Last updated on 04/17/2017
wx2sql version 0.1.1

Introduction

wx2sql is an application that takes weather observations recorded recorded by WeatherLink and enters them into a MYSQL database. To ensure cross-platform functionality, MYSQL was developed in Java using JavaFX. Wx2sql will run on, and has been tested on, Windows, Linux, and macOS. However, only a Windows installer is available for download.

With WeatherLink, the folks at Davis provide a file layout document that was essential to developing wx2sql. Also, their support team was very willing to answer a couple of questions that I had in regards to the file layout. I want to say thank you to these guys. This application would not have been possible without their openness. In addition to the quality of their products, this type of support and openness is what will make me a repeat customer.

Disclaimer

This program comes as-is, with no warranties either expressed or implied. I am not, and can not, be held responsible for any damages or loss of data that occurs as a result of downloading and using this script. Only continue if you completely understand this, accept these risks, have read and understand the known issues listed below, and have the knowledge and skills necessary to implement this process.

Table of Contents

Required Components

To get a working installation of wx2sql, you will need the following:

- a working instance of MYSQL
- wx2sql setup executable, available for download from http:/www.hoosierweather.com/software
- database setup script, available for download from http://www.hoosierweather.com/software
- Java 8

Upgrading from Previous Versions

Version 0.1.1 contains no database changes from 0.1.0. With only a couple of new features and numerous bug fixes, you are not required to make any changes to your 0.1.0 installation prior to using the new version. Please refer to the **Changes from 0.1.0 to 0.1.1** for a detailed list of enhancements.

Setting up the Databases for the First Time

It is outside of the scope of this document to provide a step-by-step guide to assist with the setup and configuration of a MYSQL server. There is plenty of good information readily available with a quick Google search. As we continue, we will operate under the assumption that you have a working instance of MYSQL.

In preparation of running our database script, please ensure the following steps have been completed.

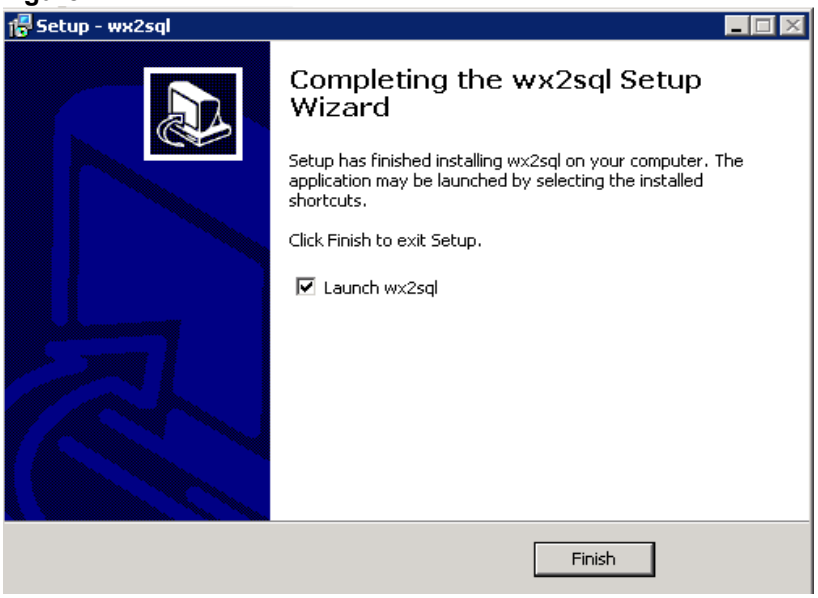
- Create two databases, wx2sql_prod and wx2sql_test. One database will be used for 'production' data, while the other can be used for any testing you may want to do without affecting your production data.
- Create a user for each of these databases. Each user will need the following permissions:
  - ALTER
  - INSERT
  - SELECT
  - UPDATE
- Download the Database script and execute it. This script will setup, or update, the necessary Tables.

Installing wx2sql

Once downloaded, launch the wx2sql_setup.exe. I won't spend a lot of time on this, it really is as easy as 'Next" / "Install" / "Finish". Please see **Figure 4-1**. wx2sql is a Java based application and has been tested on Linux, macOS, and Windows. However, as of the date of this documentation, only the Windows installer is available. If there is enough interest, I will look at packaging this on other platforms.

Installing wx2sql *(continued)*

Figure 4-1.



Installed Files

Outside of files dropped in c:\Program Files (x86)\wx2sql\, a folder is created in the home directory of the user that installed the application named **wx2sql**. This folder will contain your configuration file, job queue information, and any user-generated reports. Below is a detailed description of these items.

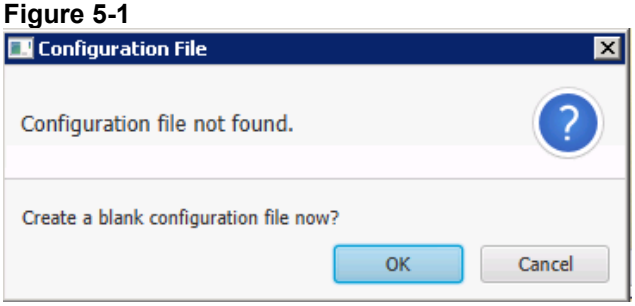
config.properties
This is the main configuration file for wx2sql. This is a straight-forward text file, and while it is not recommended, you can edit this file directly. Remember, no changes will take affect until you restart the application. Use caution when modifying the database password. Depending on your configuration setting, this could be an encrypted value. Incorrectly modifying this field will require you to reset the password from within the application.

jobQueue.xml
This xml file is used to store the progress of all jobs in the queue. This file is updated constantly while the program is running. During the shutdown process, the last task of a job is to record its current status here before it goes to sleep. **DO NOT MODIFY THIS FILE**, doing so can cause unexpected behavior in the application.

Configure Application Settings – Preferences

Once installed, configuring wx2sql is your next step. Select **Edit** → **Preferences**. If no configuration file exists, wxsql will prompt you to create one. Please see **Figure 5-1**

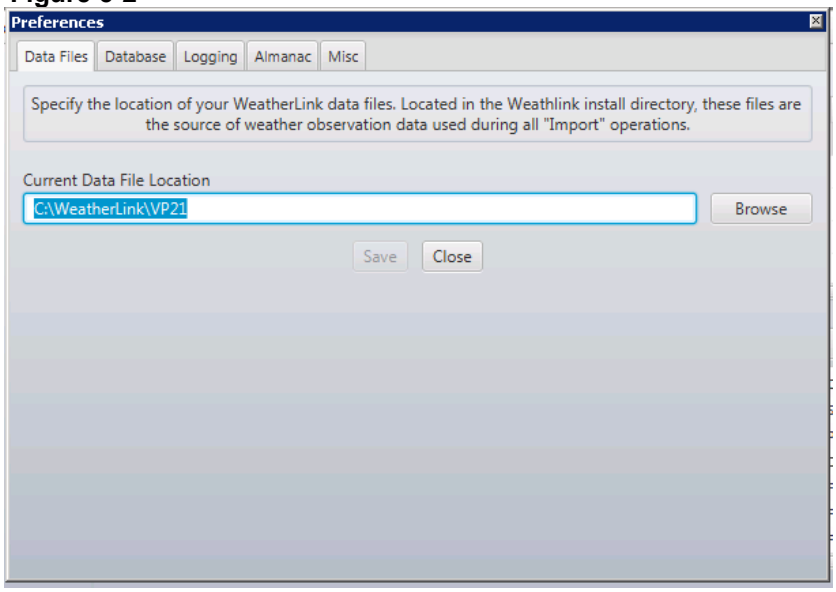
Figure 5-1



Preferences Tab - Data Files

You need to provide the location of the WeatherLink data files that will be used in the Import and Monitor jobs. Select **Browse,** then navigate to the directory where these files are located, generally in the **<Installed Drive>:\WeatherLink<WeatherStation Name>**. Select **Save** to write changes to the configuration file. Please see **Figure 5-2.**

Figure 5-2



Configure Application Settings – Preferences *(continued)*

Preferences Tab - Database

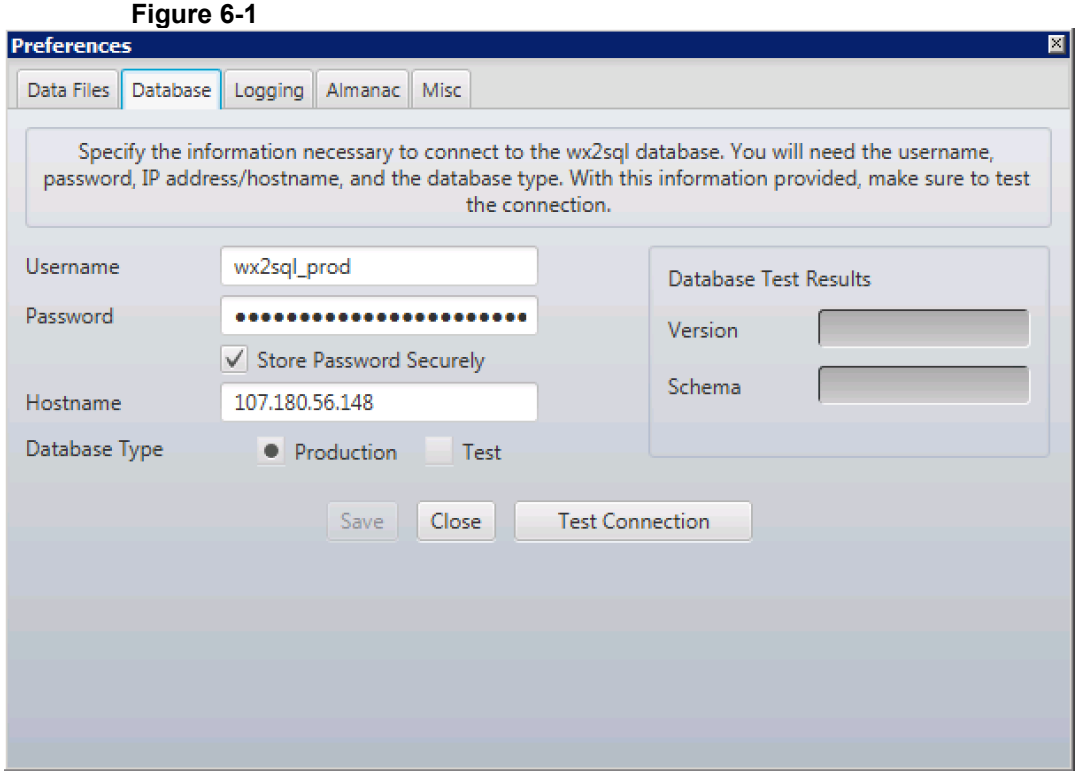
Connecting to the database will require four pieces of information:

- Database Username
- Database Password
- Hostname/IP Address of the MYSQL server
- Database type you wish to connect to, Production or Test

By default, Database Passwords are stored in clear text within the configuration. To store the password as an encrypted value, select the option **Store Password Securely**.

When these configured fields are populated, the **Test Connection** button will be enabled. After making changes to the connection settings, please test your connection. If your connection is successful, the server version and Database Schema version will be displayed. Please see **Figure 6-1**

Figure 6-1

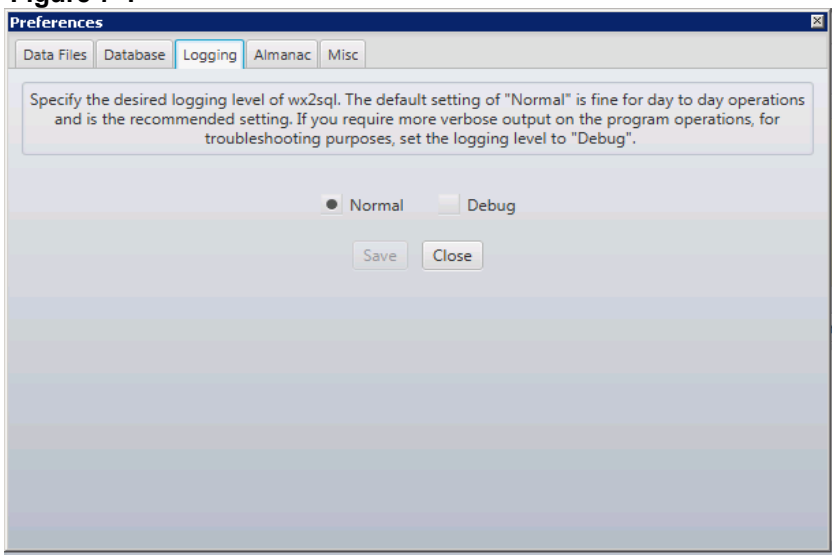


Configure Application Settings – Preferences *(continued)*

Preferences Tab - Logging

The logging subsystem of wx2sql manages log events generated by the application and will display these log entries on the application's main window in the Log Viewer. You can control how verbose the logging system is by setting the logging level to either **Normal** or **Debug**. For normal, day-to-day operation, the **Normal** setting will be sufficient. When experiencing an issue that requires troubleshooting, use the **Debug** setting to increase verbosity. Please see **Figure 7-1.**

Figure 7-1



Preferences Tab - Almanac

wx2sql can connect to the United States Naval Observatory in order to pull Moon Phase, moonrise and moonset, and sunrise and sunset. The following information is required for this operation:

City/Town
Rather self-explanatory, but this is the city for which you are going to pull information for.

State
Dropdown the list of states and select the state in which your city is located.

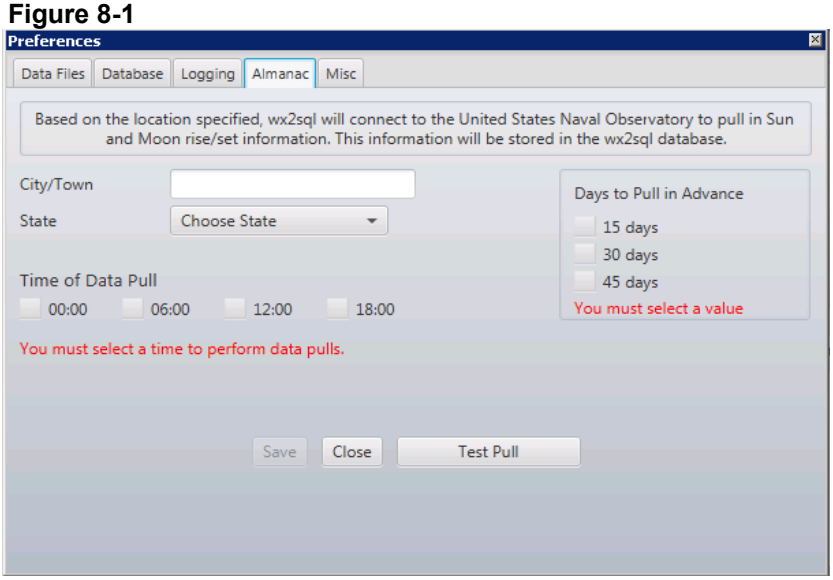
Time of Data Pull
Pulling information from the USNO will occur once a day. This can happen at 00:00, 06:00, 12:00, or 18:00. Select the your desired time.

Days to Pull in Advance
All of the previous pulled data will remain in the database, you can configure how many days in advance for which you want to pull data.

Once you have configured all of the options, it is highly recommended that you select the **Test Pull** button. This connects to the USNO web service, validating your settings. If the City/Town and State combination you entered cannot be found, an error will be displayed. Please see **Figure 8-1**

Configure Application Settings – Preferences *(continued)*

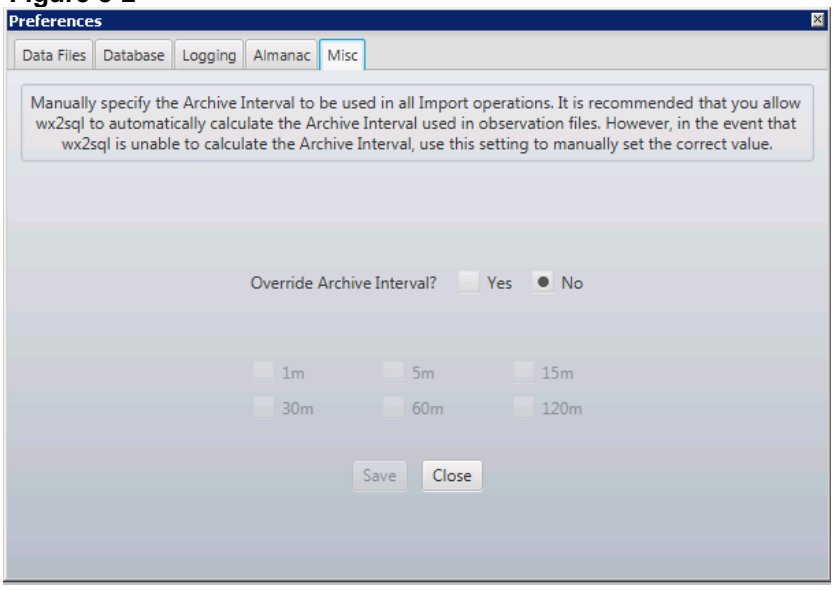
Figure 8-1



Preferences Tab - Misc

The final configurations are made on the Misc tab. As part of wx2sql's error correction model, it will attempt to automatically calculate what the Archive Interval should be for the day being processed. If this value is incorrectly calculated, it may prevent a file from being imported. If the Archive Interval is calculated incorrectly, you can choose to override the calculated value by changing the **Override** setting to "Yes:, then selecting what value the Archive Interval should be. Please see **Figure 8-2.**

Figure 8-2



WLK File Utilities – Health Check

As the first tool in a planned suite of WLK file utilities, Health Check enables you to analyze your existing WLK files, checking for a variety of potential issues. Health Check verifies the following items:

- Analyzes each day to ensure it does not contain more than the maximum allowed number of observations.
- Verifies each data record's record type to ensure that it is of a valid type
- Analyzed the data file to detect any duplicate observations
- Ensure the number of total observations in the file header matches the number of actual observations contained in the file.

Generating the Report

To analyze a file and generate a summary report, select **Data → File Utilities**. On the **Health Check** tab, click the **Browse** button. Using the file selector tool, browse to then select the data file you wish to analyze. Finally, click the **Run Diagnostics** button. Once complete, you will see the report listed in the table.

Contents of the Report

The top portion of the report contains information gathered from the Header section of the data file. This includes the file version, total observations, days with observations, and the result of the diagostic. **Please see figure 9-1.**

Figure 9-1

WLK File Health Check

Report Created on 04 13 2017 00:14:43 EDT

Data File:	OLD BROKEN2017-02.wlk	Total Observations:	38753	**Result: Failed**
File Version:	WDAT5.3	Days with Observations:	26	

Below the header summary, you will find a descriptions of the issues, if any, that were found. **Please see figure 9-2.**

Figure 9-2

Errors Found: 3

- Number of observations for day 26 exceed maximum value (1440). Found 2706 observations.
- Invalid Data Summary Record Type Found: Type 0 found on day 26 at index 1280
- 2703 Duplicate records found

The rest of the document displays data summaries for each day observations were found. If the utility found a problem for a particular day, that day will be displayed in red. **Please see figure 9-3.**

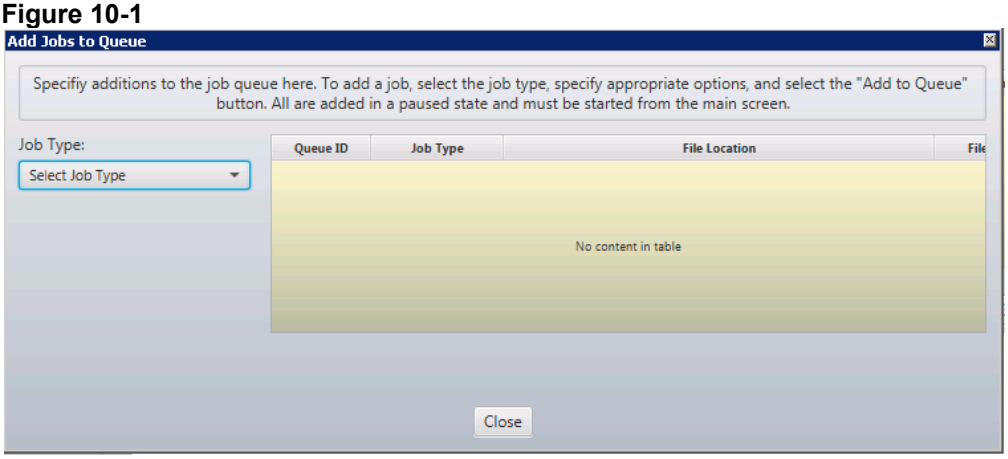
Figure 9-3

17	1442	Yes(1)	Yes(1)	1440	1442	false
18	1442	Yes(1)	Yes(1)	1440	1442	false
19	1441	Yes(1)	Yes(1)	1439	1441	false
20	1442	Yes(1)	Yes(1)	1440	1442	false
21	1442	Yes(1)	Yes(1)	1440	1442	false
22	1442	Yes(1)	Yes(1)	1440	1442	false
23	1441	Yes(1)	Yes(1)	1439	1441	false
24	1442	Yes(1)	Yes(1)	1440	1442	false
25	1442	Yes(1)	Yes(1)	1440	1442	false
26	2706	Yes(1)	Yes(1)	2703	2705	false

Job Types and Queues

Each job type, when running, is assigned to a Job Queue. Wx2sql provides you with up to five queues in which to run your jobs. You can run any combination of the available job types: Almanac Sync, Import, and Monitor. Jobs are added to the queue from the **Add Jobs to Queue** that can be access by selecting **Job Queue → Add Jobs to Queue**. The various job types each have options that are specific to the job, these are explained in more detail below. All jobs are added in the "Paused" state. Once the job is added, close the window to get back to the wx2sql main window. To start a job, right-click on the Job and Select **Resume**. In addition, by right-clicking on the job, a job can be **Canceled** and then removed from the queue all together. Please see **Figure 10-1.**

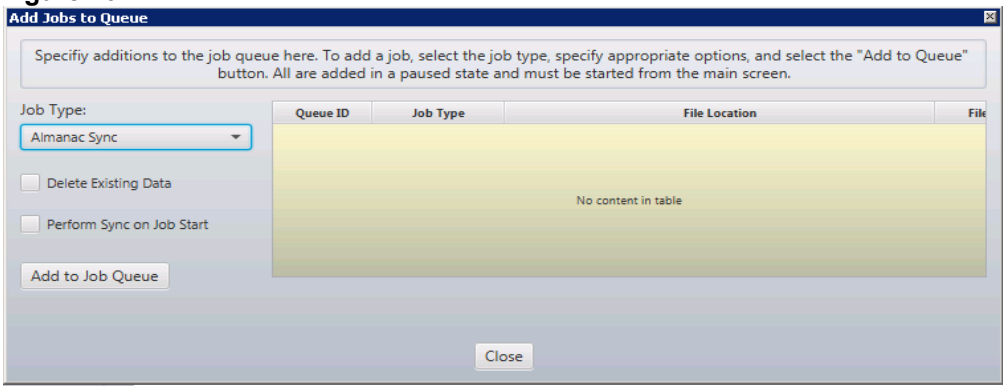
Figure 10-1



Job Type: Almanac Sync

The purpose of the purpose of the Almanac Sync job type is to download information from the USNO web service and input that data into our database. Once you have selected **Almanac Sync** for the job type, you will be presented with two more options: **Delete Existing Data** and **Perform Sync on Job Start**. **Delete Existing Data**, if selected, will remove all Almanac information from the database before attempting to pull in new data. This is down only once, and only when the job is newly added and started. If you wish to perform this task later, you will have to remove any existing Almanac Sync job and re-add it with this option selected. **Perform Sync on Job Start** will, as soon as the job is started, begin pulling in the amount of data that was specified in the Preferences section. Once complete, the job will schedule the next pull also based on the selection made in the Preferences section. Like the **Delete Existing Data** option,**Perform Sync on Job Start** will only sync once when the job is newly added and started. If you wish to perform this task later, you will have to remove any existing Almanac Sync job and re-add it with this option selected. Please see **Figure 10-2**.

Figure 10-2

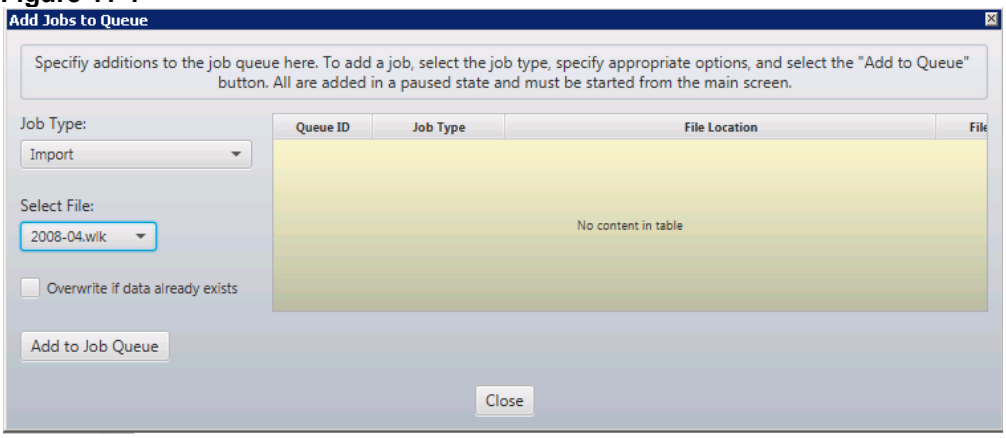


Job Types and Queues *(continued)*

Job Type: Import

The purpose of the Import job type is to read in observational data from a WeatherLink data file and input that data into our database. Once you have selected **Import** for the job type, you will be able to select the file to import. If you do not see any files listed, check your **Data File** setting in Preferences. The last option to set is **Overwrite if data already exists**. This is useful if you have a previous import job that failed, and you want to replace that data. If selected, wx2sql will analyze the data file for the date/time range of the observational data, remove that data from the database, then begin the import process from the beginning. Please see **Figure 11-1**

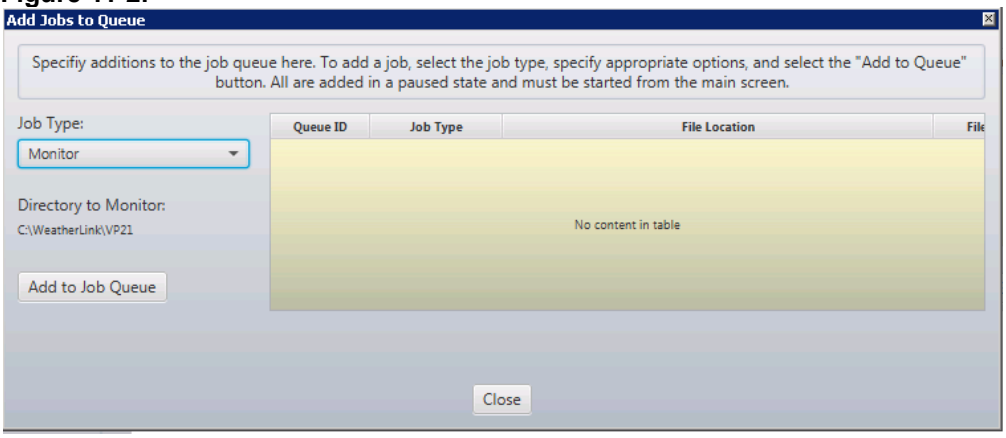
Figure 11-1



Job Type: Monitor

The purpose of the Monitor job type is to monitor the active WeatherLink data file for changes, parse the file, and input new data into the database. After selecting the **Monitor** as the job type, there are no other options to configure. Ensure the **Directory to Monitor** is correct, that it points to the 'live' WeatherLink directory that contains the *.wlk files. If that is correct, select **Add to Job** Queue. Please see **Figure 11-2.**

Figure 11-2.



Understanding the Database

Observation Data

With few exceptions, observation data is stored in
the database in the same format as it was read in from the WeatherLink data files. As such, the data can not be used exactly as it is extracted from the database. Below is a translation table that can be used to format the data in a way that makes it usable.

Column Name	Description	Necessary Formatting
obsn_ep	Time of observation in epoch time	Convert epoch to calendar date
out_tmp	Outside Temperature	Divide value by 10
in_tmp	Inside Temperature	Divide value by 10
bmeter	Barometric Pressure	Divide value by 1000
out_hum	Outside Humidity	Divide value by 10
in_hum	Inside Humidity	Divide Value by 10
wnd_spd	Wind Speed	Divide Value by 10
wnd_dir	Wind Direction	Stored as a 16 point compass value 0 = north, 4 = east, etc. 255 = no value
rain_clicks	Number of clicks from rain sensor	If your rain type = 4096, then for each click, you will have received .01” of precip. For example, if you have recorded 30 clicks (30*.01), you will have received .3” of precipitation
rain_type	Type of rain collector. Used in calculating rain measurements	4096 = each click denotes .01” of rain

Simple SQL queries can be written in order to extract data to be used in your application. An important thing to keep in mind is that, for all observation data, times are stored in Epoch time. Epoch time is the number of seconds that have elapsed since 00:00:00 UTC, Thursday, January 1st, 1970. Many languages have built in functions that will convert epoch time to a human readable time stamp. Below are a couple of links that will provide you with more information.

Wikipedia – Unix Time
https://en.wikipedia.org/wiki/Unix_time

Web app to convert epoch time to human readable and back.
http://www.epochconverter.com/

Almanac Data
The almanac tables are more normalized than the observational data, thus pulling data from these tables is a little more complicated. Below are sample SQL queries to pull information from the DB for January 8th, 2017

Current Phase
SELECT lk_moon_phase.phase FROM almanac LEFT JOIN lk_moon_phase ON almanac.curphase=lk_moon_phase.ID WHERE month='1' AND day='8' AND year='2017';

Location (including City, State, and County)
SELECT location.city, location.county, location.state FROM almanac LEFT JOIN location ON almanac.location=location.ID WHERE month='1' AND day='8' AND year='2017';

Understanding the Database *(continued)*

Sunrise
SELECT sun.time FROM almanac LEFT JOIN sun ON almanac.sundata=sun.date WHERE month='1' AND day='8' AND year='2017' AND sun.phen='R';

More queries will be added here for examples at a later date.

Error Correction

When it comes to recording individual observations, invalid data is a distinct possibility. The format docs specify invalid data values such as 32768 or 255, depending on the data type. wx2sql recognizes other values as invalid, such as a negative humidity value, or a humidity value that exceeds 100%. However, the invalid data that wx2sql is most concerned with are invalid 'Packed Time' and 'Archive Interval' values. Packed time is defined as the "minutes past midnight of the end of the archive period. Archive Interval is the number of minutes contained in the archive record.

Keeping observations in the correct chronological order is very important to ensuring data integrity. If an observation record contains an incorrect time, wx2sql will do its best to calculate the correct value and perform many tests to ensure the value is correct. Below is an example from actual data illustrating invalid Packed Time and Archive Interval Values.

Packed Time	Archive Interval	Temp	Hi Temp
1133	1	386	386
1134	1	386	386
61	0	34	97
117	0	105	108
77	0	105	99

When browsing this data from within WeatherLink, the invalid times are not displayed. Wx2sql, when browsing the data file, will read/display the data as-is. It is important for wx2sql to calculate the appropriate Packed Time so the file structure remains consistent and the observation is correctly removed when deleting observations for a specified month. A Packed Time is determined to be invalid if it matches any of the following criteria:

- Packed Time value is less than 1 or greater than 1440
- Archive Interval that are not set to the following values: 1, 5, 10, 15, 30, 60, or 120. These are the only values allowed within WeatherLink. Any other value is invalid
- The packed time currently being processed has not already been processed for the day being processed. In the example 1133 and 1134 are valid times, 61, would be invalid as that value would have already been processed.
- wx2sql determines the Archive Interval being used, and uses that value to calculate what the Packed Time value should be. If the calculated value does not match the current value, it is marked as invalid. There are instances where wx2sql might not be able to determine the Archive Interval. If this problem occurs, you can override the Archive Interval calculation by hard coding the Archive Interval in the Preferences screen. Please refer to the section on the Preferences screen in this document.
- Any given Packed Time value should be greater than the one that came before it and less than all valid values after it. If this is not true, the value is marked as invalid.

If the Packed Time value is determined to be invalid, wx2sql will calculate what the value should be and will use that value. Due to that value being invalid, all other data in that record cannot be trusted and will be marked as null.

Error Correction *(continued)*

When repairing invalid Packed Time values, wx2sql will attempt to calculate the correct value by using the initial Packed Time value and Archive Interval and count forward, then use the last Packed Time value and Archive interval to count back. If the calculated values match it will be used. If the calculated values do not match, the process will fail with an Error Code. Please refer to the section on Error Codes in this document .

Error Codes

Beginning with 0.1.0 Beta 3, errors occurring in wx2sql will display an Error Code. This will help keep the Log Viewer cleaner by not displaying error descriptions every time an error occurs. Please refer to Table X for Error Code definitions. In 0.1.0 Beta 3, this functionality is new and has not been implemented throughout the entire application. Further implementation will occur in future releases.

Error Code	Description
101	Packed Time validation failed, the value has fallen out of the expected range of 1 – 1440. The error that is logged will contain the Day, Record, and Value that was found.
102	Archive Interval validation failed, the value found was not valid. Valid values for Archive Interval are: 1, 5, 15, 30, 60, and 120. The error that is logged will contain the Day, Record, and Value that was found.
103	Duplicate detection validation failed, a duplicate of the value we are processing has already been processed. The error that is logged will contain the Day, Record, and Value that was found.
104	Packed Time validation failed, the value being processed is less than previous valid values. All previous Packed Times should be less than the current value.
105	Packed Time validation failed, the value being processed is greater than future valid values. All future Packed Times should be greater than the current value.
200	Repair Packed Time Failed, validation checks on the repaired Packed Time returned an unexpected value.

Changes from 0.1.0 to 0.1.1

- Added feature that allows the user to encrypt the database password that is stored within the configuration.
- Added WLK Utilities section. The first tool made available in this release will allow you to perform a health check on your WeatherLink data files to look for potential problems.
- Fixed bug that required the monitor process be restarted in order to begin monitoring a new month's data file.
- Numerous bug fixes.

Changes from 0.1.0 Beta 3 to 0.1.0 Final

- Added feature that allows wx2sql to fetch moon rise/set, sun rise/set, and moon phase data from the United States Naval Observatory.
- Database Schema version was incremented. This was to accommodate the new Almanac feature. To upgrade the database to the latest Schema, download and run the database script.
- Fixed an issue that caused the wx2sql Monitor process to go into the Abort Status upon a data connection failure, resulting in the process to stop. Now the process will back out of the operation and continue to run.
- Enhanced the shutdown process by adding a UI component that provides the user information on the shutdown process. Also modified the various processes to behave more orderly upon receiving a shutdown request. As a result, the application no longer appears to freeze while shutting down.
- Numerous UI enhancements, including a redesign of the Preferences Menu.
- Bug fixes.

Changes from 0.1.0 Beta 2 to 0.1.0 Beta 3

- Fixed an issue that prevented data from being input into the database when encountering unexpected data values. This required changes to the DB Schema as well.
- Database Schema version was incremented. **Due to changes made, existing databases will need to be dumped and recreated with the updated database creation script.** Existing observation data will be lost and the data will need to be re-imported
- Redesigned the "Validate Database" process.
- Made efficiency improvements to the "Import" process. This sped up the file import process.
- Added functionality to use known good observation times to accurately correct and replace invalid observation times. Invalid times can be a single occurrence or many occurring in succession.
- Began implementation of Error Codes. This will lead to better Error logging. Error Codes will be further implemented in later versions.
- Changed the database connection routine so that it is validating the DB Schema.
- Implemented a totally new error correction model. Wx2sql is now able to appropriately handle and process invalid data.
- As part of the revamped error correction model, logic for calculating the Archive Interval has been added.
- Added the ability to override the Archive Interval for error correction. The user is now able to set the override and specify a value. This has resulted in new Configuration options under Preferences.
- Added the ability to specify a production or test database. This will allow you to conduct testing in a test database before processing in the production database.

Changes from 0.1.0 Beta 1 to 0.1.0 Beta 2

- Added the ability to Export and Import your configuration file.
- Fixed an issue with the logging subsystem that prevented log events from being displayed when the counter reached ~15,000 entries.
- Switched installer program to something a little less primitive. This results in a smaller setup program, but you are now required to have Java installed on your machine.

Known Issues and Planned Improvements

- **Priority: High -** Log viewer spacing is still an issue. A few minor changes were made to 0.1.0 Final, but the layout still remains rather bad. This will be addressed in a later version.
- **Priority: Med –** Implement secure communications between the application and database server.
- **Priority: Low -** I am not fond of the database creation and update process. The plan is to bring this process into the application and eliminate the need to manually run a script.
- **Priority: Low -** The main application window has no content on the left side. You've probably noticed how the layout looks like there should be something there. The first improvement here will be the addition of database/observation summary stats. This is a low priority.
- **Priority: Low** - Being able to edit the observation data is a feature planned for a later release.
- **Priority: Low** - The ultimate goal is to allow wx2sql to export the observational data in its database into a valid WeatherLink data file that can be read by WeatherLink. There are a few data points that wx2sql is not currently storing and this would need to be addressed first.

If you have any additional issues that you encountered that you would like addressed, or have a suggestion for a new feature, please email them to webmaster@hoosierweather.com .