

PostgreSQL

Administrator's Guide

Release 5.0

Table of Contents

Preface	iii
1. Installation	1
1.1. Download and Build	1
1.2. Install	1
1.3. Automatic startup	2
1.4. Create your first database	3
1.5. Dump and restore database	3
2. Upgrade	5
2.1. Download and Build	5
2.2. Upgrade	5
3. Downgrade	7
3.1. Downgrade	7
4. PGFouine Third Party	8
4.1. Download and install	8
4.2. Configure	8
4.3. How To Use	9

Preface

PostgreSQL, the highly scalable, SQL compliant, open source object-relational database management system.

PostgreSQL is an object-relational database management system (ORDBMS) based on POSTGRES, Version 4.2, developed at the University of California at Berkeley Computer Science Department. POSTGRES pioneered many concepts that only became available in some commercial database systems much later.

PostgreSQL is an open-source descendant of this original Berkeley code. It supports a large part of the SQL standard and offers many modern features:

- complex queries
- foreign keys
- triggers
- views
- transactional integrity
- multiversion concurrency control

Also, PostgreSQL can be extended by the user in many ways, for example by adding new

- data types
- functions
- operators
- aggregate functions
- index methods
- procedural languages

And because of the liberal license, PostgreSQL can be used, modified, and distributed by everyone free of charge for any purpose, be it private, commercial, or academic.

The PostgreSQL web site [<http://www.postgresql.org>] carries details on the latest release and other information to make your work or play with PostgreSQL more productive. And The mailing lists are a good place to have your questions answered, to share experiences with other users, and to contact the developers.

1

Installation

1.1. Download and Build

Download and save postgres source archive from <http://www.postgresql.org>

```
cd /usr/local/src
tar -xjf postgresql-X.X.X.tar.bz2
md5sum postgresql-X.X.X.tar.bz2
cd postgresql-X.X.X
./configure --prefix=/usr/local/pgsql-X.X.X --enable-locale --enable-multibyte --enable-syslog
make
make install
```

1.2. Install

Install binaries and create required user and directories.

```
ln -s /usr/local/pgsql-X.X.X /usr/local/pgsql
groupadd -g 53 postgres
useradd -g 53 -u 53 -s /bin/bash -d /home/postgres -m postgres
mkdir -p /var/local/pgsql-X.X.X/data
ln -s /var/local/pgsql-X.X.X /var/local/pgsql
chown postgres /var/local/pgsql/data
mkdir -p /var/log/pgsql
chown postgres /var/log/pgsql
mkdir /var/local/pgsql/dump
```

Create postgres system files.

```
su - postgres
/usr/local/pgsql/bin/initdb -D /var/local/pgsql/data
```

Create default configuration files.

```
mkdir -p /etc/pgsql/conf
mv /var/local/pgsql/data/*conf /etc/pgsql/conf
ln -s /etc/pgsql/conf/* /var/local/pgsql/data
```

Add postgres commands in PATH, modify /etc/profile file.

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

Customize default configuration, modify `/etc/pgsql/postgresql.conf` file.

```
max_connections = 512
superuser_reserved_connections = 2
shared_buffers = 65536 # min 16 or max_connections*2, 8KB each
work_mem = 32768 # min 64, size in KB
effective_cache_size = 131072 # typically 8KB each
max_fsm_pages = 100000 # min max_fsm_relations*16, 6 bytes each
```

Increase system limit for postgres, modify `/etc/sysctl.conf` file.

```
kernel.shmmax = 1073741824
```

And modify `/etc/security/limits.conf` file.

```
postgres soft nofile 4096
postgres hard nofile 65536
```

1.3. Automatic startup

Copy default startup script in standard directory.

```
cd /usr/local/src/postgresql-X.X.X/contrib/start-scripts
cp linux /etc/init.d/postgres
cd /etc/init.d
```

Customize this file `/etc/init.d/postgres`.

```
#!/bin/sh
## EDIT FROM HERE
# Installation prefix
prefix=/usr/local/pgsql
# Data directory
PGDATA="/var/local/pgsql/data"
# Who to run pg_ctl as, should be "postgres".
PGUSER=postgres
# Where to keep a log file
PGLOG="/var/log/pgsql/serverlog"
# Startup Options
PGOPTIONS="-i"
## STOP EDITING HERE
# Check for echo -n vs echo \c
if echo '\c' | grep -s c >/dev/null 2>&1 ; then
    ECHO_N="echo -n"
    ECHO_C=""
else
    ECHO_N="echo"
    ECHO_C='\c'
fi
# The path that is to be used for the script
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
# What to use to start up the postmaster
DAEMON="$prefix/bin/pg_ctl"
set -e
# Only start if we can find pg_ctl.
test -f $DAEMON || exit 0
# Parse command line parameters.
case $1 in
```

```

start)
    $ECHO_N "Starting PostgreSQL: "$ECHO_C
    su - $PGUSER -c "$DAEMON start -D '$PGDATA' -o '$PGOPTIONS' -s -l $PGLOG"
    echo "ok"
    ;;
stop)
    echo -n "Stopping PostgreSQL: "
    su - $PGUSER -c "$DAEMON stop -D '$PGDATA' -s -m fast"
    echo "ok"
    ;;
restart)
    echo -n "Restarting PostgreSQL: "
    su - $PGUSER -c "$DAEMON restart -D '$PGDATA' -s -m fast -l $PGLOG"
    echo "ok"
    ;;
reload)
    echo -n "Reload PostgreSQL: "
    su - $PGUSER -c "$DAEMON reload -D '$PGDATA' -s"
    echo "ok"
    ;;
status)
    su - $PGUSER -c "$DAEMON status -D '$PGDATA'"
    ;;
*)
    # Print help
    echo "Usage: $0 {start|stop|restart|reload|status}" 1>&2
    exit 1
    ;;
esac
exit 0

```

Add execution right and register automatic startup.

```

chmod a+x /etc/init.d/postgres
update-rc.d -f postgres defaults

```

1.4. Create your first database

Start database service.

```
/etc/init.d/postgres start
```

Create database in unicode charset (recommended).

```

su - postgres
/usr/local/pgsql/bin/createdb -E utf-8 test

```

1.5. Dump and restore database

An example of dump command.

```
/usr/local/pgsql/bin/pg_dump -U postgres -b -Fc -o -v -f /var/local/pgsql/dump/test.dump test
```

An example of restore command.

```
/usr/local/pgsql/bin/pg_restore -U postgres -Fc -O -d test /var/local/pgsql/dump/test.dump
```

2

Upgrade

2.1. Download and Build

Same as installation section.

2.2. Upgrade

Backup all old database. (optional)

```
#!/bin/sh -x
BACKUP_DIR=/data1/backup-update-postgres
DB_LIST=`/usr/local/pgsql/bin/psql -U postgres template1 -R " " -At -c "select datname from pg_database"
if [ ! -d $BACKUP_DIR ] ; then
    mkdir -p $BACKUP_DIR
fi
# save all databases
for DB in $DB_LIST
do
    if [ "$DB" != "template0" ]; then
        if [ "$DB" != "template1" ]; then
            if [ "$DB" != "postgres" ]; then
                echo Dumping $DB
                /usr/local/pgsql/bin/pg_dump -U postgres -b -Fc -o -f $BACKUP_DIR/$DB.dump $DB
            fi
        fi
    fi
done
```

List all database in order to recreate after upgrade.

```
/data1/backup-update-postgres# /usr/local/pgsql/bin/psql -U postgres -l > listdb.txt
```

Upgrade old version to the new version.

```
/etc/init.d/postgres stop
rm /usr/local/pgsql
ln -s /usr/local/pgsql-X.X.X /usr/local/pgsql
mkdir -p /var/local/pgsql-X.X.X/data
mkdir -p /var/local/pgsql-X.X.X/dump
rm /var/local/pgsql
ln -s /var/local/pgsql-X.X.X /var/local/pgsql
chown postgres /var/local/pgsql/data
su - postgres
/usr/local/pgsql/bin/initdb -D /var/local/pgsql/data
```

```
cp -r /etc/pgsql/conf /etc/pgsql/conf.YYYYMMDD
mv /var/local/pgsql/data/*conf /etc/pgsql/conf
ln -s /etc/pgsql/conf/* /var/local/pgsql/data
/etc/init.d/postgres start
```

Now you can recreate and restore your old databases, see backup and restore section. Or use this example script.

```
#!/bin/sh -x
BACKUP_DIR=/data1/backup-update-postgres
DB_LIST=`/usr/local/pgsql/bin/psql -U postgres template1 -R " " -At -c "select datname from pg_database"
if [ ! -d $BACKUP_DIR ] ; then
    mkdir -p $BACKUP_DIR
fi
# save all databases
for DB in $DB_LIST
do
    if [ "$DB" != "template0" ]; then
        if [ "$DB" != "template1" ]; then
            if [ "$DB" != "postgres" ]; then
                echo Restoring $DB
                /usr/local/pgsql/bin/pg_restore -U postgres -Fc -O -d $DB $BACKUP_DIR/$DB.dump
            fi
        fi
    fi
done
```

Maybe an update or JDBC driver and other database connector is required, read official PostgreSQL documentation.

3

Downgrade

3.1. Downgrade

Downgrade the current version (X.X.X) to the previous (Y.Y.Y).

```
/etc/init.d/postgres stop
rm /usr/local/pgsql
ln -s /usr/local/pgsql-Y.Y.Y /usr/local/pgsql
rm /var/local/pgsql
ln -s /var/local/pgsql-Y.Y.Y /var/local/pgsql
rm /etc/pgsql/conf/*
cp -r /etc/pgsql/conf.YYYMMDD /etc/pgsql/conf
/etc/init.d/postgres start
```

4

PGFouine Third Party

4.1. Download and install

Download and save PGFouine binary archive from <http://pgfouine.projects.postgresql.org> [<http://pgfouine.projects.postgresql.org/>]

```
cd /usr/local
tar xzf pgfouine-X.Y.tar.gz
ln -s /usr/local/pgfouine-X.Y /usr/local/pgfouine
#install required libraries
apt-get install php4-cli
apt-get install libgd2-dev php4-gd
```

4.2. Configure

We must modify postgres configuration in order to provide correct log level. We choose syslog logging, it's better for the big requests. Edit and add lines in `/etc/pgsql/conf/postgresql.conf`

```
#-----
# ERROR REPORTING AND LOGGING
#-----
log_destination = 'syslog'
redirect_stderr = off
log_directory = '/var/log/pgsql'
log_filename = 'postgresql-%Y-%m-%d_%H%M%S.log'
log_min_duration_statement = 200 # replace 200 by N ms.
silent_mode = on
log_statement = 'none'
```

`/etc/syslog.conf`

```
local0.* /var/log/pgsql/stats.log
#ajouter local0.none à la fin
*.=info;*.=notice;*.=warn;\
    auth,authpriv.none;\
    cron,daemon.none;\
    mail,news.none;local0.none /var/log/messages
```

Restart services.

```
/etc/init.d/syslogd restart
/etc/init.d/postgres restart
```

4.3. How To Use

Generate PGFouine reports of PostgreSQL activities.

```
php pgfouine.php -debug \  
-file /var/log/syslog \  
-title "PostgreSQL Database Report : lm-proddb-01" \  
-format html-with-graphs \  
-report report-queries.html=overall,bytype,slowest,n-mosttime,n-mostfrequent,n-slowestaverage \  
-report report-hourly.html=overall,hourly \  
-report report-errors.html=overall,n-mostfrequenterrors
```