**Purpose**

[PostgreSQL](https://info.crunchydata.com/blog/topic/postgresql) [PostGIS](https://info.crunchydata.com/blog/topic/postgis) New features and better performance get a lot of attention, but one of the relatively unsung improvements in PostGIS over the past ten years has been inclusion in standard software repositories, making installation of this fairly complex extension a "one click" affair.

Once you've got PostgreSQL/PostGIS installed though, how are upgrades handled? The key is having the right versions in place, at the right time, for the right scenario and knowing a little bit about how PostGIS works.

## Install PostgreSQL 10 and PostGIS 2.4

To exercise an upgrade scenario we install slightly older versions of PostgreSQL and PostGIS.

Starting from a bare Centos 7 box, you'll want to update all your packages and add the [EPEL](https://fedoraproject.org/wiki/EPEL) repository.

sudo bash

# as root

# update all installed software

yum update -y

# add repository for extra packages

yum install -y epel-release

Then install PostgreSQL 10 and PostGIS 2.4 from the PostgreSQL Global Developer Group (pgdg) [yum repository](https://yum.postgresql.org/).

# as root

# install repository package for pgsql 10

rpm -ivh https://download.postgresql.org/pub/repos/yum/10/redhat/rhel-7-x86\_64/pgdg-centos10-10-2.noarch.rpm

# install pgsql 10

yum install -y postgresql10-server postgis24\_10 postgis24\_10-client unzip

Note that installing the PostGIS package brings along quite a large list of dependencies, including:

* geos37 or geos36, a computational geometry library
* proj49 or proj50, a reprojection library
* gdal11 or gdal24, a raster format and processing library
* json-c, a Javascript format parsing library

If you experience problems during an upgrade, it is often due to inconsistencies in these library versions, which is discussed in the "Problems" section below.

Enable the PostgreSQL service, initialize the data area, and start your server.

# as root

# enable service

systemctl enable postgresql-10

# initialize data area

/usr/pgsql-10/bin/postgresql-10-setup initdb

# start server

systemctl start postgresql-10

Just to prove that we're doing a safe upgrade of the spatial components, create a spatially enabled database.

# as postgres

su - postgres

createdb postgis

psql -c 'create extension postgis' -d postgis

Now add a table of spatial data from [Natural Earth](https://www.naturalearthdata.com/) to your database.

# as postgres

wget https://www.naturalearthdata.com/http//www.naturalearthdata.com/download/10m/cultural/ne\_10m\_populated\_places.zip

unzip ne\_10m\_populated\_places.zip

shp2pgsql -s 4326 -I -D ne\_10m\_populated\_places places | psql postgis

So, software installed and data loaded -- we are ready to upgrade!

### Upgrade Scenario #1: PostgreSQL 10 to 11, PostGIS 2.4 to 2.5

Perhaps the most common upgrade path will be a simultaneous upgrade of PostgreSQL and PostGIS to the latest released versions. In our case to PostgreSQL 11 and PostGIS 2.5.

First, we stop the server, then install the latest software.

# as root

# stop the server

systemctl stop postgresql-10

# install repository package for new version

rpm -ivh https://download.postgresql.org/pub/repos/yum/11/redhat/rhel-7-x86\_64/pgdg-centos11-11-2.noarch.rpm

# install new versions

yum install -y postgis25\_11 postgresql11-server postgis25\_11-client

With the server stopped and the new software in place, we are ready to run an in-place upgrade.

# as postgres

su - postgres

# initialize a new data area

/usr/pgsql-11/bin/initdb -D /var/lib/pgsql/11/data/

# run the upgrade

/usr/pgsql-11/bin/pg\_upgrade \

--old-datadir=/var/lib/pgsql/10/data/ \

--old-bindir=/usr/pgsql-10/bin \

--new-datadir=/var/lib/pgsql/11/data/ \

--new-bindir=/usr/pgsql-11/bin

With the data upgrade complete, we can remove the old software.

# check what you have installed

rpm -qa | grep postgis

rpm -qa | grep geos

rpm -qa | grep proj

# remove the old versions of postgis

rpm -e postgis24\_10 postgis24\_10-client

# if you have an older version of geos, remove it now

# so you only have the latest version

rpm -e geos36

Finally, start the new server.

# as root

systemctl start postgresql-11

Update PostGIS extension to new version.

# as postgres

psql -c 'alter extension postgis update' -d postgis

Test the version update.

# as postgres

psql -c 'select postgis\_full\_version()' -d postgis

psql -c "select st\_asgeojson(geom) from places where name = 'New York'" -d postgis