INSTALL POSTGRESQL & POSTGIS for QGIS

by Lex Berman
[document version: 12 Nov 2011]
URL: http://dbr.nu/qgis/docs/postgis_install_on_snow_leapord.pdf

CASE STUDY: MAC OS X (Snow Leapord) on 64bit Intel Duo Core MacBook
ASSUMPTION: QGIS, GDAL, Freetype, Cairo, GSL and GRASS are already installed
REQUIREMENTS: ROOT PASSWORD, KNOWLEDGE OF TERMINAL AND VI EDITOR

RELATED DOC: INSTALL QGIS ON SNOW LEAPORD
RELATED DOC URL: http://dbr.nu/qgis/docs/qgis_install_on_snow_leapord.pdf

CONVENTIONS:
$ at beginning of line = Terminal Command executed as regular logged in user
# at beginning of line = means Terminal Command executed as superuser [root]
[ ] square brackets indicate term to be supplied by you (like "[username]" change to "faust")
-> indicates the response expected from the Terminal Command, multiple lines
// at beginning of line = means comments, not commands

Before installing please consider these descriptions to get an idea of the install process:

http://linfiniti.com/2011/05/running-posgresql-8-4-and-9-0-side-by-side-on-ubuntu/
http://www.enterprisedb.com/resources-community/pginst-guide

Overview:

1) Prepare System for Install
2) Install PostgreSQL
3) Set PATH for PostgreSQL excecutables
4) Create test database
4) Install PostGIS
5) Create PostGIS instance for PostgreSQL test database
6) Install QGIS SPIT [Shapefile to PostGIS Import Tool] Plugin
7) Setup PostGIS connection and test Import and PostGIS Layer Access

Section 1 - Prepare System for Install

// Mac OS X ships with shared memory settings which are too low for running
// PostgreSQL by default. The following instructions were adapted from:
// http://crafted-software.blogspot.com/2010/11/shared-memory-settings-for-postgres-on.html
// open up Terminal and view your system default for shared memory
$ sysctl kern.sysv
-> kern.sysv.shmmax: 4194304
-> kern.sysv.shmmin: 1
-> kern.sysv.shmmni: 32
-> kern.sysv.shmseg: 8
-> kern.sysv.shmall: 1024
-> kern.sysv.semmni: 87381
-> kern.sysv.semmns: 87381
-> kern.sysv.semmnu: 87381
->kern.sysv.semmsl: 87381
-> kern.sysv.semume: 10

// you need to boost the .shmmax & .shmall by factors explained at the URL above
// the number used must be a correct multiple, see URL above for other settings

// for quick install, you can use the (minimal) changes that worked for me, below
// you need to either edit or create a file at /etc/sysctl.conf (as superuser)
$ su -
-> Password: [rootPassword]
-> root#
cd /etc
vi sysctl.conf
// enter the following text (which boosts your system memory defaults by factor of 8)
kern.sysv.shmmax: 33554432
kern.sysv.shmmin: 1
kern.sysv.shmmni: 32
kern.sysv.shmseg: 8
kern.sysv.shmall: 8192

//save the edited sysctl.conf file
// check the file permissions of the newly created file
ls -al
-> -rw-r--r-- 1 root wheel 111 Nov 12 17:19 /etc/sysctl.conf

// you may have to chown and chmod the file to get these correct… though I didn't need to
// your system should now be ready to run PostgreSQL for small database instances

Section 2 - Downloading the packages for the Install
// you may send thanks, flowers, chocolate, money, etc to William Kyngesburye for
// creating all these amazing packages for Mac OS X !   thanks Kyngchaos!

URL for .dmg packages from: http://www.kyngchaos.com/software/postgres

PostgreSQL 9.1.1-1
PostGIS 1.5.3-2 for Postgres 9.1

1) install the postgresql 9.1.1 .dmg
// if you check for which version is installed you probably won't find it: PATHs are not set
$ pg_config --version
-> -sh: pg_config: command not found

$cd /usr/local/bin
$ls -al p*
$lrwxr-xr-x 1 root wheel 9 Nov 12 17:38 pgsql -> pgsql-9.1

// postgres is there in /pgsql-9.1 and is symlinked from pgsql
// now you want to create a link for /usr/local to see the pg_config executable

$cd /usr/local
#sudo ln -s /usr/local/pgsql/bin/pg_config
ls -al pg*
-> lrwxr-xr-x 1 root wheel 30 Nov 12 18:25 pg_config -> /usr/local/pgsql/bin/pg_config

// now you should get the response from pg_config
pg_config --version
-> PostgreSQL 9.1.1
 // ** woot!

// now we can move on to creating a user – but not quite...

```
# createuser -superuser [yourUsername] -U postgres
-> -sh: createuser: command not found

# echo $PATH
-> /usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin:/usr/X11/bin

// okay basic PATH to pgsql is still missing, we can create a global path in /etc/paths.d

#cd /etc/paths.d
# vi psql
/usr/local/pgsql/bin

// now there should be at least the X11 and the psql path files in the /etc/paths.d/
# ls -l
-rw-r--r--  1 root  wheel  13 Jul 11  2009 X11
-rw-r--r--  1 root  wheel  21 Nov 12 19:00 psql
# more psql
/usr/local/pgsql/bin

close existing terminal, then fire up new terminal
$ which psql
/usr/local/pgsql/bin/psql
userName$ psql
psql: FATAL:  role "userName" does not exist
$ su -
-> Password: [rootPassword]
root# psql
psql: FATAL:  role "root" does not exist

// so psql is in the PATH but not running yet because we have not created the psql user
// try "createuser --help" for more information.
//  note double – hyphen before parameter!  Single hyphen - won't work
//  create a user with your Mac login / username as the postgres “yourUsername”
//  note, this worked for my username combined with the root password in later steps

#createuser --superuser [yourUsername] -U postgres

//exit from superuser to see if your username now has rights to create a database
#exit
$createdb [testDBname]
$ psql -h localhost -U [yourUsername] -d [testDBname]
 psql (9.1.1)
 Type "help" for help.

 testDBname=# \du
                          List of roles
 Role name |                 Attributes                 | Member of
-----------+--------------------------------------------+-----------
 yourUsername       | Superuser, Create role, Create DB, Replication | {}
 postgres  | Superuser, Create role, Create DB, Replication | {}

//  looks like we are running, woot!
//  now you are logged into psql, you can run commands
//  an old cheatsheet that mostly still works:  http://bit.ly/k0ecVN

 testDBname=# \l
```

```
                              List of databases
    Name      |  Owner   | Encoding | Collate | Ctype |   Access privileges
--------------+----------+----------+---------+-------+-----------------------
 postgres     | postgres | UTF8     | en_US   | en_US |
 template0    | postgres | UTF8     | en_US   | en_US | =c/postgres          +
              |          |          |         |       | postgres=CTc/postgres
 template1    | postgres | UTF8     | en_US   | en_US | =c/postgres          +
              |          |          |         |       | postgres=CTc/postgres
 testDBname | yourUsername  | UTF8     | en_US   | en_US |
(4 rows)
```

Section 3 - Installing postGIS

Instructions: http://users.qgis.org/planet/user/2/tag/postgres%20%26%20postgis/
KyngChaos README: http://www.dbr.nu/qgis/docs/KyngChaos_PostGIS_1-5-3_ReadMe.pdf

install PostGIS .dmg

// make sure PostGIS is now installed in the PostgreSQL contrib folder

```
#cd /usr/local/pgsql/share/contrib
# ls
postgis-1.5
```

// looks okay, now run:

```
# /usr/local/pgsql/bin/psql -U postgres -d testDBname -f
# /usr/local/pgsql/share/contrib/postgis-1.5/postgis.sql
SET
BEGIN
CREATE FUNCTION
CREATE FUNCTION
CREATE FUNCTION
CREATE FUNCTION
CREATE TYPE
…
…
DROP FUNCTION
DROP FUNCTION
DROP FUNCTION
```

// if you get long return of postgis.sql to screen, like above, it's good. then run:

```
# /usr/local/pgsql/bin/psql -U postgres -d testDBname -f
# /usr/local/pgsql/share/contrib/postgis-1.5/spatial_ref_sys.sql
BEGIN
INSERT 0 1
INSERT 0 1
INSERT 0 1
…
…
INSERT 0 1
INSERT 0 1
INSERT 0 1
```

```
COMMIT
ANALYZE

//  you should get the long list of spatial ref insertions, like above.
//  now check your database just to make sure its listening on the default port

$ psql -p 5432 -l
                          List of databases
    Name      |  Owner   | Encoding | Collate | Ctype |   Access privileges
--------------+----------+----------+---------+-------+-----------------------
 postgres     | postgres | UTF8     | en_US   | en_US |
 template0    | postgres | UTF8     | en_US   | en_US | =c/postgres          +
              |          |          |         |       | postgres=CTc/postgres
 template1    | postgres | UTF8     | en_US   | en_US | =c/postgres          +
              |          |          |         |       | postgres=CTc/postgres
testDBname | mberman  | UTF8     | en_US   | en_US |
(4 rows)

//  it's not clear to me if the following are necessary or not

$ /usr/local/pgsql/bin/createlang -U postgres plpgsql testDBname
-> createlang: language "plpgsql" is already installed in database "testDBname"
$ psql -U postgres -f /usr/local/pgsql/share/lwpostgis.sql  template_postgis
```

Section 4 - Testing the database connection from QGIS

// now it is time to test the postgreSQL and PostGIS setup in QGIS

launch QGIS
under QGIS Plugin Manager, install “SPIT” (Shapefile to PostGIS Import Tool)
after installing the plugin, launch it (from the BlueElephant button now on the menu)
create a server connection by clicking the NEW button
choose a global schema for connection "public"

- Name (create a name for your server connection)
- Service (leave blank for now)
- Host (= localhost)
- Port (= 5432)
- Database (=testDBname)
- SSL mode (=disable)
- Username (yourUsername)
- Password (your password)

click TEST CONNECT

if the test connection was successful hit OKAY, then CONNECT
after entering your password [possibly the root password] you should be connected

- Now you can use the SPIT tool to upload a Shapefile to the PostGIS
- click the ADD button
- browse to a shapefile (note, default is UTF8 encoding), then click OPEN
- the uploaded file will appear on the Filename list
- hit OKAY to leave the SPIT tool

now you can try adding the PostGIS Layer from the Main QGIS menu
- click on the Add PostGIS layer button (the Blue Database icon)
- after clicking on Add PostGIS layer, CONNECT to the new local postGIS server
- the table list should include the layer just imported to postGIS
- click on the layer you want to add, then the ADD button
- the dialog will close and the layer should now appear in the QGIS map view
- open the attribute table to make sure all is well

// note, this worked for UTF8 character set, as well as other encodings, as long as I properly selected the encoding during in SPIT tool upload setting

// woot! You've got QGIS and PostGIS.