PostGIS 2.0 pgsql2shp shp2pgsql Command Line Cheatsheet PDF Version

POSTGIS 2.0.0 PGSQL2SHP SHP2PGSQL CHEAT SHEET

shp2pgsql and pgsql2shp are all located in the bin folder of the PostgreSQL install.

pgsql2shp dumps a postgis database table, view or sql query to ESRI shape file format.

USAGE: pgsql2shp [OPTIONS] database [schema.]table pgsql2shp [OPTIONS] *database query*

shp2pgsql generates an SQL script from ESRI shape and DBF files suitable for loading into a PostGIS enabled database.

USAGE: shp2pgsql [*OPTIONS*] *shapefile* [schema.]*table*

New in 2.0.0 [1], New in 1.5[2]

General options: (P - pgsql2shp, S - shp2pgsql)

P	-b	Use a binary cursor.
S	-s *from_srid:to_srid*	If -s :to_srid [1] is not specified then from_srid is assumed and no transformation happens.
S	(-d|a|c|p)	These are mutually exclusive options:
S	-d	Drops the table, then recreates it and populates it with current shape file data.
S	-a	Appends shape file into current table, must be exactly the same table schema.
S	-c	Creates a new table and populates it, default if you do not specify any options.
S	-p	Prepare mode, only creates the table.
P	-f *filename*	Use this option to specify the name of the file to create
P S	-g *geometry_column_name*	Specify the name of the geometry column to be (S) created (P) exported.
P	-h *hostname*	Specify db server host name defaults to localhost.
S	-D	Use postgresql dump format (defaults to sql insert statments).
S	-e	Execute each statement individually, do not use a transaction. Not compatible with -D
P S	-k	Keep postgresql identifiers case.
S	-i	Use int4 type for all integer dbf fields.
S	-I	Create a GiST index on the geometry column.
S	-p *port*	Allows you to specify a database port other than the default. Defaults to 5432.
P	-P *password*	Connect to the database with the specified password.
P	-r	Raw mode. Do not unescape attribute names and not skip the 'gid' attribute.
P	-S	Generate simple geometries instead of MULTI geometries.
S	-u *user*	Connect to the database as the specified user.
P	*-w*	Use wkt format (for postgis-0.x support - drops M - drifts coordinates).
S	*-W*	*encoding* The character encoding of Shape's attribute column. (default : "UTF-8")
S	*-N*	
S	*-n*	*policy* Specify NULL geometries handling policy (insert,skip,abort)
S	*-G*[2]	Only import DBF file.
S	*-T*[1]	Use geography type instead of geometry (requires lon/lat data) in WGS84 long lat (-s SRID=4326)
S	*-X*[1]	Specify the tablespace for the new table. Indexes will still use the default tablespace unless the -X parameter is also used.
S		Specify the tablespace for the new index.
P	*-m*[1] *filename*	Remap identifiers to ten character names. The content of the file is lines of two symbols separated by a single white space.
P S	*-?*	Display this help screen

PSQL Connection options:

```
  -h, --host=HOSTNAME      database server host or socket directory
  -p, --port=PORT          database server port number
  -U, --username=NAME      connect as specified database user
  -W, --password           force password prompt (should happen automatically)
  -e, --exit-on-error      exit on error, default is to continue

If no input file name is supplied, then standard input is used.
```

LOADING DATA WITH SHP2PGSQL

Load data into PostgreSQL from ESRI shape file MA stateplane feet

```
shp2pgsql -s 2249 neighborhoods public.neighborhoods > neighborhoods.sql
psql -h myserver -d mydb -U myuser -f neighborhoods.sql
```

Do above in one step

```
shp2pgsql -s 4326 neighborhoods public.neighborhoods | psql -h myserver -d mydb -U myuser
```

Load data into PostgreSQL from ESRI shape file MA stateplane feet to geography

```
shp2pgsql -G -s 2249:4326 neighborhoods public.neighborhoods > neighborhoods_geog.sql
psql -h myserver -d mydb -U myuser -f neighborhoods_geog.sql
```

Sample linux sh script to load tiger 2007 massachusetts edges and landmark points

```
TMPDIR="/gis_data/staging"
STATEDIR="/gis_data/25_MASSACHUSETTS"
STATESCHEMA="ma"
DB="tiger"
USER_NAME="tigeruser"
cd $STATEDIR
#unzip files into temp directory
for z in */*.zip; do unzip -o -d $TMPDIR $z; done
for z in *.zip; do unzip -o -d $TMPDIR $z; done

#prepare the tables don't load data
#force non-multi and set the geometry column name to the_geom_4269, dbf is in latin1 encoding
shp2pgsql -s 4269 -g the_geom_4269 -S -W "latin1" -p fe_2007_25025_edges.shp ${STATESCHEMA}.edges | psql -U $USER_NAME -d $DB
shp2pgsql -s 4269 -g the_geom_4269 -S -W "latin1" -p fe_2007_25025_pointlm.shp ${STATESCHEMA}.pointlm | psql -U $USER_NAME -d $DB

#loop thru pointlm and edges county tables and append to respective ma.pointlm ma.edges tables
for t in pointlm edges;
do
 for z in *${t}.dbf;
  do
        shp2pgsql  -s 4269 -g the_geom_4269 -S -W "latin1" -a $z ${STATE_SCHEMA}.${t} | psql -d $DB -U $USER_NAME;
  done
done
```

OUTPUTING TO ESRI SHAPEFILE/DBF WITH PGSQL2SHP

Export query to a shape file called jpnei.shp/dbf

```
pgsql2shp -f "/path/to/jpnei" -h myserver -u apguser -P apgpassword mygisdb
        "SELECT neigh_name, the_geom FROM neighborhoods WHERE neigh_name = 'Jamaica Plain'"
```

Export a table in ma schema called streets to streets.shp/dbf

```
pgsql2shp -f "/path/to/streets" -h myserver -u apguser -P apgpassword mygisdb ma.streets
```

Boston GIS   Paragon Corporation   Postgres OnLine Journal PostGIS in Action