

Installation & Maintenance Guide

GenomePro 3.0 is a web application for processing genomic data files. The application runs online and can be addressed by `http://genomepro.cis.fiu.edu`. The site runs on Apache2 and Postgres using PHP, HTML, and CSS to display client side information. ANSI C programs underly GenomePro's tools. The following are brief, yet descriptive, steps detailing how to install the application.

Requirements

1. LINUX/UNIX Operating System
2. Cron support
3. Apache2 Web Service*
4. Postgres Database*

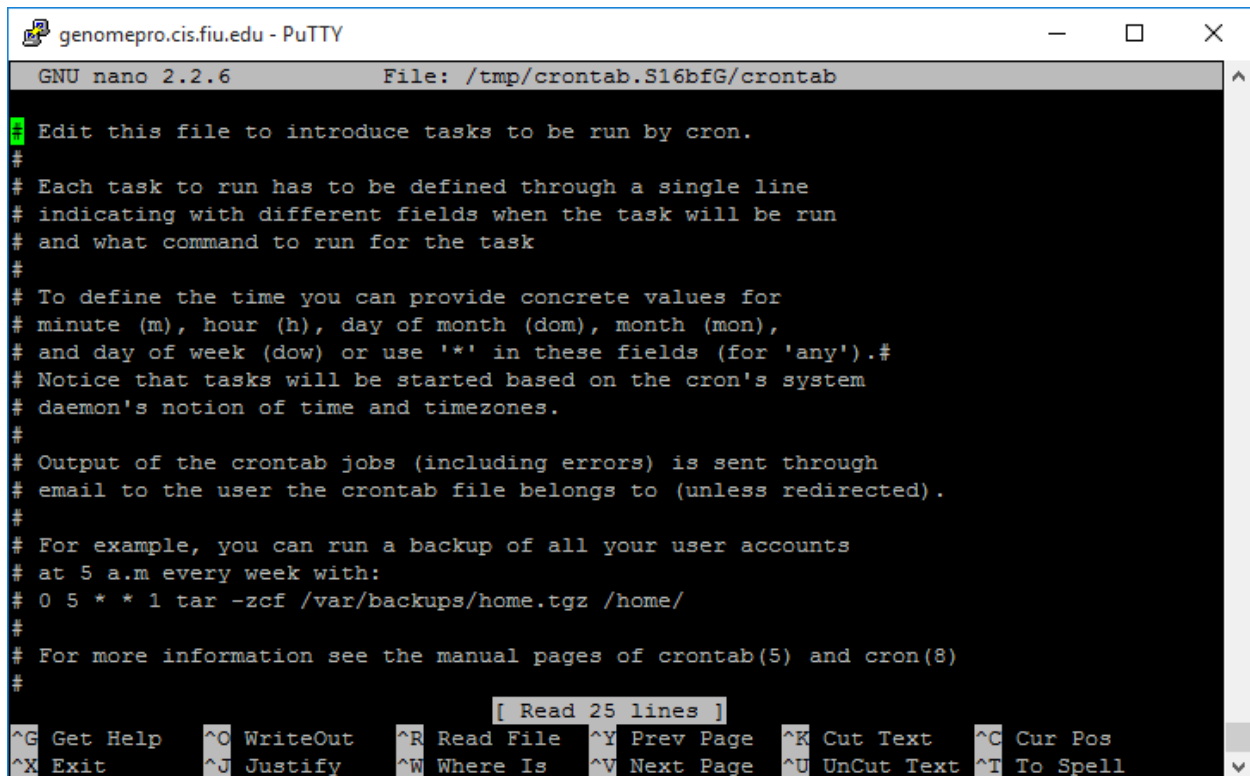
** Tutorials on this can be found online or below at the appendix*

Step 1 – Moving the Website

Save the project root directory and contents to a new directory where you would like to host the site (usually `/var/www/html/`, but this is platform-dependent). The site should immediately be accessible at the address `http://localhost/` hosted on the local machine. Please make sure not to rename anything! If you get a 404 error then Apache2 was not installed correctly or the file permissions of the copied directory are not correctly set. A quick fix is to `chmod -R 755` recursively at the installation directory, though you may want to differentiate between file permissions and directory permissions with `644`.

Step 2 – Installing CRON Jobs

The server has three different CRON jobs that must be installed on the system. Execute command `sudo crontab -e` on the terminal to do this. Enter your password if asked. A dialogue similar to this should appear if everything worked well:



```
genomepro.cis.fiu.edu - PuTTY
GNU nano 2.2.6 File: /tmp/crontab.S16bfG/crontab

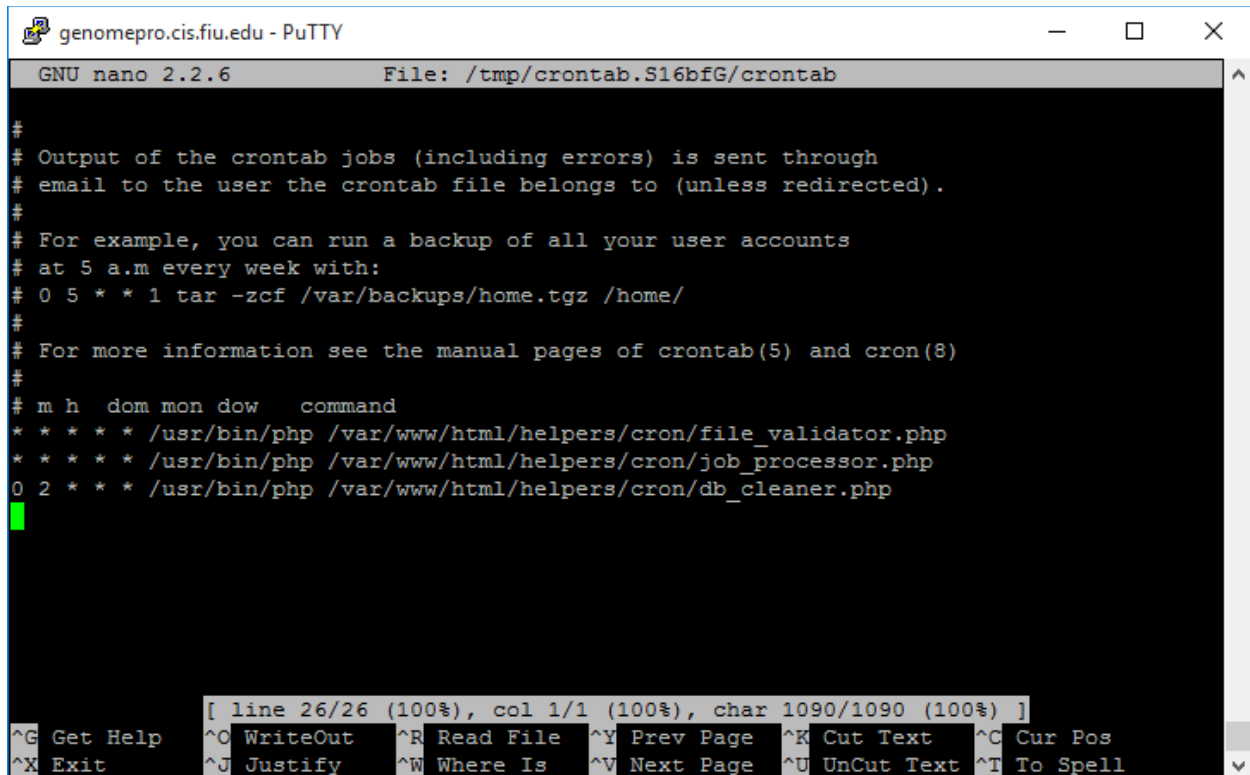
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#

[ Read 25 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text ^T To Spell
```

At the end of the file, add the following lines:

```
* * * * * /usr/bin/php /var/www/html/helpers/cron/file_validator.php
* * * * * /usr/bin/php /var/www/html/helpers/cron/job_processor.php
0 2 * * * /usr/bin/php /var/www/html/helpers/cron/db_cleaner.php
```

Make sure that the lines are changed depending on your system. The second argument, `/usr/bin/php`, is where PHP is installed (note that this location is platform-dependent). The scripts are located in `/var/www/html/helpers/cron/file_validator.php`. Please note that the arguments are absolute file paths and not relative. The GenomePro server crontab looks like this:



The screenshot shows a terminal window titled "genomepro.cis.fiu.edu - PuTTY". Inside, the GNU nano 2.2.6 editor is open, editing the file `/tmp/crontab.S16bfG/crontab`. The crontab file content is as follows:

```
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
* * * * * /usr/bin/php /var/www/html/helpers/cron/file_validator.php
* * * * * /usr/bin/php /var/www/html/helpers/cron/job_processor.php
0 2 * * * /usr/bin/php /var/www/html/helpers/cron/db_cleaner.php
```

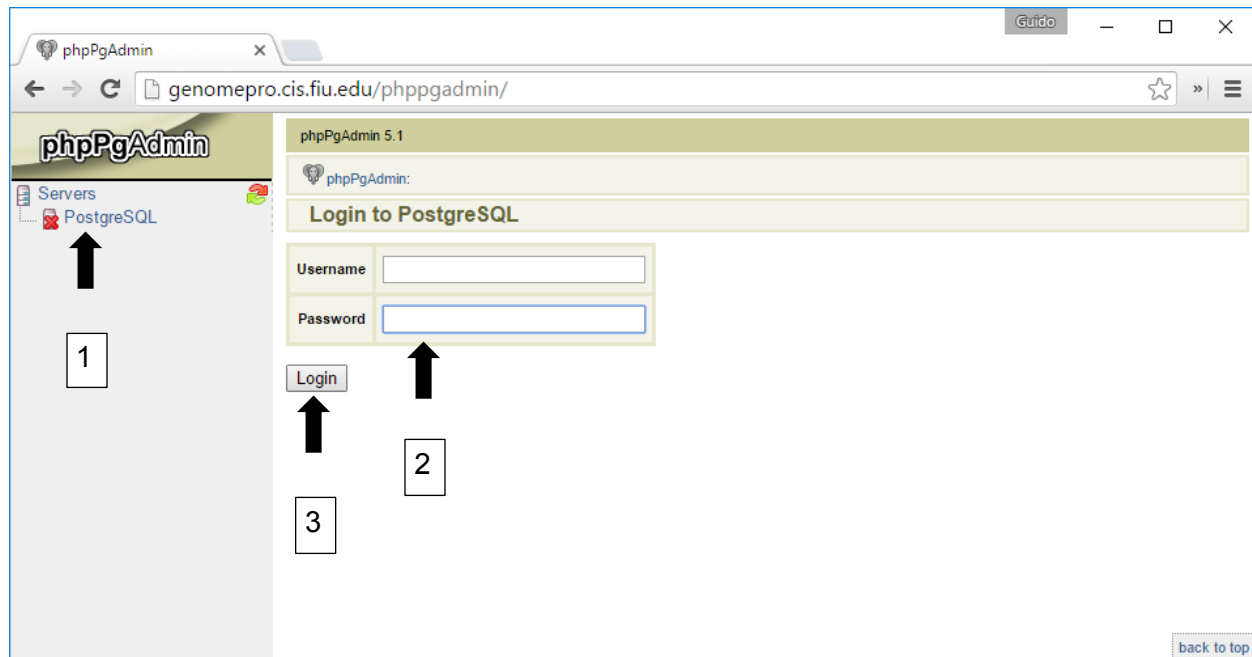
The bottom of the terminal shows the nano editor's status bar: `[ line 26/26 (100%), col 1/1 (100%), char 1090/1090 (100%) ]`. Below the status bar is a table of keyboard shortcuts:

<code>^G</code> Get Help	<code>^O</code> WriteOut	<code>^R</code> Read File	<code>^Y</code> Prev Page	<code>^K</code> Cut Text	<code>^C</code> Cur Pos
<code>^X</code> Exit	<code>^J</code> Justify	<code>^W</code> Where Is	<code>^V</code> Next Page	<code>^U</code> UnCut Text	<code>^T</code> To Spell

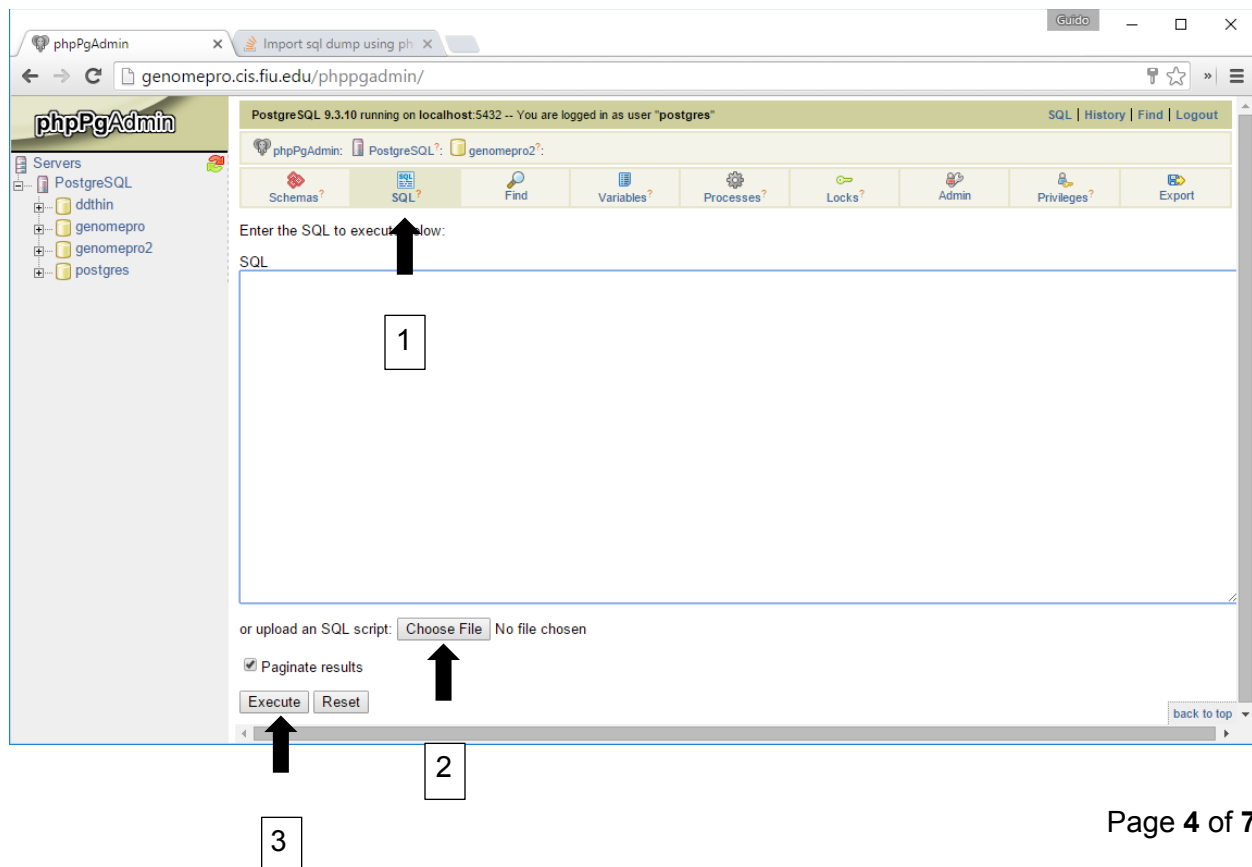
Step 3 – Creating the Database

The GenomePro server can work with any Postgres database that has the same structure as the previous one. There are SQL dump files that you can use to convert your Postgres database to GenomePro's structure. They can be found in `/Code/Database`.

Visit `localhost/phppgadmin` or `[your_site_domain]/phppgadmin`, depending on if you're on a local or remote connection, and log in by clicking 'PostgreSQL' on the left hand side. Use the credentials provided during installation or the default ones. A visual explanation can be found below:

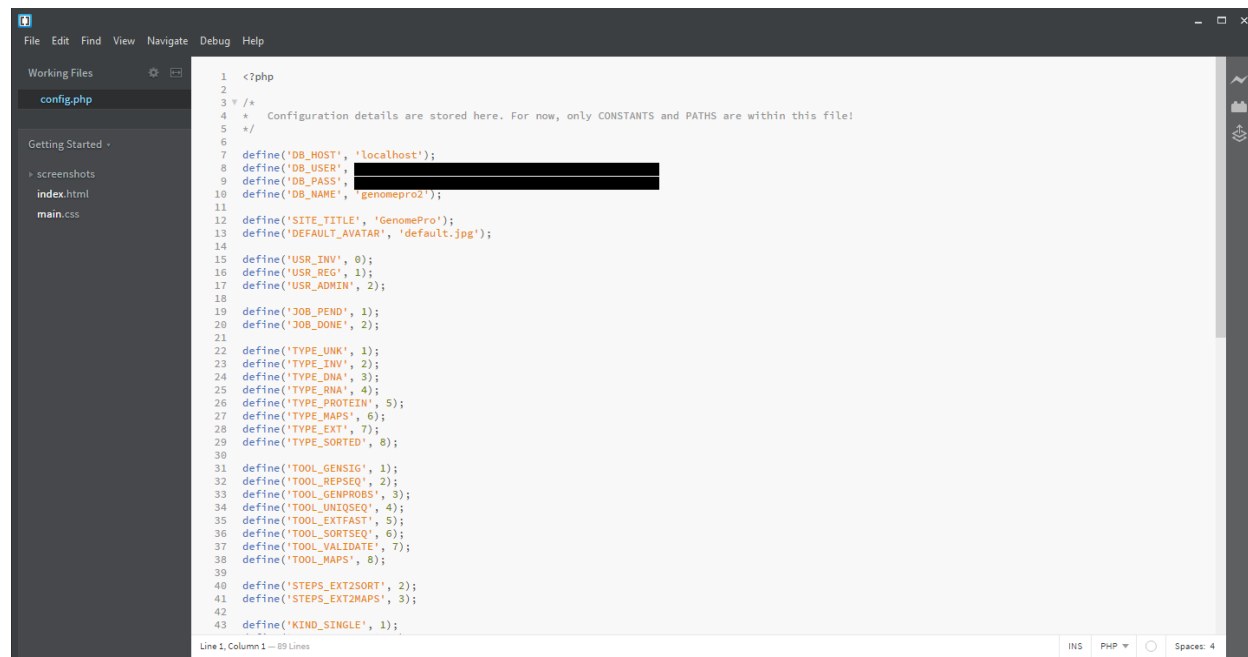


Once you are logged in, create a new database, click on "SQL" in the middle and load your SQL script to create the database as well as the tables required for GenomePro. It is highly suggested that you don't change these, as parts of the site may not function as expected (or at all).



Step 4 – Editing Config File

The last step required so that GenomePro successfully works on your server is configuring the config file found in [root]/config/config.php. This file must be updated with new values including, but not limited to, the database's new name, username, password, etc. The changes should be minimal. Here's an example config file:



```
1 <?php
2
3 /*
4  * Configuration details are stored here. For now, only CONSTANTS and PATHS are within this file!
5  */
6
7 define('DB_HOST', 'localhost');
8 define('DB_USER', 'root');
9 define('DB_PASS', 'root');
10 define('DB_NAME', 'genomepro2');
11
12 define('SITE_TITLE', 'GenomePro');
13 define('DEFAULT_AVATAR', 'default.jpg');
14
15 define('USR_INV', 0);
16 define('USR_REG', 1);
17 define('USR_ADMIN', 2);
18
19 define('JOB_PEND', 1);
20 define('JOB_DONE', 2);
21
22 define('TYPE_UNK', 1);
23 define('TYPE_INV', 2);
24 define('TYPE_DNA', 3);
25 define('TYPE_RNA', 4);
26 define('TYPE_PROTEIN', 5);
27 define('TYPE_MAPS', 6);
28 define('TYPE_EXT', 7);
29 define('TYPE_SORTED', 8);
30
31 define('TOOL_GENSIG', 1);
32 define('TOOL_REPSEQ', 2);
33 define('TOOL_GENPROBS', 3);
34 define('TOOL_UNIQUEQ', 4);
35 define('TOOL_EXTFAST', 5);
36 define('TOOL_SORTSEQ', 6);
37 define('TOOL_VALIDATE', 7);
38 define('TOOL_MAPS', 8);
39
40 define('STEPS_EXT2SORT', 2);
41 define('STEPS_EXT2MAPS', 3);
42
43 define('KIND_SINGLE', 1);
```

Appendix

Below you can find some quick guides on how to install Apache2 and Postgres courtesy of the GemomePro 2.0 Team led by Guido Ruiz and Mardoqueu Mesquita. We hope that the below tutorials can be of assistance and suggest that you verify the instructions below for version differences. You may have to adjust the instructions for your platform (e.g. to use the correct package manager for your distribution of Linux). Here, we assume installation will occur on an Ubuntu-like distro running apt-get.

Install PostgreSQL 9.4

1. Install PostgreSQL
 - o `sudo apt-get install postgresql postgresql-contrib`
2. Access PostgreSQL command prompt
 - o `sudo -u postgres psql postgres`
3. Set "postgres" user password
 - o `postgres=# \password postgres`
4. Enter your new password (twice):
 - o `postgres=# \q`

Install phpPgAdmin

1. Install phpPgAdmin:
 - o `sudo apt-get install phppgadmin`
2. By default, you can access phppgadmin using:
 - o `http://localhost/phppgadmin`

Access Remote phpPgAdmin

1. Edit file `/etc/apache2/conf.d/phppgadmin` with your text editor of choice:
 - o `sudo vim /etc/apache2/conf.d/phppgadmin`
2. Comment out the following line:
 - o `#allow from 127.0.0.0/255.0.0.0 ::1/128`
3. Uncomment the following line to make phppgadmin from all systems:
 - o `allow from all`
4. Edit `/etc/apache2/apache2.conf`:
 - o `sudo vim /etc/apache2/apache2.conf`
5. Add the following line:
 - o `include /etc/apache2/conf.d/phppgadmin`
6. Then, restart apache service.
 - o `sudo service apache2 restart`

Configure phpPgAdmin

1. Edit file /etc/phppgadmin/config.inc.php
 - o `sudo vim /etc/phppgadmin/config.inc.php`
2. Find the following line:
 - o `$conf['servers'][0]['host'] = '';`
3. Modify it as follows:
 - o `$conf['servers'][0]['host'] = 'localhost';`
4. Then, find the line:
 - o `$conf['extra_login_security'] = true;`
5. And change the value to false:
 - o `$conf['extra_login_security'] = false;`
6. Find the line:
 - o `$conf['owned_only'] = false;`
7. Set the value to true:
 - o `$conf['owned_only'] = true;`

Restart PostgreSQL service and Apache services

1. Save and close the file.
 - o `sudo service postgresql restart`
 - o `sudo service apache2 restart`
2. Important Link.
 - o <http://www.unixmen.com/install-postgresql-9-4-phppgadmin-ubuntu-14-10/>