SigQC Database Administrator's Guide

1.1

Phil Wilkin

Signalysis, Inc.
431 Ohio Pike
Suite 182 South
Cincinnati, Ohio 45255
ph. 513-528-6164
fax 513-528-6181

1/28/2002 5:20 PM

Abstract

Most installations of the SigQC software do not fall under the contract of the plant's IT department and therefore it becomes the quality engineer's job to maintain the system. This guide is designed to give that engineer a clear understanding of how to quickly and easily manage the database associated with the SigQC system. Properly administrating this database is essential to the continued smooth operation of the test system and critical to ensuring the integrity of the data.

State of the Document

This document is designed to be an in-depth guide but currently falls far short of its mark. Currently it provides a good overview of some of the more important topics. It will provide little value to someone who is not at all comfortable with the concepts of computers, databases, and file systems. But it will provide a good base for someone who is comfortable with these topics and should led that person to discover the specifics of how to manage the SigQC database.

Revision History

Revision	Date	Initials	Description
1.0	28 Jan 02	PTW	Birth of formal database documentation
1.1	24 Jan 03	PTW	Added more on crashing, added info on coverting

1. Anatomy of a Database

A SigQC database is made up of numerous files in a specific folder. These files are data and key files that all have the name SIGQCDB. These files make up the bulk of the day-to-day information storage for your production test system. There are, however, additional system files that complete the stored information for the SigQC application. These files are more general and can be used by multiple databases, but since the data and key files reference them, they should be considered as part of each database. These system files include graph preferences, communications setup, digital I/O setup and analyzer setup files. The system files are located in folders other that the database folder.

1.1 Data Files (.D??) and Key Files (.K??)

Name: *SigQcDb.D??* and *SigQcDb.K??* where the ?? represents a two digit number
Location: Any folder on the computer, but the total path length cannot exceed 40 characters.
Description: This collection of files contain all the tables and records that make up the SigQC database of information. This includes product definitions, acceptance test case setups, unit data, pass/fail criteria and much more.

1.1.1 Data File Records

1.2 Graph Preferences File

Name: sigqc.gpr
Location: In the application installation folder where the SigQC executable file exits. The default installation makes this location *c:\program files\Signalysis\SigQC*.
Description: Contains the graph preferences that are setup through the *File* | *System Preferences* | *Graphs* menu. These preferences contain the information about graph colors, trace markers, etc… Since this file is stand-alone, it is used for all databases and users that work with SigQC. If this file does not exist it will be created the first time it is accessed.

1.3 Communications Setup File

Name: qcsercom.scm
Location: In the application installation folder where the SigQC executable file exits. The default installation makes this location *c:\program files\Signalysis\SigQC*.
Description: Contains the serial communications definitions that are setup through the *File* | *System Preferences* | *PLC Serial I/O* menu. These define information about serial communication input and output command structures. Since this file is stand-alone, it is used for all databases and users that work with SigQC. If this file does not exist it will be created the first time it is accessed.

1.4 Digital I/O Setup File

Name: plcsetup.shw
Location: In the application installation folder where the SigQC executable file exits. The default installation makes this location *c:\program files\Signalysis\SigQC*.
Description: Contains the digital I/O definitions that are setup through the *File* | *System Preferences* | *PLC Digital I/O* menu. These define the information about digital input and output labels and bit groupings. Since this file is stand-alone, it is used for all databases and users that work with SigQC. If this file does not exist it will be created the first time it is accessed.

1.5 Analyzer Setup Files

Each analyzer used to collect data for SigQC requires specific files of its own. Since these files are accessed directly by SigQC, their location and backup is essential to the proper operation of the application.

1.5.1 Bruel and Kjaer PULSE Files

Name: Have the extension .PLS
Location: Any folder on the computer. It is recommended practice to create a subfolder to the SigQC folder and store all related analyzer files there. Or store the analyzer file with the database where it is used.
Description: This file is generated by the B&K PULSE system and it contains all the setup information about the analyzer's configurations, measurements, functions, displays, and workspaces.

1.5.2 Signalysis SigAnalyzer Files

Name: Have the extension .SAN
Location: Any folder on the computer. It is recommended practice to create a subfolder to the SigQC folder and store all related analyzer files there. Or store the analyzer file with the database where it is used.
Description: The Signalysis SigAnalyzer software generates this file and it contains the setup information about channel configuration, tachometer, post-processing options, and etc…

2. Database Version History

The SigQC database often needs to change structure with the new release of the SigQC application. There are certain changes to the database structure that can be made that allow for forward compatibility of the database without requiring manual conversion (no database is ever backward compatible with respect to application version). Other changes to the database require a manual conversion to upgrade the database. Each section below describes a series of SigQC releases where a particular database structure was forward compatible. If the application upgrade requires the database to cross to a new section, then a manual upgrade of the database will be required. For example upgrading SigQC from 1.03.07 to 1.03.09 does not require a manual database upgrade, but to change from version 1.03.09 to 1.03.13 would require a manual upgrade.

2.1 Database Version 1.01.03

SigQC version 1.03.07 through 1.03.10
Dxx Files = D01, D02, D03, D04, D05, D06, D07
Kxx Files = K01, K02, K03

2.2 Database Version 1.01.04

SigQC version 1.03.12 through 1.03.14
Dxx Files = D01, D02, D03, D04, D05, D06, D07
Kxx Files = K01, K02, K03

2.3 Database Version 1.01.05

SigQC version 1.03.15 through 1.03.16
Dxx Files = D01, D02, D03, D04, D05, D06, D07
Kxx Files = K01, K02, K03

2.4 Database Version 1.01.06

SigQC version 1.03.17 through 1.03.21
Dxx Files = D01, D02, D03, D04, D05, D06, D07
Kxx Files = K01, K02, K03

2.5 Database Version 1.01.07

SigQC version 1.03.23 thru 1.04.00
Dxx Files = D01, D02, D03, D04, D05, D06, D07
Kxx Files = K01, K02, K03

2.6 Database Version 1.01.08

SigQC version 1.04.01 through 1.04.04
Dxx Files = D01, D02, D03, D04, D05, D06, D07, D08
Kxx Files = K01, K02, K03, K04

2.7 Database Version 1.01.11

SigQC version 1.04.05 through 1.04.11
Dxx Files = D01, D02, D03, D04, D05, D06, D07, D08, D09, D10, D11, D12, D13, D14
Kxx Files = K01, K02, K03, K04, K05, K06

2.8 Database Version 2.00.00

SigQC version 2.00.00 through 3.07.09
Dxx Files = D01, D02, D03, D04, D05, D06, D07, D08, D09, D10, D11, D12, D13, D14, D15, D16, D17, D18
Kxx Files = K01, K02, K03, K04, K05, K06, K07

2.9 Database Version 3.07.10

SigQC version 3.07.10 through current
Dxx Files = D01, D02, D03, D04, D05, D06, D07, D08, D09, D10, D11, D12, D13, D14, D15, D16, D17, D18
Kxx Files = K01, K02, K03, K04, K05, K06, K07, K08

3. Database Registration

4. Making Backups

It is recommend that periodic backups be made of the information files that are used by SigQC. This backup can be to any media that will accommodate the size of the data that needs to be saved. This usually includes a network file server hard drive, writable CD-ROMS, or tape backup systems. There is no mechanism built into SigQC to perform a backup, so the process is completely manual. To perform a manual backup, SigQC must be shutdown completely. Then each of the files described in the section Anatomy of a Database should be copied to the media of choice.

The purpose of the backup is to keep the production downtime to a minimum if the SigQC database should become corrupt or the SigQC computer system should go down and need to be fixed or replaced. It essentially provides a replacement database in emergencies when the SigQC system encounters certain fatal errors.

5. Maintaining Archives

The purpose of an archive is to have a history of the data for all units that have been tested by the SigQC system. This allows for the data for any unit to be recalled and reviewed in the future. This usually takes the form of creating a CD of the SigQC database which essentially provide a snapshot of all the data taken for a particular date range. Maintaining archives of your data and maintaining the database’s size are closely integrated and both should be considered together to create a complete database administration of the SigQC database.

6. Managing Database Size

It is extremely important that the SigQC database size is managed properly to maintain the speed and integrity of the database. There are three physical limitations to size of a SigQC database. The first is the size of the hard drive, which in these times of 100G+ hard drives makes this the least of concerns. The second is the limitation that no D?? file or K?? file can be larger than 2G in size which does not appear to be a large concern due to the layout of the data in the database, but could be if large time history records are stored routinely. The third limit is that no D?? file can contain more that 16.7 million records, which has been shown by experience, is a limit that can be reached if the size of the database is not managed properly.

There are three main techniques for managing the size of the SigQC databases. The first is to use the automated deletion features of SigQC. Secondly the size can be maintained by periodic manual data deletion. The third is a technique of replacing the database with a fresh one on a regular basis. Each of these will be discussed further below.

6.1 Automatic Data Deletion

The concept of auto deletion is simply that at a specific user defined time throughout the day, the SigQC system will search the database for data older than a user specified amount of time and delete that data. This keeps the database at a relatively constant size since each day old data is deleted and new data is added. Automatic deletion can be enabled and configured in SigQC through the *File | System Preferences | PLC Digital I/O* menu. The following precautions should be reviewed/taken before enabling auto deletion of data:

?? When data deletion has been changed from disabled to enabled, the first deletion sequence could take a substantial amount of time. For example lets say auto deletion has just been enabled to check at midnight and delete all data that is older than two months. If the database has data for an entire year then tonight at midnight it will need to delete ten months of data. Tomorrow and all following days at midnight it will only need to delete one day of data.

?? The time chosen for auto deletion should be at a time when SigQC will be running, but production will not. Depending on the amount of data collected during a day, and the number of units run for a particular day, the amount of time required to delete one day's worth of data may vary. If the production line is running during the data deletion process, the production cycle time may become longer, but both processes are designed to work together.

?? Once the data is deleted it is gone from the database for good. If an archive of all past unit's data is required then it is essential that the archive be performed regularly to prevent data being automatically deleted before it is archived. Archiving is a manual process and if neglected the auto deletion can cause the archives to be incomplete.

6.2 Manual Data Deletion

Manual data deletion requires the user to set aside time on a regular basis to run the function in SigQC to manually delete data. This functionality is found in the menu *Edit | Delete Production Units* and allows for specification of acceptance test case to delete data from and the date range to delete data for. To manage the database size, a periodic schedule needs to be established to manually delete old units from the database. If data archiving is also desired then that needs to be considered when doing the manual delete. A schedule example for managing database space and keeping archives would go something like the following:

?? First Time
  - Close SigQC
  - Make sure you have backups of the data
  - Archive entire database
  - Reopen SigQC
  - Manually delete all data older than two months

?? Every two months thereafter
  - Close SigQC
  - Make sure that you have backups of the data
  - Reopen SigQC
  - Manually delete all data older than last archive date
  - Close SigQC
  - Archive the database
  - Reopen SigQC

6.3 Swap Method

The swap method works well if the setup of SigQC rarely changes. Using this method, things such as the target limits and production line sequence can only be changed if a swap of databases is done. The basic concept of the swap method is that you keep a database that has had all the data deleted from it and has been compacted. Essentially this database then just contains the setup items such as the target limits, production sequence, acceptance test cases, etc… Then periodically you archive away the SigQC database and replace it with this basically empty database to be filled with data until you swap next time. A schedule example for managing database space and keeping archives using the swap method would go something like the following:

?? First Time
  o Close SigQC
  o Make sure you have backups of the data
  o Archive the entire database
  o Reopen SigQC
  o Manually delete all units from the database
  o Close SigQC
  o Compact the database
  o Copy the database to a new folder named “Swap Db”
  o Reopen SigQC

?? Every two months thereafter
  o Close SigQC
  o Make sure you have backups of the data
  o Archive the entire database
  o Delete the database
  o Copy the contents of the “Swap Db” folder to the database folder
  o Reopen SigQC

?? If you need to make changes to the setup either
  o Make sure you have backups of the data
  o Make your changes to the database
  o Treat this as a First Time case as above

?? or
  o Close SigQC
  o Make sure you have backups of the data
  o Archive the entire database
  o Delete the database
  o Copy the contents of the “Swap Db” folder to the database folder
  o Make your changes to the basic database
  o Copy the contents of the database folder to the “Swap Db” folder replacing the old one
  o Reopen SigQC

6.4 Compacting

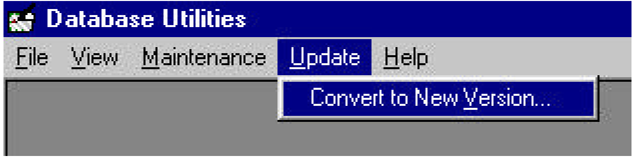
Deleting data from a database does not reduce the size of the database. Records are tagged as deleted and then are re-used when new records are written to the database. Therefore to change the size of the files after deleting units a compaction process must be performed. One of two utilities is required to compact the size of a database – either the DOS utility DbDefrag.exe or the Windows utility DbUtil.exe.

7. Updating Database to New Version

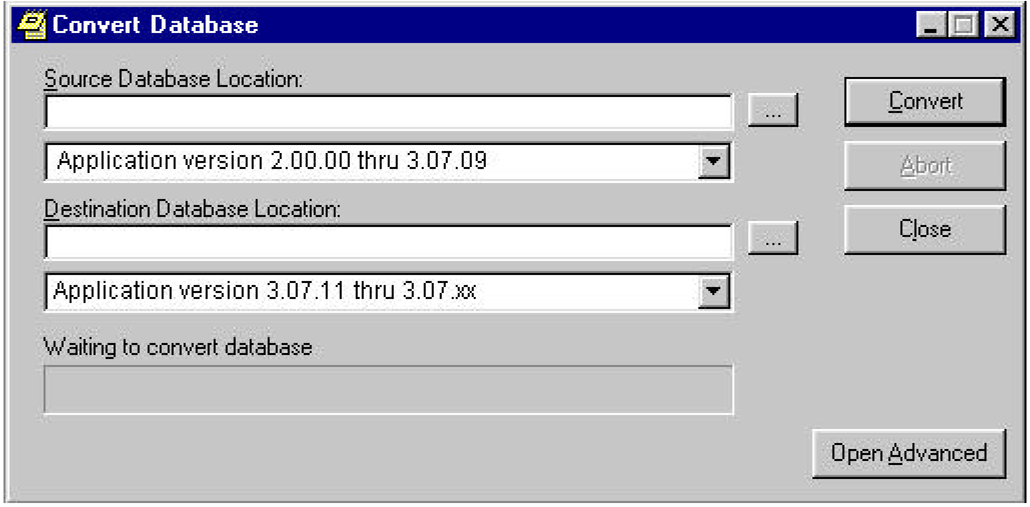
7.1 Automatic Update

7.2 SigDbUtil Update

Run the SigbUtil application that shipped with the newest version of SigQC and select the menu *Update | Convert to New Version* to begin the conversion process.



Then select the location of the original database and the location of the destination that the updated database will occupy. Do this by selecting the Browse button (…) next to each of the source and destination boxes and selecting the folder with the dialog supplied.



Source Database Location:

Application version 2.00.00 thru 3.07.09

Destination Database Location:

Application version 3.07.11 thru 3.07.xx

Waiting to convert database

If you click the Open Advanced button the dialog will expand to show more options. The letter prefix for the options show the command parameter to be used if this program is run in command mode. Note: In most cases the default options will never need to be changed. The options are:

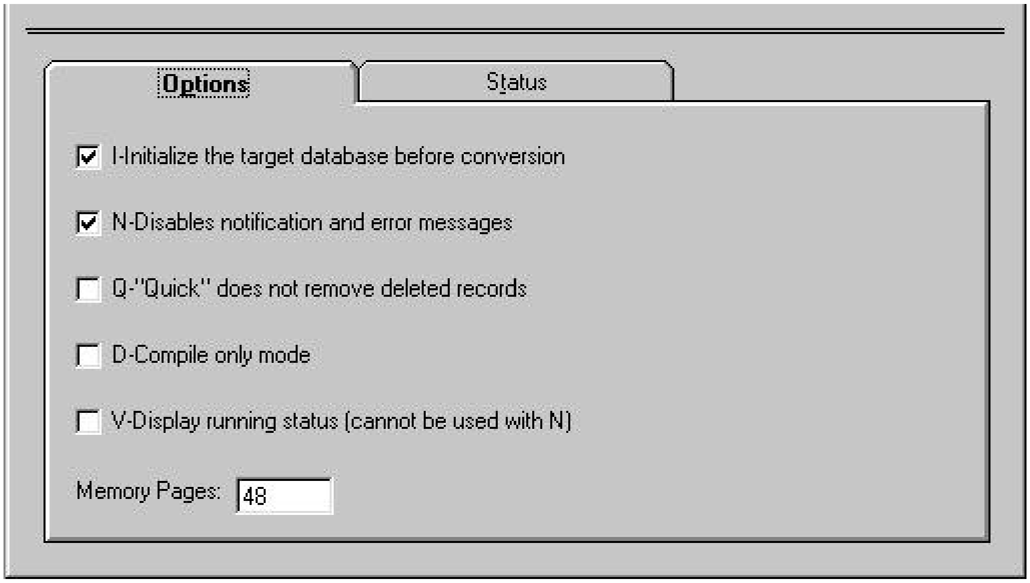
?? Initialize the target database before conversion – Initializes the database before the conversion is started, deleting any records that existed. There are only extreme cases where this would not be required. Default is checked.

?? Disables notification and error messages – Select to reduce the notices given from the program when errors occur. Default is checked.

?? Quick – Clicking this will speed up conversion by not deleting empty records in the database. This is only useful if you have a large number of deleted records in your old database and do not wish for the new database to remove these to save hard disk space. Default is unchecked.

?? Display running status – Displays verbose messages about the status of the conversion. This requires more user interaction because the messages must be cleared before revision will continue. Default is unchecked.

?? Memory Pages –The number of database pages for the utility to store in memory. The higher the number the faster the conversion will go but you run a risk of not having enough memory on your system; 18 is the minimum. Default is 48.



☑ I-Initialize the target database before conversion

☑ N-Disables notification and error messages

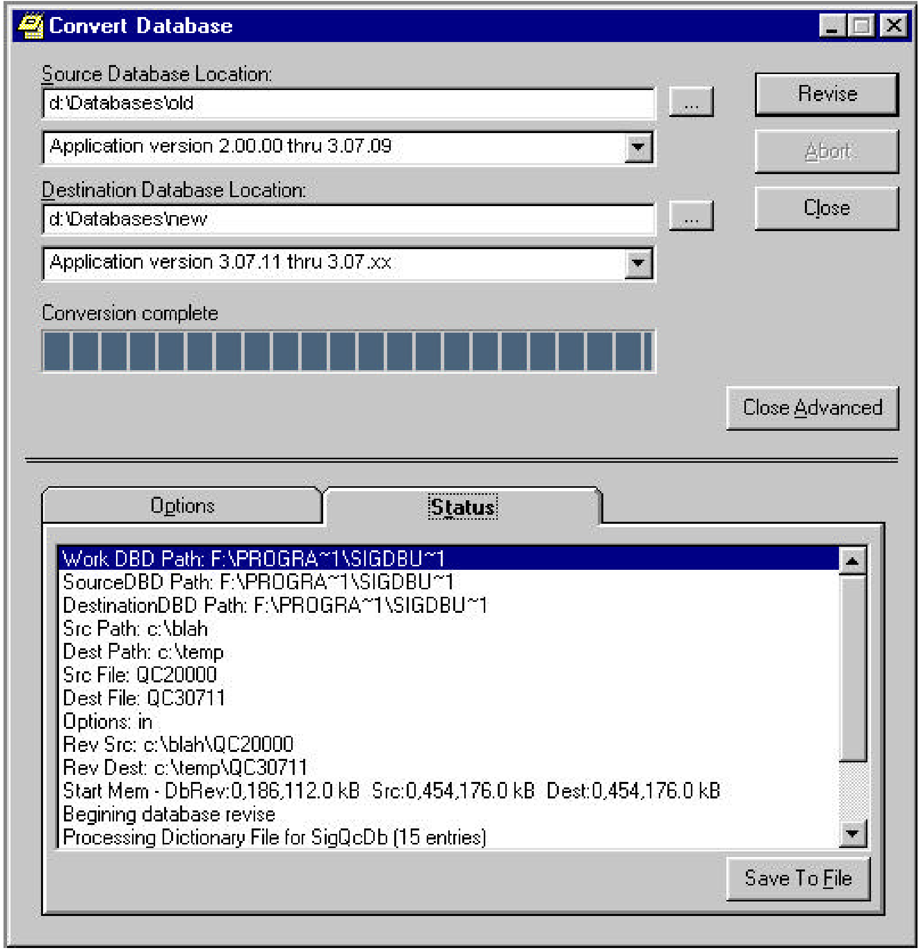
□ Q-"Quick" does not remove deleted records

□ D-Compile only mode

□ V-Display running status (cannot be used with N)

Memory Pages: 48

When the conversion is finished the Convert button will change to a Revise button, allowing you to rerun the conversion. Additionally the Status tab under the advanced options will be updated with information regarding the conversion process. At the bottom of the dialog there is an option to save the Status log to a file so that it can be shown to a tech support professional.



Source Database Location: d:\Databases\old

Application version 2.00.00 thru 3.07.09

Destination Database Location: d:\Databases\new

Application version 3.07.11 thru 3.07.xx

Conversion complete

```
Work DBD Path: F:\PROGRA~1\SIGDBU~1
SourceDBD Path: F:\PROGRA~1\SIGDBU~1
DestinationDBD Path: F:\PROGRA~1\SIGDBU~1
Src Path: c:\blah
Dest Path: c:\temp
Src File: QC20000
Dest File: QC30711
Options: in
Rev Src: c:\blah\QC20000
Rev Dest: c:\temp\QC30711
Start Mem - DbRev:0,186,112.0 kB  Src:0,454,176.0 kB  Dest:0,454,176.0 kB
Begining database revise
Processing Dictionary File for SigQcDb (15 entries)
```

7.3 Fast way to upgrade 2.00.00- 3.07.09 database to 3.07.11- 3.07.xx

A new database key was added to the database which requires this upgrade. The data structures in the database were not modified substantially. Due to the nature of the database change between these versions, just rebuilding the keys for the newer version can perform an upgrade. This should only be used by SigQC tech support representatives and not the general user.
The steps are as follows:

- ?? BACKUP your databases -- this upgrade is done on the actual databases. NO copies are made.
- ?? Open your old version of SigQC and unregister all your databases
- ?? Install new version of SigQC
- ?? Install SigDbUtil version 2.01.00.01 or greater
- ?? Run SigDbUtil
- ?? Note: Key build sequence starts here
- ?? Select Maintenance | Rebuild Key Files menu
- ?? In the Rebuild Key Files dialog select your database location using the Source Database browse button
- ?? The version of the database will automatically be detected as 2.00.00 - 3.07.09
- ?? You need to override this version setting and select version 3.07.11 - 3.xx.xx
- ?? The system will complain when you try this, but it will allow you to select it
- ?? Rebuild the keys by selecting the Build Keys button
- ?? The DOS window will show you the progress of the key build and any errors
- ?? If the key build is error free then close the DOS window and SigDbUtil
- ?? Your database folder should now contain a ".K08" and a ".D00" file
- ?? Repeat the key build sequence for each database
- ?? Run SigQC 3.07.11 and Open each of your converted databases.
- ?? All databases were unregistered so you will need to enter both the directory and name for each database

8. Recovering from a System Crash

There are a couple different consequences of a system crash. Below are some of the most common, with descriptions and solutions where necessary.

8.1 SigQC takes a long time to start

To avoid data loss or corruption in the database, the engine maintains transaction log files. When data is written to the database it is first written to a transaction log file. If the system were to crash at this point the log file entry would not be completed and when the database restarted it would be cleared from the log file. This would cause the data to be lost but the database would not be corrupted. If the write to the transaction log file is successful then the data is written into the database files. If the system were to crash at this point the database would be corrupt, but since the engine has the transaction log file, when it restarts it can repair the corruption because it knows it did not complete the writing of the transaction to the database. It is because of this transaction repair of the database that it could take several minutes to restart after the system crashed. This is typical if the crash were to occur in a long process such as deleting the data for 10,000 units from the database.

8.2 SigQC shows the splash screen and then shuts down

The SigQC database engine uses transaction processing to try and prevent the loss of data when the application is unexpectedly shutdown. There is a certain case (as of yet undefined) where the transaction log is corrupted by the unexpected shutdown and so when the engine tries to restart, it cannot clear the transaction log and exits without warning.

The best fix to this problem is to try and prevent the unexpected shutdown altogether. This means using UPSs on the computer to help during power brownouts and blackouts. Also making sure that the users know how to properly halt production and close SigQC.

The corrupt transaction log can be cleared by deleting some files, but you must realizes that it is likely that some data loss will occur and possible database corruption. If a database backup were available then using it would ensure that the database has not been corrupted by the lost transactions. Below are the steps required to clear this issue:

?? Backup the database that was in use when the system went down unexpectedly

?? If using a previous backup, delete the database files for database that was in use and replace them with the files from the backup

?? Locate the Catalog folder by looking and the Windows Environment variable CATPATH. Typically this folder is located in C:\Program Files\ Common Files\Signalysis\Catalog.

?? Note that the Catalog folder has a brother folder called Syslog

?? From the Catalog folder, delete the file RDM.CHI

?? Do NOT delete the Syslog folder, but do delete the entire contents of the Syslog folder

?? Run SigQC and it should now start properly.

The next upgrade of the database engine (by a third party company) will likely fix the transaction log corruption.

Efforts have begun to make SigQC realize that the database engine was the cause of the last failed run of SigQC and prompt the user if he would like the offending transaction log cleared. This would then delete the necessary files.

Efforts are also underway to make SigQC easier to shutdown properly, and better at responding to Windows messages for shutdown.

There is still no way, and likely will never be a way, to handle the case for shutdowns caused by the user pressing the power button or loss of power to the computer.

9. Frequently Asked Questions

Q. What is the quickest method for corrupting the data in my database?
A. By not properly shutting down the SigQC application. When shutting down the computer, use the proper Windows method for shutting down. Powering off the system without shutting down SigQC causes information in memory to be lost without properly writing it to the hard drive. While SigQC is open, it also maintains a couple temporary files that can be lost or corrupted if the system is not shutdown properly. Therefore it is also recommended that the SigQC computer system be attached to an Uninterruptible Power Source to avoid improper shutdown when there is a power blackout.