

# Simple PostgreSQL Installation Guide

**Roberto Mello (rmello@fslc.usu.edu)**

A very simple PostgreSQL installation guide to aid OpenACS users, based on the PostgreSQL Administrators Guide. Not intended to be a full-blown guide to every imaginable architecture on the planet. This was written in a *GNU/Linux* box, so some adaptation may be necessary for other platforms. This guide does not apply for Windows environments. Check the PostgreSQL documentation for more details on installing PostgreSQL on Windows and other platforms ([Chapter 5 of the Administrators Guide](#)).

---

## Table of Contents

1. [Before you start](#)
    - 1.1. [Should I compile or use a binary \(RPM, DEB\) ?](#)
  2. [Compiling and Installing the best free RDBMS](#)
    - 2.1. [Preliminaries](#)
    - 2.2. [Optional - How to increase the blocksize in PostgreSQL \[1\]](#)
    - 2.3. [Compiling and installing](#)
    - 2.4. [Getting it to start with every boot](#)
    - 2.5. [Testing](#)
  3. [Installing Binary Packages](#)
    - 3.1. [RPMs](#)
    - 3.2. [DEBs](#)
  4. [Acknowledgements](#)
- 

[Next](#)

Before you start

# 1. Before you start

- You will need GNU Make to build PostgreSQL. GNU Make is the default in GNU/Linux systems, in other systems it may be called gmake. We are simply going to refer to it as `make`. You can download GNU Make from <ftp://ftp.gnu.org>.
- You will need a C compiler. GCC (GNU Compiler Collection) is what I am using.
- You will need C/C++ development libraries. On GNU/Linux systems, this usually means `glibc` and `glibc-devel`.
- For convenience, install GNU Readline and its development packages (usually `readline` and `readline-devel`).
- (Optional) If you are going to use `psql` from an X Terminal, then install the X development libraries.
- A minimum of 35 Mb of disk space is necessary to build PostgreSQL (including the sources). Of course, you will need more to create databases, so be generous :-)

## 1.1. Should I compile or use a binary (RPM, DEB) ?

Compiling gives you more flexibility because you can compile PostgreSQL with some extra options (and we will mention some here). It is not a super trivial process, but it is not too hard. Following this guide will help a lot, but some understanding of what's going on is expected of you.

If you are completely new to UNIX systems or if you just want to get it up and running fast, then go with a binary installation, such as the ones provided by RPMs (Red Hat packages, provided in most GNU/Linux distributions) and DEBs (Debian GNU/Linux packages).

## 2. Compiling and Installing the best free RDBMS

### 2.1. Preliminaries

As *root*, create a user and group (if you haven't done so before) for PostgreSQL. This is the account that PostgreSQL will run as since it will not run as root. Also give the postgres user a password:

```
groupadd web
useradd -g web postgres -d /usr/local/pgsql
passwd postgres
```

Download PostgreSQL from the mirror closest to you. The list of mirrors is at <http://www.postgresql.org/sites.html>. You can download it in one big file (Ü 8 Mb) or in smaller files (base, support, doc, test). Put it in a temp directory such as /tmp.

Untar with the command:

```
tar xzvf postgresql-xxxxx.tar.gz
```

Repeat the tar command for each file if you are downloading the multiple-file distribution.

### 2.2. Optional - How to increase the blocksize in PostgreSQL [\[1\]](#)

This is not required to compile PostgreSQL.

By default PostgreSQL is compiled with a blocksize of 8 Kb. You can compile PostgreSQL to have 16 Kb blocksize instead, which will allow for bigger text and lztext data types. NOTE: If you've built PostgreSQL in that directory before, you need to a) delete the config.cache file (created by autoconf) and b) do a make clean to give it a fresh start.

Here's how you do it:

1. Go into the src/include directory of the PostgreSQL source distribution.
2. Edit the file config.h.in
3. Look for the line `#define BLCKSZ 8192` and change it to `#define BLCKSZ 16384`
4. Save it.

## 2.3. Compiling and installing

\* You will need to run the command `./configure` from within the `src/` directory of the PostgreSQL distribution. This will create the files with the options that will be used to compile PostgreSQL specifically to your machine (`./configure --help` will show you all the available options).

\* If you are going to access `psql` (the PostgreSQL's interactive query tool) from within an X terminal (such as `xterm`, `kterm`, `Eterm`,...) and you have the X development libraries installed then do:

```
./configure --with-x
```

Otherwise simply do a `./configure`

\* After it's done fire up a:

```
make
```

\* The compilation process will take place. Go look at OpenACS and see on what you can help make the toolkit even more butt-kicking. After it's done, you should see a message like this:

```
All of PostgreSQL is successfully made. Ready to install.
```

\* If you are upgrading your PostgreSQL, follow the instructions on [Chapter 5 of the Administrators Guide](#).

\* Create the directories to house PostgreSQL:

```
mkdir /usr/local/pgsql
chown postgres.web /usr/local/pgsql
```

\* Become the postgres user: `su - postgres`

\* Then install the PG executable files and libraries by running:

```
make install
```

\* Then you need to initialize the database templates and other things by doing this:

```
# su - postgres
$ cd /usr/local/pgsql
$ mkdir data
$ cd bin
$ ./initdb -D /usr/local/pgsql/data
```

\* Tell your system how to find the new shared libraries. This can be accomplished in two ways:

Editing the file `/etc/profile` and adding the lines:

```
LD_LIBRARY_PATH=/usr/local/pgsql/lib
export LD_LIBRARY_PATH
```

OR editing the file `/etc/ld.so.conf` and adding the line:

```
/usr/local/pgsql/lib
```

and then running the command `/sbin/ldconfig`.

\* You probably want to install the man and HTML documentation. Type

```
cd /usr/local/src/pgsql/postgresql-7.0/doc (or wherever you untarred it)
make install
```

This will install files under `/usr/local/pgsql/doc` and `/usr/local/pgsql/man`.

\* Setup some environment variables to make your life easier:

You can do this by editing the `/etc/profile` file and including these lines:

```
PATH=$PATH:/usr/local/pgsql/bin
MANPATH=$MANPATH:/usr/local/pgsql/man
export MANPATH
```

\* As the user *postgres*, create the database installation (the working data files). It can be anywhere that is writable by the *postgres* user (usually `/usr/local/pgsql/data`):

```
mkdir /usr/local/pgsql/data
/usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
```

\* Start the database server with the command:

```
/usr/local/pgsql/bin/postmaster -D /usr/local/pgsql/data
```

This will start the server in the foreground (CTRL+C to stop it). To make it detach to the background, you can use the `-S` option, but then you won't see any log messages the server produces. A better way to put the server in the background is :

```
nohup /usr/local/pgsql/bin/postmaster -D /usr/local/pgsql/data < /dev/null >>
server.log 2>>1 &
```

## 2.4. Getting it to start with every boot

Unfortunately there's no easy answer here. Every system (even GNU/Linux distributions) is a bit different than the other. In Red Hat / Mandrake systems I modify the *postgresql* startup script that comes with the RPM distribution to fit a compiled version.

In the *contrib/linux* directory of the PostgreSQL source distribution there's a sample startup script that you can use to include in your `/etc/init.d` directory.

## 2.5. Testing

After you've started the database server, try creating a database and connecting to it through psql:

```
createdb mydb  
psql mydb
```

If you see a prompt after a few messages, you're set ! Congratulations.

### Notes

[1] This is unnecessary in PG 7.1 as it is rid of the blocksize limit.

---

[Prev](#)

Before you start

[Home](#)

[Next](#)

Installing Binary Packages

## 3. Installing Binary Packages

### 3.1. RPMs

\* Download the RPMs from the PostgreSQL mirror closest to you (list at <http://www.postgresql.org/sites.html>)

The RPMs come split into separate packages so you can install only what you need. Download the packages:

`postgresql-7.xxx.i386.rpm`

`postgresql-devel-7.xxx.i386.rpm`

`postgresql-server-7.xxx.i386.rpm`

`postgresql-test-7.xxx.i386.rpm` (for the regression tests).

(if you want `pgaccess`, PostgreSQL graphical tool, also install `postgresql-tk-7.xxx.i386.rpm`)

\* Install them in the sequence above, with the command: (as root)

```
rpm -ivh postgresql-xxx.i386.rpm
```

If it asks for dependency packages, install them from the CD or after downloading them (from `rpm.net` for example).

The RPM packages will very nicely create the `postgres` user and group, set the environment variables, install the documentation and other goodies for you. Cozy huh ? Thank Lamar Owens for that.

\* Then start the database server with the command:

```
/etc/rc.d/init.d/postgresql start
```

If everything went fine, you should be able to, as user *postgres*, create a database and connect to it through `psql`:

```
su - postgres
createdb mydb
psql mydb
```

## 3.2. DEBs

\* The DEB packages come split into separate packages so you can install only what you need.

As root, fire up *dselect* and go into *Select* .

Do a `/ postgresql` to search for postgresql packages. Search again by pressing `\`

Mark for installation (by pressing `+`) the following packages:

postgresql

postgresql-pl

postgresql-client

postgresql-dev

postgresql-doc

postgresql-test

(if you want pgaccess, PostgreSQL graphical tool also install the package pgaccess)

Then hit `Enter` to get out of *dselect* and go into *Install* .

That's it. It should download, install and start the server for you. Hard huh ? Thanks to Oliver Elphick for providing these packages.

\* If the setup process doesn't start the server, you can start it with: (as root)

```
/etc/init.d/postgresql start
```

\* If everything went fine, you should be able to, as user *postgres*, create a database and connect to it through *psql*:

```
su - postgres
createdb mydb
psql mydb
```

---

[Prev](#)

[Home](#)

[Next](#)

Compiling and Installing the best  
free RDBMS

Acknowledgements

[Prev](#)

---

## 4. Acknowledgements

Several people have contributed to this document in either content or error reporting and I would like to thank them. Please let me know if I forgot to include your name.

Contributors (in no specific order):

Don Baccus, Mark Rohn, Bob O'Connor.

---

[Prev](#)[Home](#)

Installing Binary Packages