**Purpose:**

This document describes how to add CSV data to PostGIS located on a remote computer using the Linux operating system, for subsequent use as a spatial layer in GeoServer. Data is often organized as a table, and CSV is a useful exchange format. An alternative is to create the file in shape format, and load that.

The version of Linux used was CentOS, which is a non-graphical interface. Thus, the process should work for virtually all operating systems. The description incudes the process of first passing data to a remote computer. The processes were adapted from descriptions on the web.

| **Step** | **Major Activity** | **References, Forms and Details** |
| --- | --- | --- |
| 1 | Pass the file to the remote computer. In Linux, this is:   * $ scp foobar.txt   [your\_username@remotehost.edu:/some/remote/directory](mailto:your_username@remotehost.edu:/some/remote/directory)  To pass multiple files in a single step:   * $ scp foo.txt bar.txt [your\_username@remotehost.edu:~](mailto:your_username@remotehost.edu:~) | Other file-moving variants are mentioned at:  <http://www.hypexr.org/linux_scp_help.php> |
| 2 | Log onto the remote computer   * ssh –t <username>@remotehost.edu |  |
| 3 | Move to the location of the CSV file(s)   * cd /some/remote/directory |  |
| 4 | Create an empty table in PostGIS   * psql <database <username> * DROP TABLE IF EXISTS <tablename>; * CREATE TABLE <tablename> (   Col\_1\_name <data\_type>,  Col\_2\_name <data\_type>,  . . .  Col\_n\_name <data\_type>  ); | * Note opening and closing parentheses for create table * Common numeric and character types:   + The **real** type typically has a range of at least 1E-37 to 1E+37 with a precision of at least 6 decimal digits.   + The **double precision** type typically has a range of around 1E-307 to 1E+308 with a precision of at least 15 digits.   + The type **integer** is a common choice, as it offers the best balance between range, storage size, and performance.   + The notations **varchar(n)** and **char(n)** are aliases for **character varying(n)** and **character(n),** respectively. **character** without length specifier is equivalent to character(1). |
| 5 | Still in psql, and if the file has a header:  \COPY <tablename> FROM '/path/to/csv/filename.csv'  DELIMITER ',' CSV HEADER; | * If no header, omit the HEADER attribute * Use \copy (and not simply copy) because one does not need special privileges to perform. |
| 6 | Add a numerical column for primary key:  ALTER TABLE <tablename> ADD COLUMN gid serial PRIMARY KEY; | * Creates a binary tree search index |
| 7 | Create a geometry column:  ALTER TABLE <tablename> ADD COLUMN geom geometry(POINT,4326); | * Geometry may be other than POINT. Projection may be other then 4326 (latitude, longitude on datum WGS84) |
| 8 | Update the geometry column:  UPDATE <tablename>  SET geom = ST\_SetSRID(ST\_MakePoint(longitude,latitude),4326); | * This could identify a projected coordinate system, for instance, an UTM projection with northings and eastings. |
| 9 | Create a spatial index:  CREATE INDEX <suitable\_name>\_geom\_idx  ON <tablename> USING GIST(geom); | * The index name is a free choice, but should be meaningful |
| 10 | Test for correctness  \d <tablename> | * Should see serial PKey and GIST index, as well as columns |