Technical Note

DataLogger Tips and Tricks

The purpose of this document is to provide users with tips and tricks for using the KEPServerEX® DataLogger plug-In. Please verify that the data source is currently supported before using DataLogger.

For more information, refer to kepware.com.

1. Logging Data Consumes Disk Space

Both the amount of data and the frequency at which it is logged can quickly consume disk space. Users must be aware of how their logging choices impact disk space consumption. For example, Kepware® used a wide data format when testing to log 1000 points of data every 10 milliseconds. It took about three days to consume 5GB of disk space.

Most applications do not require such fast rates, but if data is required to be logged at this rate then this disk space usage cannot be avoided.

2. Designing the DataLogger Project

The best way to design a DataLogger project is to determine how the logged data will be used. Knowing this enables users to create data tables in the most optimized format and to use the Data Source Names (DSN) and DataLogger project to populate tables accordingly. For example, a user that intends to log product data by shift (in order to generate a weekly production report) would create a table with records for Shift #, Up Time, Down Time, Parts Produced, Parts Rejected, and Date/Time Stamp.

For more information on database design, refer to the Five Simple Database Design Tips article from Tech Republic.

3. Optimizing Data Logging

3.1 DSNs and Log Groups

The DataLogger threading model can be tailored to suit users' specific needs. DataLogger interprets each DSN as a queue to create a new logging thread. For example, if "LogGroup01" uses a DSN named "MySQL_ONE" and "LogGroup02" uses a DSN named "MySQL_TWO," then DataLogger will generate two distinct logging threads even though both "MySQL_ONE" and "MySQL_TWO" may point to the same database. If "LogGroup01" and "LogGroup02" both use "MySQL_ONE," they will take isochronous (or equal opportunity) turns processing data.

Important: Having more than one DSN connection to the same data table increases the chance of one connection locking the data table in order to insert a record while another connection also attempts to insert a record. This will result in both an error and a loss of data.

3.2 Items per Log Group

The scalability of DataLogger allows users to log tens of thousands of points simultaneously. In the above example, two Log Groups were created that both logged to distinct DSNs pointing to the same MySQL database.

However, this is not a guarantee of logging performance, especially when DataLogger is targeting heavily utilized, network-connected databases. It is important to keep the following factors in mind:

- When triggered, each Log Group will insert a maximum of one row (or record) at a time to the target database. This behavior allows DataLogger to easily integrate with a wide variety of ODBC implementations that may individually differ in row insert batch processing. For column mapping in DataLogger:
    - A row in narrow format means one tag.
    - A row in wide format means multiple tags.
- When the Store and Forward feature is employed, the buffered rows are still only inserted into the database one record at a time. If the Store and Forward buffer fills faster than the buffer can be purged, it can yield the following results:
    - The records in the database will be seconds, minutes, hours, or days older than the newer records stored in the Store and Forward buffer because the buffer uses a FIFO approach (first in first out).
    - The buffer can fill to the maximum storage capacity configured, after which point no additional data can be stored by the Store and Forward buffer.
- To handle such results:
    - Create more Log Groups and a smaller number of tags per Log Group. With a small number of tags (400 or less) per Log Group, the odds of creating large Store and Forward buffers is decreased. Testing is recommended to determine the appropriate number of tags and Log Groups for specific use-cases.
    - If using a Static Interval logging trigger, consider changing the Update Rate defined in the Log Group properties to be equal to the Static Interval trigger rate.

Note: This avoids over-sampling an unchanging value, filling the Store and Forward buffer with unnecessary data.

3.3 Table Size Considerations

With current versions of Kepware and its DataLogger plug-in, target tables with DataLogger should be considered temporary storage areas and care should be taken to archive or remove records that are no longer of use.

4. Logging to Microsoft Excel

DataLogger supports logging to existing tables in Excel files via the Microsoft Access Link Tables feature and the ODBC Driver. For information on logging to an existing table in an Excel file, refer to the instructions below.

Note: DataLogger cannot be used to create new tables in Excel.

1. To start, create a new Access DSN that points to the Access database where the Link Tables process will be performed for the specific Excel spreadsheet.

   Note: It is recommended that users do not utilize a pre-existing Access DSN for the same database. In one test case, doing so yielded poor results.

2. Next, open the associated Access database and then click **File | Get External Data**. Then, select **Link Tables**.
3. Browse to and select the Excel file.

   Tip: When designing a project, choose descriptive names for the table headers in Excel. This will make it easier to map the OPC server items to the table headers through the Map Item Fields dialog.

5. Data Source Handling of Timestamp and Date/Time

DataLogger maintains two timestamp values for each OPC server item in a log group. One timestamp is updated each time the OPC server polls the data source. This timestamp is used when Log on Static Interval is selected as the logging condition. The other timestamp is updated when DataLogger detects a change of value. This timestamp is used when Log on Data Change is selected.

Each database handles the date and time a little differently; however, in some cases, the formatting is very different:

- **Microsoft Access Timestamp:** Access stores date/time internally as a double precision floating point number. Its syntax is xxxx.yyyy. Because the time value is "00:00" at midnight, the yyyy part will not be displayed in the Access user interface.
- **Microsoft SQL Timestamp and Date/Time Differences:** "TIMESTAMP" is interpreted by MS-SQL internally as a binary blob used to process image information. It has nothing to do with OPC timestamps, dates, or time information. Users should always utilize "SQL_DATE" or "SQL_DATETIME" to log OPC timestamp information.

Note: DataLogger can either log data to an existing database table or automatically create new database tables. When creating tables automatically, it will suggest a set of column names and SQL data types that are compatible with the specified database. For example, when DataLogger detects a request to create an "SQL_TIMESTAMP" column in an MS-SQL database, it will post a warning to the Event Log and override the column type (using "SQL_DATETIME" instead).

6. Error Handling – Database Full

Each data source will have a slightly different error response when the database is full. KEPServerEX will post a message to the Event Log indicating that the RecordSet could not be queried. It is recommended that users monitor the "_DataLogger.._Error" Tag from their OPC client application to see when an error has occurred. An error state is indicated when the tag's value changes from zero to one.

7. Monitoring the Data Insertion Rate

Each Log Group has System Tags that specify the amount of time it takes to open the record set for the insertion of a record. If this time is longer than the rate at which data is triggered to log, it is likely that records will be lost. For example, the DataLogger is logging on data change. If the device values change every 250 milliseconds, but the insertion rate is 400 milliseconds, records will be lost if a change occurs while waiting to insert a record.