PQLX Server Analysis Program

Introduction

Once a PQLX database has been instantiated, the server is ready for execution.

The PQLX server process is executed to act on behalf of a single PQLX database instance and is responsible for carrying out the following actions:

- Inspect all files in the data directories defined as part of the DB definition, identifying all new files since last execution (both trace files and response files);
- Analyze all new files, writing all analysis statistics back to the PQLX database. The current analysis statistics being computed are:
    - **TRACE Statistics** - trace file characteristics: Start/Stop Times, number of samples, MIN, MAX, and MEAN amplitudes, and, for mini-SEED format files only, number and location of Gaps and Overlaps.
    - **PSD Statistics** - computed according to algorithm laid out by D. McNamara and R. Buland (see discussion document as well as other references here), used for subsequent client-side PSD and PDF display.
    - Additional global statistics for each channel over all time are also maintained.

Execution may be carried out in one of several ways:

- via client-side program 'pqlx-admin', see document 'PQLX-3 DB Administration.pdf' for complete details
- via direct execution from the command line:

  bash> `$PQLXBIN/pqlxSrvr --dbName=`*`[srvrHost:]PQLXdb-name`* `--numCPU=`*`numCPUs`*

  where:
    - ***[srvrHost:]*** is the name of the server machine hosting the PQLX database, i.e., where the MySQL database server is running. No specification defaults to ***localhost,*** i.e., the MySQL database server and the pqlxSrvr are executed to run on the same machine.
    - ***PQLXdb-name*** is the PQLX database name to execute against.
    - ***numCPUs*** is the number of CPUs the server should execute on.

The server program guarantees that the server will complete, automatically restarting itself if it crashes during the course of its work. If a crash occurs, a message will be output indicating this happened as well as the filename containing a list of the traces the server was processing when the crash occurred. In this manner, follow-up investigation is easy. As well, the list of trace files causing the crash may be used for any further script processing, e.g., to easily move all traces to a “side” directory for later investigation and/or to remove them from the data directory of valid or processable trace files. (On the other hand, once a file has been processed, it will not be re-processed unless the time-stamp on the file changes.)

The **pqlxSrvr** program is event-driven, reading from the database the next action it is to carry out. The general order of analysis operations is:

1. Traverse all PQLX-defined data directories identifying all new instances of data files and inserting this identity information into the database.

2. Fork ***numCPUs*** event processors, performing all currently defined data analyses on the newly identified trace files, storing all results back to the database.

3. Traverse again all PQLX-defined data directories identifying all new files since completion of step 1.

   This re-traversal is done so that the server can “catch up” to the current date of available/defined data files. Initial execution of the server (depending on the amount of data, speed of executing machine, number of used CPUs, etc.) may require days to complete. In the meantime, new data files may have come to exist which now require processing. This guarantees, then, that once the server completes the database will accurately reflect all currently existing data.

   In addition, any trace files receiving real-time data transmission will be identified only one time per execution of the server itself on the same day. That is, they will not be “re-identified” as part of this step, thus avoiding a rather inconvenient infinite loop.

4. Loop over steps 2 and 3 until all identified files have been analyzed and no new files have been found.

5. Process any necessary deletions of PSD data. PSD data is required to be deleted when a trace file has been replaced or re-identified during the file identification phase.

6. Update channel and directory statistics held on the database.

7. Update all System PDF plots for channels having had PSD calculations performed in step 2.

   Currently, the following so-named channels are automatically tagged/identified to have PSD calculations performed: BH*, LH*, HH*, EP*, BL*, HG*, HL*, BG*, BN*, SH*, EH*. Other channels can be added to this list very simply by adding an entry to the table PSDCFG. Insert a new entry into the table providing all definition information as for other entries.

8. If PNG output of the System PDF's is specified (see PQLX DB Administration document for details) , all updated system PDF's are output to the defined directory as PNG plots.

pqlxSrvr_safe & Cron Execution

Setting up a cron job to execute the server is relatively straightforward. The execution of the server script **pqlxSrvr_safe** requires the environment variables defined in **$PQLXPROD/PQLXprodVars** to exist for execution (see the PQLX-Installation document for details on setting up a production-only environment).

A sample shell script that can be used as a template for a cron job is provided in $PQLXBIN/pqlxCron.sh. Edit this file and copy to a separate location (subsequent upgrades to the system will overwrite any previously edited version) for execution by cron.

Additionally, this script may contain any other commands required as part of an automatic execution of the PQLX server. For example, if data should be extracted from some data source before the PQLX server is executed, this could be provided within the same script. In such a way, if there ever exist dependencies whereby actions must be carried out in a specific order, these can be easily met by simply executing these actions serially within the same script.

For a daily update of a PQLX database, it is recommended that the server be executed locally sometime after 00:00 GMT. In order to minimally guarantee that all trace files from yesterday are fully processed.

pqlxSrvr – Direct Execution

As noted, using the script **pqlxSrvr_safe** is intended for automatic production environment purposes and normal executions of the server. It is possible, however, to execute the server analysis program directly. This can be done in the case when wanting to re-analyze data files previously analyzed. If, for example, response file information has changed for data in the past which has already been analyzed (or, said otherwise: the PSD data is incorrect due to incorrect response information previously provided), an optional argument to the **pqlxSrvr** program allows these files to be defined for re-analysis.

Executing the **pqlxSrvr** program directly takes the following arguments (execute with no arguments for usage details):

```
--dbName=[SERVER:]pqlxDB  PQLX Database Name - required
--numCPU=#                Number of CPUs to use - optional (default=1)
--identFile=path-to-file  Filename Listing Traces to be Re-Analyzed - optional
--noScan                  Do Not Scan Data Directories - optional
--TRStart=HR:MN           Time-Restricted Execution - Start Time - optional
--TREND=HR:MN             Time-Restricted Execution - End Time - optional
```

Providing a file of traces via the **–identFile** argument will force the server to re-analyze only the trace files listed in this file, replacing all previous analysis information. The format of this file is one trace file (full pathname) per line. N.B. No scanning of the data directories occurs when executing using the --identFile option. Once the re-analysis is complete, all channel statistics are updated and system PDF's recomputed.

Log file Output

The following log files are created as part of the execution of **pqlxSrvr**:

1. **srvr.*dbName.YYYY.JJJ.HH:MM.*log** – standard log file providing information related to the normal course of execution.
2. **srvr.*dbName.YYYY.JJJ.HH:MM.*err** – error file holding any error messages generated during the course of execution. If pqlxSrvr is executed via pqlxSrvr_safe, this file is removed if empty once the server completes.
3. **srvr.*dbName.YYYY.JJJ.HH:MM.*crash** – file containing, one per line, the full pathnames of all trace files which caused the server to crash in the course of its operations. For problem data, this allows for easy follow-up investigation. If pqlxSrvr is executed via pqlxSrvr_safe, this file is removed if empty once the server completes.

It is recommended that these files be variously consulted to ensure that system operation

is running smoothly and as expected/required.

EVENTQ Monitoring and Errors

In order to monitor the state of the server, as well as determine the nature of errors encountered during normal processing, a helper script **listQ.sh** is provided and is described here.

listQ.sh – Outputs EVENTQ table information currently held by the database. The EVENTQ table contains all data processing events processed by the server. Examination of this table is very helpful, if not crucial, in ascertaining the origins of various problems that can arise. This script outputs information of two sorts: current status of the EVENTQ table itself (useful when wanting to know where the server is in the course of its processing), and listing of the errors encountered by the server.

Option 1: **listQ.sh *dbName* ALL** - will output all events currently existing on the EVENTQ.

Possible events include:

- Trace file meta-data analysis,
- Trace file PSD Analysis
- Overlapping Trace file PSD Analysis
- PSD Data Table – index creation
- PSD Data Table – delete PSD's
- System PDF – creation and statistics
- PDF PNG – creation of PNG files

Possible STATUS values:

- INPROC – event is currently being processed by the server
- UNPROC – event is yet unprocessed, waiting for server execution, currently queued
- NOPROC – event is yet unprocessed, waiting for server execution, not yet queued
- PROCD – event is complete and has been processed by the server. If this STATUS is listed, then by definition there was an error processing this event since all events successfully processed are immediately deleted from the EVENTQ table.

This script is helpful during execution of the server and when wanting to know which events are currently being processed, how many to go before completion, what kinds of errors are being produced, etc.

Option 2: **listQ.sh *dbName* ERRORS DETAIL** – will write a **detailed list** of all errors currently existing on the EVENTQ table to output file in $PQLXLOG directory.

Errors include, but are not limited to: missing response files, trace files that do not overlap, trace files not containing enough data for processing, etc.

Option 3: **listQ.sh *dbName* ERRORS SUMMARY** – will write a summary of all errors currently existing on the EVENTQ table; output written to $PQLXLOG directory.

Other Server-Side Programs

In addition to the main server analysis program, other server-side programs are provided:

1. **pqlxPNG –** generate PNG format plots of PDF's.
2. **pqlxDBMaint** – a database maintenance program
3. **rePSD** – re-execute PSD computations for a particular channel and time range

pqlxPNG

This program generates PNG plots of PDF's. Execution may be either automatic (e.g., as part of a cron job), or manual (from the command line), and may be executed directly on a server machine, or from a client machine.

Two types of executions are possible. Option 1 generates PNG plots for all System PDF's currently held by a PQLX database. Option 2 generates a single PNG plot for a user-provided input file.

Option 1

To generate PNG plots of all System PDF's, command-line usage is the following:

```
bash> $(PQLXBIN)/pqlxPNG --systemPDF --dbName=[SERVER:]pqlxDB [--pngDir=/path/to]
```

Where **--systemPDF** and **--dbName=** arguments are both required. If executing from a client machine specify the server machine name or IP address (where the PQLX database resides) as part of the dbName parameter (e.g., --dbName=SrvrName:MYPQLX).

Optional Arguments:

--icon
If specified, will additionally output an icon version of the System PDF plot.

--pngDir=
If not specified, **pqlxPNG** outputs the PNG plots to the directory specified by the **dbWWWDIR** entry in the PQLX database definition file. If this is not defined, then pqlxPNG will quit without doing anything. If this is case, or when wanting to override the database-defined directory, specifying a directory with this option will output the PNG plots here.

Option 2

To generate a single PNG plot based on user-provided input, command-line usage is the following:

```
bash> $(PQLXBIN)/pqlxPNG --inputPDF --pngName=title --pngDir=/path/to
```

Where all arguments are required. In this case, an input file must be provided for PDF generation. Typically, this is done using the output of one of the PDF extract routines provided in $PQLXBIN and piping this to pqlxPNG. These extract routines are one of the following: **exPDFfreq** and **exPDFhour** (execute without arguments for complete usage details).

Alternatively, this file may be generated by the user himself. The format of this file is the

first line containing PDF definition information followed by the PDF values. Create an output file using one of the PDF extract routines above and simply match the format.

Other Options

Other options affecting the output of pqlxPNG:

--width=
--height=
--noBorder
--icon

Where --width= and --height= indicate, in pixels, the exact width and height of the PNG plot to be generated, and --noBorder indicates that no border should be drawn around the plot itself.

Providing a width of less than 240 pixels also indicates that a thumb-nail version of the PDF plot will be made. In this case, no Y-axis (dB scale) and no Color Bar are drawn. This is useful for creating small PDF plots to be used for a web page implementation.

--icon option will additionally create an icon version of either the System PDF's or the input data.

pqlxDBMaint

Previously, execution of pqlxDBMaint was required to be managed by the PQLX DB administrator. This has been reworked whereby the server will automatically execute the pqlxDBMaint program according to the following rules:

- If server has executed at least eight (8) times since the last time the pqlxDBMaint program was executed
- If server is executing on a Saturday
- Whichever happens first

Reasoning

Over time, a PQLX database, depending on exact usage, may become fragmented, causing the pqlxSrvr program to gradually require more and more time to execute, as well as possibly be moved to a “crashed” state by the MYSQL database server.

As well, all events with errors existing on the EVENTQ are deleted, with one exception. Error “No Response File” are never deleted, thus retained forever. This allows for the case when new channels of data arrive perhaps before the meta-data describing the instrument is known or available. Once the response file is made available to the system and identified, all events previously marked with this error are reset to status 'UNPROC', essentially returning them to the EVENTQ for processing.

rePSD

One method of forcing a re-computation of all PSDs for a given channel and date range is via the shell script **rePSD**. This will delete all previously computed PSDs for the given

channel and date range and create PQLX server events to re-analyze the corresponding trace files.

Usage:

```
$(PQLXBIN)/rePSD [HOST:]DBName NTW STN LOC CHN START-DATE END-DATE [ numCPUs ]
```

where:

```
START-DATE & END-DATE define the time period for which PSDs should
    be recomputed (inclusive)
    format: YYYY-MM-DD

numCPUs: optional parameter specifying the number of CPUs to use when
         re-initiating the server.
    Not specifying numCPUs will delete the PSDs and recreate pqlxSrvr events
    for subsequent processing, but WILL NOT re-initiate the server.
    This is useful when wanting to delete several ranges of PSDs
    (in time or by channel) before re-initiating the server.
```

examples:

```
bash> $(PQLXBIN)/rePSD micros IU ANMO -- BHE 1996-12-01 1998-02-01
```

to delete the PSDs for channel IU.ANMO.--.BHE between 01-DEC-96 and 01-FEB-98 and create PQLX events for re-computation, but DO NOT execute the server

```
bash> $(PQLXBIN)/rePSD micros IU ANMO -- BHZ 1999-04-01 1999-06-01 4
```

to delete the PSDs for channel IU.ANMO.--.BHZ between 01-APR-99 and 01-JUN-99 and re-initiate the server to process all outstanding PQLX events, to execute on 4 CPUs

PQLX Client Execution

Once the server has completed its initial execution, the PQLX database is ready for reading and displaying of data by the client GUI, **pqlx**. The client may be executed as:

```
bash> $PQLXBIN/pqlx
```

Once started, different databases (and/or servers) may be connected to. Please see the document '**PQLX-5 Data Display Program.pdf**' for a quick overview of the client program's functionality and usage.