leading cloud providers offer PostgreSQL as a service (Database as a Service, managed database).

In this case, the provider takes over most of the administration tasks. This limits your control over the instance. Moreover, management capabilities for tasks such as performance monitoring or backup and recovery are limited by the tools offered by the service provider.

We will not consider the features of individual cloud solutions within the course. You can study the documentation offered by different service providers to learn more.

# Takeaways



Two installation options — package or source code

Cloud solutions available

The server is run by a dedicated OS user

Initialize a database cluster before using the server

Specific commands for server management

Enabling checksum calculation for a cluster.

1. Stop the server.
2. Check whether checksums are calculated for the cluster.
3. Enable checksum calculation.
4. Start the server.

2, 3. Use the `pg_checksums` tool. To run it, enter the full path:  
`/usr/lib/postgresql/16/bin/pg_checksums`

<https://postgrespro.com/docs/postgresql/16/app-pgchecksums>

## 1. Stopping the Server

```
student$ sudo pg_ctlcluster 16 main stop
```

## 2. Verification

To see if page checksum calculation is enabled, run the `pg_checksums` utility with the `--check` option:

```
student$ sudo /usr/lib/postgresql/16/bin/pg_checksums --check -D /var/lib/postgresql/16/main
pg_checksums: error: data checksums are not enabled in cluster
```

## 3. Enabling Checksum Calculation

Run `pg_checksums` with the `--enable` option:

```
student$ sudo /usr/lib/postgresql/16/bin/pg_checksums --enable -D /var/lib/postgresql/16/main

Checksum operation completed
Files scanned: 1244
Blocks scanned: 3749
Files written: 1026
Blocks written: 3749
pg_checksums: syncing data directory
pg_checksums: updating control file
Checksums enabled in cluster

Checksum calculation is now enabled.
```

## 4. Starting the Server

```
student$ sudo pg_ctlcluster 16 main start
```

1. Install PostgreSQL from source code as shown in the demo.  
Create a database cluster, start the server.  
Make sure the server is running.  
Stop the server.

## **1. Compiling PostgreSQL from Source Code**

All necessary commands were shown during the demo, but were not executed. Repeat the process by yourself within the provided virtual machine.

Note that compiling of the source code may take a while (several to tens of minutes, depending on your computer resources).