PQLX Database Setup & Preparation

Introduction

Before creating a new PQLX Server Database, some preparatory actions must be carried out:

- Define which channels are to be contained in a single PQLX Database
- Define the location of all data to be analyzed per PQLX Database

There is no limit to the number of PQLX databases that may be created, hence, consider the most logical way that data should be segregated. For example, one database for a real-time network, another for new instrument testing, another for archived data, etc.

Data Preparation

The PQLX system is driven off the existence of data. For each PQLX database instance, one or more directories are required to exist, containing a mix of either **trace files and response files** for the channels whose analysis is to be held by the database.

Decide for each PQLX database those channels (or stations) whose data should be grouped together in a single database. Subsequent client-side queries will be limited to comparisons of data contained within this single database during any given client connection with the server.

There are no requirements as to filenames or directory structures, configuration thereof is left entirely up to the system administrator. All trace and response files may be located under a single directory structure or may be segregated between numerous directories and/or sub-directories, how this is managed is completely left to the implementor.

However, experience suggests that ultimately the most user-friendly configuration is one where individual directories are maintained for response files and trace data files. Both of these directories being separate directories to be scanned by the PQLX server process. In this manner, it is possible to maintain a single directory of response files that may be used across multiple PQLX database instances.

Waveform Data File Formats

The following list defines the seismological file formats supported and auto-detected by PQLX. Any PQLX data directory for any given data directory may contain any combination of the following seismological trace formats:

- mini-SEED
- SAC
- SEGY
- AH
- DR100
- NANO

Response File Format

Response files must be provided in a single format, namely, the format produced by the program rdseed, (also available via direct download from the IRIS website). (N.B. This format is used for all trace file formats, not just mini-SEED. For non mini-SEED trace file

formats, modify a mini-SEED response file, providing the appropriate values for the channel in question.)

Response file requirements are:

1. Each response file must contain information for only a single channel.
2. Each response file must contain response information over the entire lifetime of the instrument. That is, a single channel may not use multiple response file instances to represent responses for different time periods.

If after executing the server, no PSD data is generated, the response file is a likely culprit. Check the errors on the EVENTQ (via helper scripts listQ.sh, pqlx-ErrorDetail.sh, and pqlx-ErrorSummary.sh), as well as the server error logs. As well, confirm that the response file adheres to the format described here and start again.

Data Set Preparation

Before creating and initializing a PQLX database, then, the following decisions must be made:

1. Decide how data should be segregated by database, i.e., decide which channels should be contained within a single database instance.
2. For each PQLX database, set up one or more directories containing **all** PQLX-related files per database, trace files and response files. There are no hierarchy or naming requirements for this directory or the files it holds, all directories and files may be named however desired, though planning a logical hierarchy may save headaches in the future.

   It is recommended that a separate directory be created to hold all Response Files for all channels; this will make maintenance easier.

MySQL/PQLX Environment Initialization

If the PQLX environment was automatically initialized during installation, this section may be skipped. (You can check if the system has already been initialized by executing the init script below. When the system has already been initialized, the script will refuse to execute.)

If system initialization was deferred, no aspect of PQLX may be executed without this initialization.

The PQLX system is initialized with the following command, (located in $PQLXBIN):

```
bash> initPQLXdb.sh
```

This initialization step is required only once per PQLX server instance and is responsible for creating the PQLX database users as well as the PQLX Meta database containing the definitions of all PQLX databases held on the server.

This initialization script will request the MYSQL user 'root' password, necessary for PQLX system initialization. This is the only time that the MYSQL root password is required. It is

not saved or ever used again.

PQLX Database Creation

Use the PQLX client program 'pqlx-admin' to create a PQLX database; see the document '**PQLX-3 DB Administration.pdf**' for more details.

Alternatively, it is possible to create a PQLX database manually outside 'pqlx-admin'. Previously, before the admin tool existed, PQLX DB management was based on the existence of a PQLX DB definition file, used in conjunction with various PQLX scripts.

The following describes this process for those cases when the Administration tool is not possible. However, the DB definition directory is now marked read-only for non-root users. Thus, creation of the database definition file must be done with the 'sudo' command.

To seed and initialize a PQLX database instance, the following actions must be carried out:

1. Define a database definition file (located in directory $PQLXDBDEF) for each PQLX/PDF database instance desired. The name of this file must adhere to the following file naming convention:

   dbName.pqlx

   where

   dbName is the name of the PDF Server Database, and

   .pqlx is as is

   (See an example of this file contained in **$PQLXDBDEF/samplePQLXdb.pqlx**. See file **$PQLXDBDEF/PQLXdbTemplate.pqlx** for a template of this file.)

   Edit the DB definition file and provide all required information:

   - Database Description
   - Organization Providing Database
   - Technical Administrator Name
   - Technical Administrator Email Address
   - Data Administrator Name
   - Data Administrator Email Address
   - Database Type (either PRODUCTION (i.e., permanent) or SCRATCH (i.e., temporary))
   - Database Access (either PUBLIC (all clients able to read contents) or PRIVATE (only the database creator is able to read contents)
   - System Administrator Password
   - WWW Directory holding PQLX system-defined PDFs (or NONE to produce no WWW plots)
   - Define which PDF statistics should be displayed when creating PDF PNG plots.
   - Define the date format to use when creating PDF PNG plots.

   All fields are required, any field missing or containing invalid values will result in PQLX database creation failure.

2. Create the PDF Database using the script **$PQLXBIN/makePQLXdb.sh**, specifying all directories defined in Data Preparation Section, Item 2 above (execute with no arguments for Usage help).

Once a PQLX Database instance has been created, the PQLX server is ready for

execution. See server documentation **PQLX-Server.pdf** for details.

Other Database Helper Scripts

delPQLXdb.sh – To delete a previously created PQLX database.

updatePQLXdb.sh – To re-read the contents of the PQLX database definition file (created in step 1, section **PQLX Database Initialization** above), and update the database itself with the new information. Settings having an effect on server output will take effect the next time the server is executed.

PQLXDataDirectory.sh – Allows a PQLX administrator to maintain the data directories defined for a particular database. Actions include: **listing** all data directories currently defined, **adding** an additional data directory, and **enabling** and **disabling** the reading of a data directory. Execute without arguments for complete details. N.B. This can only be executed on the server machine itself.

listPSDchannels.sh – Provides a list of all PSD channels currently defined for a database.