

PIBASE installation guide. ver 200808



Fred P. Davis, HHMI-JFRC
davisf@janelia.hhmi.org
<http://pibase.janelia.org>

August 27, 2008

Abstract

This document describes how to set up a local PIBASE installation.

PIBASE can be installed locally by (1) downloading a compiled version of the database from <http://pibase.janelia.org/download.html>, or (2) downloading the PIBASE software package and compiling your own version of the database. Both routes lead to a complete installation that can be accessed through the mysql, perl, or web interfaces.

Part I

Installing the precompiled PIBASE v200808

1 Downloading data

All PIBASE files are available for download at <http://pibase.janelia.org/download.html> under the GPL license.

1.1 PIBASE software

Download the PIBASE software package (`pibase_src.20080825.tar.gz`) and uncompress it in the directory where you want to install the software:

```
cd yourinstalldirectory
tar xvfz pibase_src.20080825.tar.gz
```

1.2 MySQL data dump

Create an empty MySQL database to hold the PIBASE contents, for example:

```
mysql -u root -p
mysql> CREATE DATABASE pibase ;
mysql> GRANT ALL ON pibase.* TO pibaseuser@hostname IDENTIFIED BY pibasepassword;
mysql> FLUSH PRIVILEGES;
```

Download the PIBASE mysql dump (`pibase_dump.20080825.out.gz`) and load it into the database you just created:

```
zcat pibase_dump.20080825.out.gz | mysql -u YOURUSERNAME -p YOURDATABASENAME
```

1.3 Data files

Besides the data in the MySQL tables, there are several kinds of compressed text files that are space-prohibitive as MySQL tables. For example the actual structure files for individual domains, or the list of pairwise contacts at interfaces. The files you need to download depend on what type of queries or data you are interested in accessing. For example, for a webserver installation, only the `bdp_topology_graphs` and `subset_files` are required. If you are only interested in querying properties that are present in the MySQL tables, you don't need any of these at all.

- `bdp_topology_graphs` (240MB) - png and eps pictures of complex topology graphs.
- `subset_files` (18GB) - PDB files of individual domains.
- `metatod` - meta tables on disk; the *_file MySQL tables point to these files.
 - `bdp_residues` (1.3GB)- residue listing for each BDP file.
 - `bdp_sec_strx` (0.85GB)- secondary structure assignment for each BDP file.
 - `bindingsite_contacts` (2.5GB) - contacts between residues in each binding site
 - `bindingsite_secstrx_basic_contacts` (0.64GB) - contacts between secondary structure elements in each binding site
 - `interface_contacts` (1.1GB)- pairwise contacts across each interface
 - `interface_contacts_special` (0.62GB) - "special" contacts (H-bond, salt bridge, disulfide bridges) across each interface
 - `interface_secstrx` (1.5GB) - secondary structure content at each interface
 - `interface_secstrx_basic_contacts` (0.47GB) - contacts between basic secondary structure elements at each interface
 - `interface_secstrx_contacts` (0.47GB) - contacts between detailed secondary structure elements at each interface
 - `patch_residues` (0.67GB) - residues in each patch (contiguous region of residues in each binding site)
 - `subsets_residues` (0.85GB) - residues in each domain

2 Web interface

Setting up a own web interface to PIBASE requires that you have the (1) software downloaded, the (2) MySQL database created, and (3) the data files for `bdp_topology_graphs` and `subsets_files` downloaded and uncompressed somewhere. Next, there are a few lines to edit in `pibase.pm` to reflect your database specifications and local directory structure. Then, just copy over the html, cgi-bin, and perl library to your webserver, and it should be ready to query.

In detail:

1. Edit `pibase.pm` to reflect your MySQL database speccs. (`src/perl_api/pibase.pm` lines 57-60).

```
my $pibase_specs = {  
    db => 'pibasemysqldatabasename',  
    host => 'mysqlserverhostname',  
    user => 'pibaseusername' ,  
    pass => 'pibasepassword',  
    root => 'doesntmatterforthewebserver',  
}
```

2. Edit `pibase.pm` to reflect the html and cgi-bin directories of your web server. (`src/perl_api/pibase.pm` lines 203, 204).

```
$pibase_specs->{web}->{html_dir} = "/var/www/websites/pibase/html/" ;  
$pibase_specs->{web}->{cgi_dir} = "/var/www/websites/pibase/cgi-bin/" ;
```

3. Edit `pibase.pm` to reflect the correct URLs for the html and cgi-bin directories of your web server. (`src/perl_api/pibase.pm` lines 206, 207).

```
$pibase_specs->{web}->{base_url} = "http://localhost/pibase";  
$pibase_specs->{web}->{basecgi_url} = "http://localhost/pibase-cgi";
```

4. Edit `pibase.pm` to reflect the directories where you uncompressed the `bdp_topology_graphs` and `subsets_files` tar files. (`src/perl_api/pibase.pm` lines 209, 213).

```
$pibase_specs->{web}->{bdp_topology_graphs_baseurl} =  
    $pibase_specs->{web}->{base_url} . '/data_files/bdp_topology_graphs'  
  
$pibase_specs->{web}->{subsets_files_basedir} =  
    $pibase_specs->{web}->{html_dir} . '/data_files/subsets_files' ;
```

5. The static html pages (`web/html/*.html`) assume that the CGI directory sits `../cgi-bin` relative to the html directory; if this isn't true for your webserver, edit the form submit lines in the html pages to reflect the path to your CGI directory
6. Lastly, copy the contents of the html, cgi, and perl_api directories to your webserver

```
cp -r web/html/* yourwebserver/html
cp -r web/cgi-bin/* yourwebserver/cgi-bin
cp -r src/perl_api yourwebserver/cgi-bin/perl_lib
```

That should be it, the webserver should be functional now.

Part II

Building your own PIBASE

The process is more simple than it appears below. First, edit `pibase.pm` to specify the file paths for your local directory structure. Then, `perl src/scripts/build_pibase.pl` should run through the build process in the correct order.

The documentation in this section is not yet complete. This will improve as time allows.

3 Building *my own*. started 080229

- set up local PDB and PQS mirrors

Update/start local pdb mirror

```
rsync -a rsync.ebi.ac.uk::pub/databases/rcsb/pdb-remediated/data/structures/divided/pdb/ .
```

start local pqs mirror (eta 4-5 hr on (default) 20 puf connections) had to split the url list into 100 and run through them serially, otherwise puf was hanging

```
perl ../src/scripts/build_pibase.pl -mirror_pqs 1 > mirror_pqs.out
grep ^http mirror_pqs.out > pqs_urls.out
grep ^mv mirror_pqs.out > pqs_directory_layout.out
~/software/scripts/setup_clusterrun.pl pqs_urls.out tasklist
cd $PQS_DIR
ls split.pqs_urls.out.a* | awk '{print "/groups/karpova/home/davisf/software/puf/puf-1.0.0"}'
bash puf_downloads.sh
bash pqs_directory_layout.out
```

check integrity of downloaded pqs files

```
perl ../src/scripts/build_pibase.pl -check_pqs_files 1 > check_pqs_files.out
grep -v "\tcomplete$" check_pqs_files.out | awk '{print $1}' |xargs -i{} grep {} puf_downl
```

manually follow up the ones that aren't marked as complete

```
cat check_pqs_files.out | grep -v 'complete'
CHECKING EXTERNAL DATA (pibase::build::fetch_external_data) Sun Mar 2 15:51:24 2008
* astral files:
  * scopseq-1.73.tgz: x
```

```

* cath files:
  * CathDomall.v3.1.0: x
  * CathNames.v3.1.0: x
  * CathDomainDescriptionFile.v3.1.0: x
  * CathDomainList.v3.1.0: x
  * CathChainList.v3.1.0: x
* pdb files:
  * obsolete.dat: x
  * entries.idx: x
* pqs files:
  * ASALIST: x
  * RANKING: x
  * BIOLIST: x
  * LIST: x
* scop files:
  * dir.cla.scop.txt_1.73: x
  * dir.hie.scop.txt_1.73: x
  * dir.des.scop.txt_1.73: x
1fhp_1.mmol      not found
1htz_1.mmol      incomplete
1htz_2.mmol      incomplete
1htz_3.mmol      incomplete
1htz_4.mmol      incomplete
1htz_5.mmol      incomplete
1htz_6.mmol      incomplete
1iw7_1.mmol      incomplete
1iw7_2.mmol      incomplete
1nmc_1.mmol      incomplete
1ruf_1.mmol      incomplete
1ruf.mmol        incomplete

```

manually check these - some of them ok: 1htz_* ok, 1iw7_* ok, 1nmc_1 ok only bad ones: 1fhp_1
 obsoleted, 1ruf_1 and 1ruf - messed up on the PQS site itself.

remove these entries from the PQS directory, and from the PQS LIST file itself

PQS warnings

- fetch and process external data

```
080229_1823: perl ../src/scripts/fetch_external_data.pl > fetch_external_data.out
```

```
080301_1056: perl ../src/scripts/build_pibase.pl -process_external_data > process_external_data.out
```

ran again to get PDB and PQS info files.

```
080301_1611: perl ../src/scripts/build_pibase.pl -fetch_external_data > fetch_external_data.out
```

- extract model #1 from all PDB NMR structures

NMR model 1 extractor - runs online regular pibase module, requires database structure

```
080305_1243: perl ../src/scripts/build_pibase.pl -extract_nmr_model1 1 > extract_nmr_model1
```

- generate internal structure list

```
080305_1729: perl ../src/scripts/build_pibase.pl -maketable_bdp_files 1 > maketable_bdp_files
```

- calculate residue info

```
davisf@tangerine manual_build> perl ../src/scripts/build_pibase.pl -call_residue_info 1 >
submitted pb.call_residue_info.ONJNM.SGE.sh job 9967279
```

don't know why, but bdp_residues_tables also had an entry with bdp_id of 0 - deleted it manually.
- must have been some stuff that was there from before, reran the pibase::mysqlimport line and loaded properly this time.

```
delete from bdp_residues_tables where bdp_id = '0' ;
```

- calculate chain info

```
davisf@tangerine manual_build> perl ../src/scripts/build_pibase.pl -call_chain_info 1 > ca
```

```
davisf@tangerine manual_build> perl ../src/scripts/build_pibase.pl -calc_bdp_chains_equiv
```

- INCOMPLETE - moved over to build_pibase.pl exclusively, order specified in pibase::build.pm