# DataLogger

# Table of Contents

## DataLogger

Help version 1.021

## CONTENTS

### Overview
What is DataLogger?

### Initial Setup Considerations
How do I configure a DataLogger?

### Setting up DataLogger
How do I run a DataLogger?

### System Tags
What are valid system tags?

### Event Log Messages
What messages does DataLogger produce?

*All company and product names mentioned herein are the trademarks or registered trademarks of their respective owners.*

## Overview

DataLogger is an application that logs data from an OPC server to an ODBC-compliant database. DataLogger's tight integration with the OPC server provides substantial, unique benefits such as simple installation, high-efficiency performance, and easy tag browsing in the OPC browse space.

### Feature Highlights

- Supports several common ODBC-compliant database management systems.
- Has a user-friendly installation and configuration. If a Database Source is defined before DataLogger is launched, users can have an active logging configuration in less than five steps.
- Has flexible triggering. Data logging can be enabled at the following times: always, at absolute times, or when an expression is true (such as when a tag's quality is bad). When enabled, logging can occur based on static / time interval, on log group item data change, on monitor item data change, and / or on start / stop condition transitions.
- Has improved, reliable information logging through the local store and forward file, which is used to bridge network and database outages or delays.
- Includes an OPC server Simulator Driver.
- Includes two hours for evaluation while in Time Limited mode.

### Runtime Performance Features

- Runs as a System Service.
- Can be easily scaled through its support of multiple concurrent logging processes (threads).
- Logs data directly from the local item list without reliance on external OPC servers.
- Supports both automatic table creation and the ability to append data to an existing table.
- Supports error recovery and can automatically reconnect if a DSN connection is lost.
- Supports an optional automatic configuration backup (in which the most recent copy of the configuration file is saved).
- Supports _System Tags that allow optional Runtime control from OPC client applications (such as enabling / disabling logging and monitoring logging status).

## Initial Setup Considerations

The following topics should be reviewed before the first DataLogger Configuration is created.

**System Requirements**
**External Dependencies**
**Supported Data Types**

## System Requirements

The server has minimum system requirements for both software and hardware. These requirements must be met for the application to operate as designed.

This application supports the following Microsoft Windows operating systems:

- Windows 10 x64 (Pro and Enterprise Edition)[3]

- Windows 10 x86 (Pro and Enterprise Edition)

- Windows 8.1 x64 (Windows 8, Pro, and Enterprise Edition)[3]

- Windows 8.1 x86 (Windows 8, Pro, and Enterprise Edition)

- Windows 8 x64 (Windows 8, Pro, and Enterprise Edition)[3]

- Windows 8 x86 (Windows 8, Pro, and Enterprise Edition)

- Windows Server 2019 x64[3,4]

- Windows Server 2016 x64[3,4]

- Windows Server 2012 x64 R2[3]

- Windows Server 2012 x64[3]

🔹 **Notes**

1. When installed on a 64-bit operating system, the application runs in a subsystem of Windows called WOW64 (Windows-on-Windows 64 bit). WOW64 is included on all 64-bit versions of Windows and is designed to make differences between the operating systems transparent to the user. WOW64 requires the following minimums:

    - 1 GHz Processor

    - 1 GB installed RAM (defer to the suggestion for the OS)

    - 180 MB available disk space

    - Ethernet Card

2. Verify the latest security updates are installed for the operating system.

3. Runs in the 32-bit compatibility mode.

4. Windows Server Core deployments are not supported.

◆ *Additional resources are available on the Kepware and PTC websites. In particular, the following resources are helpful in planning stages: KEPServerEX Install Guide, Secure Deployment Guide. Contact a staff system engineer for guidance on requirements and recommendations for more complex systems.*

## External Dependencies

This application has external dependencies. It requires that the ODBC driver for the database management system used is installed on the PC running the OPC server.

DataLogger supports the following ODBC- compliant databases (including, but not limited to):

- Microsoft SQL Server
- PostgreSQL
- MySQL
- Microsoft Access
- Oracle

**Note**: Although DataLogger supplies TimeStamp values with a resolution to one thousandth of a second, certain databases do not support millisecond resolution. For more information on a specific database, refer to the product vendor.

### Recordsets

**Tip**: MSSQL uses Dynamic-type recordsets; MS Access uses Dynaset-type recordsets; all others use Snapshot.

**Dynaset-type Recordset** — the result of a query that can have updatable records. A dynaset-type recordset object is a dynamic set of records that can add, change, or delete records from an underlying database table or tables. A dynaset-type recordset object can contain fields from one or more tables in a database. This type corresponds to an ODBC keyset cursor.

**Snapshot-type Recordset** — a static copy of a set of records that can find data or generate reports. A snapshot-type recordset object can contain fields from one or more tables in a database, but can't be updated. This type corresponds to an ODBC static cursor.

**Dynamic-type Recordset** — a query result set from one or more base tables to add, change, or delete records from a row-returning query. Further, records other users add, delete, or edit in the base tables also appear in the recordset. This type corresponds to an ODBC dynamic cursor (ODBCDirect workspaces only).

### Supported Data Types

DataLogger supports the following data types.

| Data Type | Description |
|-----------|-------------|
| Boolean | Single bit |
| Byte | Unsigned 8-bit value |
| Char | Signed 8-bit value |
| Word | Unsigned 16-bit integer |
| Short | Signed 16-bit integer |
| BCD | Two-byte packed binary coded decimal |
| LBCD | Four-byte packed binary coded decimal |
| DWord | Unsigned 32-bit integer |
| Long | Signed 32-bit integer |
| Float | 32-bit floating point value |
| Double | 64-bit floating point value |
| String | ASCII text string |
| Date | Floating-point OLE automation date |

**Note**: 64-bit integer types, LLong and QWord, are not supported.

## Setting up a DataLogger Configuration

A DataLogger configuration defines how data is extracted from an OPC server project and logged into a database. There is one DataLogger configuration for each OPC server project. Within a DataLogger configuration there are one or more log groups. A log group is a "data pipeline" between the OPC server project and a database table. A log group defines the following:

- A DSN connection to the database.
- The server items (such as OPC server tags) logged to the database.
- The format and name of the table within the database.
- The triggers that govern when data is logged (at a specific time, on data change, and so forth).

The following is the typical process to create a DataLogger project:

1. Create a log group and connect it to a database table. See **Log Group**.

2. Add log items to the log group. See **Log Items**.

3. Create or edit the database column mappings. See **Column Mappings**.

4. Create or modify the logging trigger conditions. See **Triggers**.

5. Enable the log group to begin logging data.

## Opening DataLogger and Adding a Log Group

To access DataLogger, confirm DataLogger is enabled under **View** in the Configuration menu bar. To create a new log group, either:

- Select the **Add a new log group** button from the Configuration button bar.
- Right-click **DataLogger** and select **New Log Group**.

To make a copy of an existing log group, right-click on the log group and select **Copy**. To remove (delete) a log group, right-click on the log group and select **Delete**.

## Setting up a DSN

Before any DSN connection to a database can be used, it must be configured. After it has been configured, the DSN is displayed in the drop-down list for the **DSN** field.

❋**Important:** Users can configure a DSN using the Microsoft® ODBC Data Source Administrator launched from the operating system's Start menu. Because the server is a 32-bit application, it is necessary that the 32-bit version of the administrator be used when configuring a System DSN. A 64-bit operating system launches the 64-bit version of the administrator by default; therefore, users may need to browse to the SysWOW64 directory to access the 32-bit version.

● **Note:** The ODBC drivers for the Database Management System being used must be installed on the PC that is running the OPC server. For a list of supported ODBC drivers, refer to **External Dependencies**.

1. Open or create a **Log Group**.

2. Click the browse button in the **DSN** field to access **Data Source Selection**.

3. Click **Configure DSN…**to access the **Microsoft ODBC Data Source Administrator**.
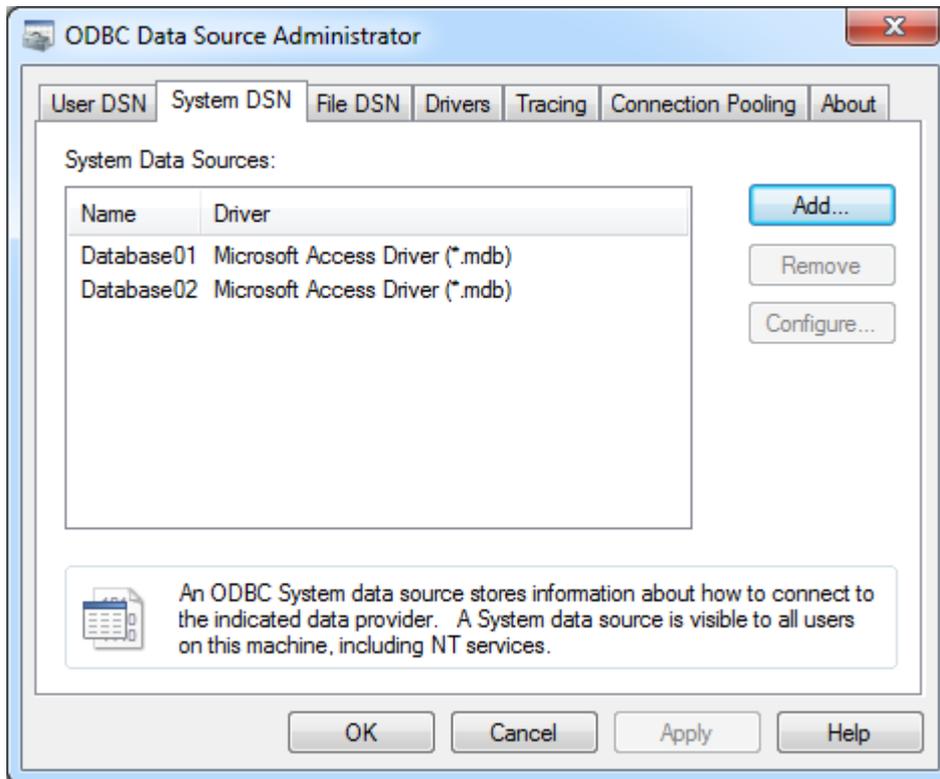


4. Access the **System DSN** tab.

    ● **Note:** For further help with the Microsoft® ODBC Data Source Administrator dialog, click **Help**.

5. Next, click **Add**.

6. In **Create New Data Source**, scroll down the list of drivers to locate the driver for the specific database.

7. Select the driver and then click **Finish**.

8. Continue through the wizard and complete the dialogs displayed. The dialogs and fields vary according to the database being connected.

9. The final dialog of the DSN Wizard may include a **Test** button for the Data Source. Whether or not a **Test** button is displayed depends upon the DSN being configured. If so, click the button to verify that the DSN setup is functional.

## Log Groups — General

| Property Groups | □ **Identification** | |
|---|---|---|
| **General** | Name | DownTimeData_L03 |
| Advanced | Description | |
| | □ **Configuration** | |
| | Enabled | Yes |
| | Update Rate | 100 |
| | Update Rate Units | milliseconds |
| | Map Numeric ID to VARCHAR | No |
| | Use Local Time | Yes |
| | □ **Data Source** | |
| | DSN | Downtime |
| | Username | postgres |
| | Password | ******** |
| | Login Timeout | 10 |
| | Query Timeout | 15 |
| | □ **Table** | |
| | Table Selection | Create new table and always append to this table |
| | Table Name | DownTimeData_L03 |
| | Table Format | Narrow - Each item shares a column mapping |

### Identification

**Name**: Specifies a name for the new log group. The maximum length is 256 characters. It cannot begin with an underscore, or include a period or double quote. The default name is LogGroup1.

**Description**: Specifies a longer, descriptive identifier. The maximum length is 4096 characters.

### Configuration

**Enabled**: Enables the log group. When first created, the log group is disabled. This allows the necessary components to be assembled and configured before the log group is enabled.
 **Note**: For changes to take affect, log groups must be DISABLED first. After making changes, update the **Enabled** parameter back to **Yes** to proceed with the updated settings.

**Update Rate**: Specifies the update rate for data coming from the OPC server to the log group. The default setting is 100 milliseconds.

**Update Rate units**: Specifies the update rate unit for data coming from the OPC server to the log group. The range for each unit is as follows:

- **Milliseconds**: 1 to 99999999
- **Seconds**: 1 to 4294967
- **Minutes**: 1 to 71582
- **Hours**: 1 to 1193
- **Days**: 1 to 49

**Map Numeric ID to VARCHAR**: When enabled, each item in the log group maps the numeric ID column to the VARCHAR(64) data type by default as opposed to the INTEGER data type.
 **Note**: Disabling clears all numeric IDs associated with the log group items.

**Use Local Time**: When enabled, DataLogger uses the local time for timestamp values. When disabled, the timestamp values are in Universal Coordinated Time (UTC).

**Note**: The timestamp written in records by DataLogger is indicative of the most recent data change event.

## Data Source Properties

**DSN**: This drop-down list specifies the Data Source Name (DSN). If the DSN is not listed, it must be configured. To do so, select **Configure DSN** and refer to **Setting up a DSN**. Once a DSN has been configured, it is available to select from the drop-down list.

**Username**: Specifies a user name for the data source. It is optional unless the data source requires it.

**Password**: Specifies a password for the data source. It is optional unless the data source requires it.

**Note**: Some data sources require that Windows Authentication be used to gain access. If the data source is configured to use Windows Authentication, it ignores the user name and password set in Configuration and uses the network login ID instead.

**Login Timeout**: Specifies how long the server waits for a response when attempting to connect to the DSN. At the end of that time, the connection attempt times out. The valid range is 1 to 99999 seconds. The default setting is 10 seconds.

**Query Timeout**: This property specifies how long the server waits for a response when attempting to query the recordset or insert a new record. At the end of that time, the query times out. The valid range is 1 to 99999 seconds. The default setting is 15 seconds.

## Table Properties

**Table Selection**: Specifies both the table and how to log data to the table. Options include Log to an existing table, Create a new table every time the log group is started, and Create a new table once and always append to this table. The default setting is Log to an existing table. Descriptions of the options are as follows:

- **Log to An Existing Table**: When selected, this option logs data to an existing table. Mapping between log item values and database columns will need to be performed manually. *See* **Column Mappings**.
- **Create a new table every time the log group is started**: When selected, this option generates a new table in the database each time the OPC server enters Runtime. Enter a name for the table in the **Table Name** field. The first time this log group is enabled, the table is created. The next time the log group is enabled, another table is created with the name "tablename0". For example, a table named "Baseline" is created the first time the log group is enabled. The next time the log group is enabled, a table named "Baseline0" is created, then "Baseline1", "Baseline2", and so on. This option will produce a default mapping of the database columns, which you may then edit. *See* **Column Mappings**.
- **Create a new table once and always append to this table**: When selected, this option generates a new table in the database the first time that the log group is enabled. Data is logged to that same table each time the OPC server enters Runtime thereafter. Enter a name for the table in the **Table Name** field. The first time the log group is enabled, the table is created. Every subsequent time it is enabled, the data is appended to that same table. This option will produce a default mapping of the database columns, which you may then edit. See **Column Mappings**.

**Table Name**: Selects a table (from the drop-down menu) or specifies a name for the new table to be created.

⬤ **Note**: When using DataLogger to create tables, the maximum length of table names is 256 characters. In practice, however, the maximum depends on the database being used and ODBC driver's limits. Often, the maximum is 64 or 128 characters. Table names should only consist of letters and numbers, because non-alphanumeric characters can cause errors. If non-alphanumeric characters must be used in the table name, consult with the database and ODBC driver specifications for table naming restrictions.

**Table Format**: Specifies the format of the data table. Options include Narrow and Wide formats. The default setting is Narrow. Descriptions of the options are as follows:

- **Narrow - Each item shares a column mapping**: Each item shares a single column map. Each row includes 5 columns: Name, Numeric ID, Value, Quality, and Time.
- **Wide - Each item gets its own column mapping**: Each item gets its own map. Each row includes Numeric ID, Value, Time and Quality for every server item.

✤ For a more detailed description of the narrow and wide formats, see **Narrow vs Wide Table Format**.

## Log Groups - Advanced



### Store and Forward

Enabling store and forward prevents data from being lost when connectivity to the ODBC data source is lost or when the database is unable to process requests fast enough. When these situations occur, the Log Group records the data that was captured during the connection loss to disk. Once connectivity is restored, the data on the disk is read, written to the ODBC source, and deleted.

⬤ **Note**: Clients can use the _Buffering Tag to monitor when the store file is being used. *For more information, refer to* **System Tags**.

**Enabled**: Enables the store and forward functionality. A store file is created at runtime with the path "<storage directory>\<log group name>.bin". When disabled, the other Store and Forward settings are also disabled. The default setting is disabled.

**Note**: When store and forward functionality is enabled, the log group name must only consist of valid characters.

**Storage directory**: Specifies the directory in which a store file is created. The directory may be any valid absolute path that begins with "\\" or "<drive letter>:". The default setting is "<application profile directory>\DataLogger\".
**Note**: Access to the store file depends on the permissions that are configured for the selected directory. The default directory allows read and write permissions to all users.
**Important**: Utilizing a network drive as the store file location is risky. If the network goes down, any store files located on that network drive is inaccessible. As such, it is not recommended that a network drive be used as the storage directory.

**Maximum Storage Size (MB)**: Specifies the maximum file size for the store and forward file. New data is lost if the file size limit is reached. The valid range is 1 to 2047 MB. The default setting is 10 MB.
**See Also**: *Store and Forward Table Constraints*

### Memory

**Maximum Row Buffer Size**: Specifies how many records are held in the row output buffer before logging. The valid range is 1 to 99999. The default setting is 1000.
**Note**: This should be set equal to or greater than the number of log items in the log group or some records may not be logged properly.

### Batch Identifier

**Batch ID Item**: This optional property specifies the ID of the server item to be used as a batch identifier.
**Note**: If a Batch ID item is entered, the item is mapped to a database column by default. When a database transaction is triggered, the current value of the item is applied to the column for all rows associated with the transaction.

**Data Type**: Identifies the data type of the value that will be produced by the server.

**Update Rate**: Specifies the update rate for the server item defined in the **Batch ID Item** field. The default setting is 1000 milliseconds.

**Update Rate units**: Specifies the update rate unit for update rate property above. The range for each unit is as follows:

- **Milliseconds**: 1 to 99999999.
- **Seconds**: 1 to 4294967.
- **Minutes**: 1 to 71582.
- **Hours**: 1 to 1193.
- **Days**: 1 to 49.

### Regenerate Column Mapping Rules

**Regenerate on DSN change**: Controls whether the database column mappings are reset when the DSN is changed. The default value is **Yes**.

**Regenerate on Batch ID change**: Controls whether the database column mappings are reset when the Batch Item ID is changed. The default behavior is to reset on this condition. The default value is **Yes**.

**Regenerate on Table Name change**: Controls whether the database column mappings are reset when the table name selection is changed. The default behavior is to not regenerate on this condition. The default value is **No**.

**Regenerate on Table Selection change**: Controls whether the database column mappings are reset when the Table Selection is changed. The default behavior is to not regenerate on this condition. The default value is **No**.

### Actions

**Reset Column Mappings**: Resets all database column mappings to their default settings. If the **Table Selection** property is set to **Log to an Existing Table**, this action clears all column mappings. Otherwise a default mapping is created.

## Store and Forward and Table Constraints

Store and Forward cannot recover records that fail to insert due to table constraint violations.

- Records that fail to insert into the database table due to a table constraint will log to the data store file. All subsequent records will log to the data store file until the table constraint violations are resolved.
- Options for table constraint violation resolution:
    - With No Data Loss
        - Remove the table constraints from the database table.
        - Change the log group Table Selection to **Create a new table once and always append to this table** to create and log to a new table. Tables automatically created by DataLogger will not have constraints on data fields.
    - With Data Loss
        - Disable Store and Forward. DataLogger will no longer attempt to recover records in the data store file. Subsequent records that fail to insert are discarded.
- A record in wide data format may contain tags values from different sources. Trigger events generated by tags from one data source may initiate a record insert before data is available from other tag sources. As such, record inserts in wide data format are inherently susceptible to table constraint violations.

## Log Items

| Property Groups | Identification | |
|---|---|---|
| **General** | Name | LogItem1 |
| | Description | |
| | **General** | |
| | Item ID | |
| | Numeric Item ID | 0 |
| | Data Type | Default |
| | **Deadband** | |
| | Deadband Type | None |
| | Deadband Value | 0 |
| | Deadband Range Low | 0 |
| | Deadband Range High | 0 |

### Identification

**Name**: This property specifies the name of this log item.

**Description**: This property specifies a longer, descriptive identifier. The maximum length is 255 characters.

### General

**Item ID**: This property specifies the full name of the server item. Use the Browse button to search the OPC server project for an item. If an existing server item was chosen and **Modify** was selected, the Item ID field displays the ID of the server item being modified. The maximum length is 256 characters.

**Numeric Item ID**: This property specifies a numeric alias or identifier for the server item. The valid range is 0 to 2147483647. The default setting is 0.
**Note**: When the VARCHAR mapping is enabled, any string of length 64 is valid. Disabling limits the numeric value between 0 and 2147483647.

**Data Type**: This property identifies the data type of the value that will be provided by the specified monitor item.
**Note**: Data types available are based on the types supported by the plug-in(s) and driver(s) being logged (with some limitations).

### Deadband

**Deadband Type**: This property specifies the item's deadband type. Options include None, Absolute, and Percent:

- **None**: No deadband is used. This is the default setting.
- **Absolute**: If the difference between two consecutive item data updates is more than the amount specified in the Value field, the update is valid and can produce events. Otherwise, the update is ignored.
- **Percent**: This option is similar to Absolute except that the Value field is a percentage of the specified range. For example, if the Value is 10 (10%) and the expected range is 0 to 10, the Absolute deadband would be ±1.

**Deadband**: This parameter specifies the deadband value. Any positive number is supported. If the Deadband Type is set to None, this parameter is disabled. The default value is 0.

**Deadband Range Low**: This property specifies the minimum limit of the monitor item's deadband range. The default value is 0.

**Deadband Range High**: This property specifies the maximum limit of the monitor item's deadband range. The default value is 0.

## Column Mappings

The Column Mappings are used to specify how the properties of the Log Items are mapped to the columns of a database table.

- If DataLogger is creating a new table, the item-to-column mappings is done automatically by DataLogger. Although users do not need to map the item fields, they can customize the columns.
- If DataLogger is using a table that already exists, then Column Mapping items must be updated to map the server items to database columns.

If the log group has been configured to use narrow table format there will be one column mapping that is used for all log items. In wide mode there will be a column mapping object for each log item in the log group. *For more information see Narrow vs Wide Table Format.*

| Property Groups | Identification | |
|---|---|---|
| **General** | Name | TableAlias |
| | Description | |
| | **General** | |
| | LogItem | __internal |
| | **Name Field Mapping** | |
| | Database Field Name | |
| | Data Type | SQL_UNKNOWN_TYPE |
| | Length | 0 |
| | **Numeric ID Field Mapping** | |
| | Database Field Name | |
| | Data Type | SQL_UNKNOWN_TYPE |
| | Length | 0 |
| | **Quality Field Mapping** | |
| | Database Field Name | |
| | Data Type | SQL_UNKNOWN_TYPE |
| | Length | 0 |
| | **Timestamp Field Mapping** | |
| | Database Field Name | |
| | Data Type | SQL_UNKNOWN_TYPE |
| | Length | 0 |
| | **Value Field Mapping** | |
| | Database Field Name | |
| | Data Type | SQL_UNKNOWN_TYPE |
| | Length | 0 |

### Identification

**Name**: This parameter specifies a name for the new column mappings.

**Description**: This parameter specifies a longer, descriptive identifier. The maximum length is 255 characters.

## General

**LogItem**: This property specifies the LogItem associated with this mapping. If the relevant Log Group is in narrow mode, the LogItem is "__internal". In wide mode, each LogItem will have its own mapping.

## Field Mapping

The types of Field Mappings include:

- **Name**: The name of the LogItem.
- **Numeric ID**: The numeric ID for the LogItem.
- **Quality**: The quality of the current data.
- **Timestamp**: The timestamp of the current data.
- **Value**: The data value of the item.

The following properties apply to each Field Mapping type:

- **Database Field Name**: The database column to which the LogItem fields should be mapped. If DataLogger is using a table that already exists, a drop-down of the available column names is provided.
- **Data Type**: This property specifies the SQL data type from the dropdown menu of supported data types.
- **Length**: The length of the data.
  ● **Note**: If the LogItem provides data that is longer than the data length which has been set in the column mapping, this will cause an error which will prevent data from being recorded.

## Character limits

DataLogger enforces a 256-character limit on the Item ID, Server Item, and Database Field Name. In practice, however, the maximum length also depends on the limits of the relevant database and ODBC driver. The maximum name length for a database column is often 64 or 128 characters. When data is logged in Wide Format, the Server Item and Database Field Name values consists of the channel, device, and group names prefixed to the Item ID name. This further limits the serviceable name length of the Item ID. ● *For more information, refer to* **Narrow vs. Wide Table Format**.

● **Note**: Character limits are also affected by software versions and array size (see also **Large Arrays**).

## Triggers

The Triggers object is used to define one or more triggers for a log group. When first created, a trigger contains one default trigger. By default, the trigger is Always Triggered with a 500ms update rate.

| Property Groups | Identification | |
| --- | --- | --- |
| **General** | Name | Trigger1 |
| Logging Conditions | Description | |
| | **Type** | |
| | Trigger Type | Always Triggered |

**General**

**Name**: This property specifies the name of this trigger.

**Description**: This property specifies a longer, descriptive identifier. The maximum length is 255 characters.

**Trigger type**: This property determines when a trigger is true. The default trigger type is Always Triggered.

- **Always Triggered**: The trigger is always true so long as the Log Group is enabled. For example, the trigger does not have a false state, except for when the OPC server is inactive.
- **Time Based**: The trigger is true only for certain days and hours. For example, from Monday to Friday, 8 AM to 5 PM.
- **Condition Based**: The trigger is true when conditions in an expression are true. For example, a trigger can be defined to be true when the value of tag XYZ is greater than 100.

| Property Groups | Logging Conditions | |
|---|---|---|
| General | Log on Static Interval | Yes |
| **Logging Conditions** | Static interval | 500 |
| | Update rate units | milliseconds |
| | Log on data change of any value withi... | Yes |
| | Log entries for all items when the monit... | No |
| | Monitor Item ID | |
| | Monitor item update rate | 1000 |
| | Monitor item update rate units | milliseconds |
| | Data Type | Default |
| | Deadband type | None |
| | Deadband Value | 0 |
| | Deadband Range Low | 0 |
| | Deadband Range High | 0 |

**Logging Conditions**

The Logging Conditions properties are common to all trigger types.

**Log on Static Interval**: When enabled, data is logged on a static interval (time-based). The valid range is 10 milliseconds to 49 days. The default setting is checked. The default value is 500 milliseconds. For more information, refer to **Update Rate**.

**Static Interval**: This property specifies the static time interval for logging. May be set in milliseconds, seconds, minutes, hours or days. The valid range is 10 milliseconds to 49 days. The default interval is 500 (ms).

**Update rate units**: This property specifies the unit used for the static interval property. The default unit is milliseconds. The range for each unit is:

- **Milliseconds**: 10 to 99999999.
- **Seconds**: 1 to 4294967.
- **Minutes**: 1 to 71582.
- **Hours**: 1 to 1193.
- **Days**: 1 to 49.

**Log on Data Change**: When enabled, data is logged when the value of any server item in the log group changes.

## Monitor Item

**Log All Items**: When enabled, data is logged for all items in the log group whenever the value of the monitored item changes.

**Monitor Item ID**: This property specifies the full name of the server item. To locate an item in the OPC server project, click **Browse**. Upon selection, the Monitor Item ID property is populated with the ID of the selected server item. The maximum length is 256 characters.

**Monitor item update rate**: This property specifies the update rate for data coming from the OPC server for the selected item. The default setting is 1000 milliseconds. The range for each unit is:

- **Milliseconds**: 10 to 99999999.
- **Seconds**: 1 to 4294967.
- **Minutes**: 1 to 71582.
- **Hours**: 1 to 1193.
- **Days**: 1 to 49.

**Monitor item update rate units**: This property specifies the units used by the Monitor item update rate property. The default is milliseconds.

**Data Type**: This property identifies the data type of the value that will be provided by the specified monitor item. This property is read-only.

**Deadband type**: This property indicates the item's deadband type. Options include None, Absolute, and Percent. The default setting is None. Descriptions of the options are as follows:

- **None**: No deadband is used.
- **Absolute**: The absolute change in data required to be considered a data change.
- **Percent**: This option is similar to Absolute except that the Value field is a percentage of the specified range. For example, if the Value is 10 (10%) and the expected range is 0 to 10, the Absolute deadband would be ±1.

**Deadband Value**: This property specifies the deadband value. Any positive number is supported. If the Deadband Type is set to None, this property is disabled. The default value is 0.

**Deadband Range Low**: This property specifies the minimum limit of the monitor item's deadband range. The default value is 0. The default setting for both the low and high value is 0; however, the high range value must be greater than the low range value.

**Deadband Range High**: This property specifies the maximum limit of the monitor item's deadband range. The default value is 0. The default setting for both the low and high value is 0; however, the high range value must be greater than the low range value.

**Notes:**

1. The format of the data being logged depends on whether the narrow or wide format is chosen for the log group. For more information, refer to **Narrow vs. Wide Table Format**.

2. Arrays cannot be used with the Stop and Start trigger items when a value comparison is performed.

3. The Update Rate should be set to a value less than the Static Interval value, so that the log group's server items are updated more frequently than the static interval.

4. Since the update rate determines when the server item changes are processed by DataLogger, the lower the update rate, the more frequently data changes are received and processed by DataLogger.

5. DataLogger maintains two timestamp values for each OPC server item in a log group:
   - One timestamp is updated each time the OPC server scans the controller address, regardless of whether the value has changed since the last scan. As such, this timestamp reflects the "freshness" of the value. This is the timestamp value logged to the database if the insertion was triggered by a static-interval trigger (that is, when Log on Static Interval is checked).

   - The other timestamp is updated each time DataLogger has detected a data-value change. If the value of the server item has not changed, the timestamp is not updated. This is the timestamp value logged to the database if the insertion was triggered by a change in the data value (that is, when Log on Data Change is checked).

🟢 **Tip**: If a timestamp provided by the driver or client driver is needed, enable the **Log All Items** property.

🟢 **See Also**:
**Combined Start/Stop Condition Examples**
**Triggers: Condition Based**

## Triggers: Time Based

The Time Based section is displayed when the Time Based trigger type is selected.

| Property Groups | Identification | |
|---|---|---|
| **General** | Name | Trigger2 |
| Logging Conditions | Description | |
| | **Type** | |
| | Trigger Type | Time Based |
| | **Absolute Time** | |
| | Start Logging at | 8:00:00 AM |
| | Stop Logging at | 5:00:00 PM |
| | **Days of the Week** | |
| | Sunday | Yes |
| | Monday | Yes |
| | Tuesday | Yes |
| | Wednesday | Yes |
| | Thursday | Yes |
| | Friday | Yes |
| | Saturday | Yes |
| | **Optional** | |
| | Log All Items on Start | No |
| | Log All Items on Stop | No |

- **Start Logging At**: This property specifies the time of day when the logging should start.
- **Stop Logging At**: This property specifies the time of day when the logging should stop. The stop logging value should occur after the start logging value.

- **Days of the Week**: These properties allow the selection of days during which you would like logging to occur. At least one day must be selected.
- **Log All Items on Start**: When enabled, all items in the log group are logged once when the start time or condition is met.
- **Log All Items on Stop**: When enabled, all items in the log group are logged once when the stop time or condition is met.

**Notes:**

1. The **Start logging at** and **Stop logging at** fields control the logging time for the days selected under Days of the Week. In the example shown above, the trigger would be true from Sunday through Saturday, from 8:00:00 AM to 5:00:00 PM.

2. Overlapping midnight (meaning, overlapping from one day to the next) requires that two triggers be created because this dialog defines the duration per day. For example, to start logging at 9 PM on Friday and stop at 3 AM on Saturday, then one trigger would be created with Friday checked and **Start logging at** set to 9:00:00 PM and **Stop logging at** set to 11:59:59 PM. Another trigger would have to be created with Saturday checked and **Start logging at** set to 12:00:00 AM and **Stop logging at** set to 3:00:00 AM.

## Triggers: Condition Based

The Condition Based section is displayed when the Condition Based trigger type is selected. Note the following:

- An event expression trigger can only include one start condition and one stop condition. To have the log group be triggered by multiple expressions, users must create multiple event expression triggers.
- At least one start condition is required for an event expression. A stop condition is optional.
- The log group is triggered when the start condition becomes true. The log group remains triggered regardless of subsequent changes to the start expression state.
- If there is a stop condition, the log group becomes not-triggered when the stop condition is true. If there is no stop condition, the log group remains triggered until the OPC server Runtime stops.
- If the start and stop conditions are both true, then the log group is not triggered. While the start condition is true, the stop condition controls whether or not the log group is triggered.
- Log groups are not triggered while the stop condition is true.
- Arrays cannot be used for the start and stop trigger items when a value comparison is to be performed.

| Property Groups | Identification | |
|---|---|---|
| **General** | Name | Trigger1 |
| Logging Conditions | Description | |
| | **Type** | |
| | Trigger Type | Condition Based |
| | **Start Condition** | |
| | Item ID | _System._Time |
| | Data Type | String |
| | Item Update Rate | 1000 |
| | Item Update Rate Units | milliseconds |
| | Condition Type | Item data set to bad quality |
| | Data | |
| | **Stop Condition** | |
| | Item ID | |
| | Data Type | Default |
| | Item Update Rate | 1000 |
| | Item Update Rate Units | milliseconds |
| | Condition Type | Item data set to bad quality |
| | Data | |
| | **Optional** | |
| | Log All Items on Start | No |
| | Log All Items on Stop | No |

### Start and Stop Conditions

The following properties are common to both the Start and Stop Conditions.

**Item ID**: This parameter specifies the ID of the server item that controls the condition. The server item can be one that has been selected to be logged, or it can be an item that is not being logged. To search for the server item, click the Browse button to locate and select an Item ID. The maximum length is 256 characters.

**Data Type**: This read-only property indicates the data type of the selected item.

**Item Update rate**: This property specifies the unit used for the static interval property. The default unit is milliseconds. The range for each unit is:

- **Milliseconds**: 10 to 99999999.
- **Seconds**: 1 to 4294967.
- **Minutes**: 1 to 71582.
- **Hours**: 1 to 1193.
- **Days**: 1 to 49.

**Item Update Rate Units**: This property specifies the units used by the item update rate property. The default is milliseconds.

**Condition Type**: This property specifies the Condition. The default is Item Data Set to Bad Quality. Possible selections are:

- Item data has not changed over a certain time period (ms)
- Item data set equal to a specific value
- Item data set greater than a specific value
- Item data set less than a specific value
- Item data set not equal to a specific value
- Item data set to good quality
- Item value is equal to OPC FALSE (zero)
- Item value is equal to OPC TRUE (non-zero)

**Data**: If applicable this parameter specifies the data to be used with the Condition Type.

Note: For start and stop conditions, if the Condition Type is set to "Item data has not changed over a certain time period", the Data field must be at least twice the value of the Item Update rate. This ensures the server has enough time to send data based on the update rate. Depending on the underlying driver, this time may need to be increased even more if the server cannot meet the requested update rate.

### Start Condition Examples:

1. If **Condition Type** is set to "Item value is equal to OPC TRUE (non-zero)," the condition is true if the server item in the **Item ID** field is true. For this condition, leave the **Data** field blank as it is not necessary. This assumes the value in **Data Type** is Boolean; if any other data type is entered, then the server item value cannot be zero to make this condition true.

2. If **Condition Type** is set to "Item data set to bad quality," the condition is true if the server item in the **Item ID** field is quality=bad. For this condition, leave the **Data** field blank as it is not necessary.

3. If **Condition Type** is set to "Item data set not equal to a specific value" and the **Data** field value is 144, then the condition is true if the server item in the **Item ID** field is not equal to 144 (it could be higher or lower).

4. If **Condition Type** is set to "Item data set less than a specific value" and the **Data** field value is 144, then the condition is true if the server item in the **Item ID** field is less than 144.

5. If **Condition Type** is set to "Item data has not changed over a certain time period (ms)" and the **Data** field value is 15000 (15,000 milliseconds), then the condition is true if the tag in the **Item ID** field has not changed value for 15 seconds.

**Stop Condition Examples:**

1. If **Condition Type** is set to "Item data set less than a specific value" and the **Data** field value is 144, the condition is true if the server item in the **Item ID** field is less than 144.

2. If **Condition Type** is set to "Item data has not changed over a certain time period (ms)" and the **Data** field value is 15000 (15,000 milliseconds), then the condition is true if the tag in the **Item ID** field has not changed value for 15 seconds.

3. If **Condition Type** is set to "Item data set to bad quality," the condition is true if the server item in the **Item ID** field is quality=bad. For this condition, leave the **Data** field blank as it is not necessary.

⬥ *See Also: **Combined Start/Stop Condition Examples***

## Combined Start/Stop Condition Examples

### Example One

The Start Condition is set to log on "Item value is equal to OPC TRUE (-1)" and there is no Stop Condition. The tag for the Start Condition toggles between True and False.

| Trigger Tag Value | Start Condition | Stop Condition | Log Group Trigger State |
|---|---|---|---|
| 0 | False | False | False |
| -1 | True | False | True |
| 0 | False | False | True |
| -1 | True | False | True |

### Example Two

The Start Condition is set to log on "Item value is equal to OPC TRUE (-1)" and the Stop Condition is set to "Item value is equal to OPC FALSE (0)". The Start and Stop triggers use the same tag.

| Trigger Tag Value | Start Condition | Stop Condition | Log Group Trigger State |
|---|---|---|---|
| 0 | False | False | False |
| -1 | True | False | True |
| 0 | False | True | False |
| -1 | True | False | True |

### Example Three

The Start Condition is set to log on "Item value is equal to OPC TRUE (-1)" using Tag A and the Stop Condition is set to "Item value is equal to OPC TRUE (-1)" using Tag B. Tag A rarely changes but Tag B toggles between True and False.

| Trigger Tag Value (Tag A / Tag B) | Start Condition | Stop Condition | Log Group Trigger State |
|---|---|---|---|
| 0,0 | False | False | False |
| -1,0 | True | False | True |
| -1,-1 | True | True | False |

| Trigger Tag Value (Tag A / Tag B) | Start Condition | Stop Condition | Log Group Trigger State |
|---|---|---|---|
| -1,0 | True | False | True |
| -1,-1 | True | True | False |
| -1,0 | True | False | True |
| -1,-1 | True | True | False |
| 0,-1 | False | True | False |
| 0,0 | False | False | False |

**Note:**

Arrays cannot be used with the Stop and Start trigger items when a value comparison is performed.

## Valid Trigger Requirements

For each trigger type, at least one of the following conditions must be met for the trigger to be valid.

| | Always Triggered | Time Based | Condition Based |
|---|---|---|---|
| Log on Static Interval | X | X | X |
| Log on Data Change | X | X | X |
| Log All Items on Start Log All Items on Stop | | X | |
| Start Condition Stop Condition | | | X |

## Narrow vs. Wide Table Format

DataLogger can map fields in either the **Narrow Format** or **Wide Format**. Users must specify the format to be used for each database table logged into. The setting is reached through **Column Mappings**.

### Narrow and Wide Format Examples

The following example shows the narrow and wide formats, with three server items that need to be logged.

- Ch_1.Dev_1.Temp
- Ch_1.Dev_1.Pos
- Ch_1.Dev_1.Speed

### Narrow Format Examples

The two tables below show the three server items being logged using the Narrow Format. All three server items have been logged into five columns. The table below shows the narrow format with a static time interval (meaning every *x* seconds or minutes).*

**Narrow Format with Data Logged on Static Interval**

| Name | Numeric ID | Value | Quality | Time |
|---|---|---|---|---|
| Ch_1.Dev_1.Temp | 1000000 | 38 | 192 | 2007 02 16 13:44:26.832 |
| Ch_1.Dev_1.Pos | 1000001 | 22 | 192 | 2007 02 16 13:44:26.832 |

| Name | Numeric ID | Value | Quality | Time |
|------|-----------|-------|---------|------|
| Ch_1.Dev_1.Speed | 1000002 | 103 | 192 | 2007 02 16 13:44:26.832 |

In the table below, the same three server items are being logged using the narrow format; however, data is logged on data change. In this example, Ch_1.Dev_1.Temp has changed value, so it is the only row that is logged.

**Narrow Format with Data Logged on Data Change***

| Name | Numeric ID | Value | Quality | Time |
|------|-----------|-------|---------|------|
| Ch_1.Dev_1.Temp | 1000000 | 38 | 192 | 2007 02 16 13:38:02.142 |

* There was a temperature change (item 1). The other two items did not change.

## Wide Format Example

The next example demonstrates Wide Format. When wide format is selected, four columns are logged for each server item. The four columns are: _VALUE, _NUMERICID, _TIMESTAMP and _QUALITY.

In this example, there are three server items: Ch_1.Dev_1.Temp, Ch_1.Dev_1.Pos, and Ch_1.Dev_1.Speed. In wide format, this would result in up to twelve columns. Columns can be in configured in the Column Mappings property editor so that only necessary columns are logged. In the examples below, the _NUMERICID column has been unmapped for each server item.

Ch_1.Dev_1.Temp_VALUE
Ch_1.Dev_1.Temp_TIMESTAMP
Ch_1.Dev_1.Temp_QUALITY
Ch_1.Dev_1.Pos_VALUE
Ch_1.Dev_1.Pos_TIMESTAMP
Ch_1.Dev_1.Pos_QUALITY
Ch_1.Dev_1.Speed_VALUE
Ch_1.Dev_1.Speed_TIMESTAMP
Ch_1.Dev_1.Speed_QUALITY

**Wide Format**

| Ch_1.Dev_1.Temp_ VALUE | Ch_1.Dev_1.Temp_ TIMESTAMP | Ch.1.Dev_1.Temp_ QUALITY | Ch_1.Dev_ 1.Pos |
|---|---|---|---|
| 38 | 2007 02 16 13:44:26.832 | 192 | 22 |

**Notes:**

1. Only a portion of the data is shown due to the width of the sample table. With the wide format, all fields are logged each time data is logged. In this example, nine columns would be logged every time DataLogger inserted data into the database table. If data was being logged on a static interval, nine columns would be logged every *x* milliseconds. If data was being logged on data change, nine columns would be logged every time any of the fields had a data change.

2. For more information on the options available to log data on static intervals, data change, or on trans-
   ition from a start condition to a stop condition, refer to **Triggers**. **Triggers: Condition Based**, and
   **Triggers: Time Based**.

3. Log groups are limited to 300 columns using wide table format with a MySQL database.

## Large Arrays

DataLogger supports logging arrays of the primitive data types. Array information is converted to a tab-
delimited character string for publication to the database table.

The **Database Column Mappings** collection allows the user to link the Server Item VALUE to the database
field _VALUE. The database element _VALUE defaults to VARCHAR(64), but this size may not accommodate a
large array. The type and size for _VALUE is dependent on the underlying database configured *(see the Gen-
eral tab)*.

**Example:**
With a MySQL DSN, a VARCHAR column length can be specified as a value from 0 to 255 before MySQL 5.0.3
and 0 to 65,535 in 5.0.3 and later versions. For MS SQL Server 2014 VARCHAR(N) can have a specified value
for N between 1 and 8000, although the storage size is the actual length of the data plus 2 bytes. Specifying
an invalid size for the _VALUE type can result in unrelated error messages returned from the database
server or (silent) logging of incorrect values.

 **Notes:**

1. Incorrect configuration of the DSN and/or item fields may result in failures that do not produce error
   messages. If data values appear "stale" (unchanged data logged to the database table even though
   the data changes are occurring), verify the column width is appropriate for the array size and the
   underlying database management system.

2. Arrays cannot be used with the Stop and Start trigger items when a value comparison is performed.

## System Tags

### _DataLogger (Root)

| Tag | Access | Description |
|---|---|---|
| _TriggeredGroupCount | Read Only | The number of log groups that are currently triggered. A log group is triggered when at least one of its triggers is true. |
| _ NonTriggeredGroupCount | Read Only | The number of log groups that are not currently triggered. |
| _EnabledGroupCount | Read Only | The number of log groups that are currently enabled, whether they were enabled at Runtime or via the _Enabled System Tag. |
| _DisabledGroupCount | Read Only | The number of log groups that are not currently enabled. |

### _DataLogger.<*log group name*>

| Tag | Access | Description |
|---|---|---|
| _Buffering | Read Only | When set True, the log group's store file currently contains data.* |
| _Description | Read Only | Configured description of the log group. |
| _Enabled | Read / Write | The log group is evaluating server items and processing triggers. When a log group is enabled, it may or may not be triggered. |
| _Error | Read Only | The log group is in an error state (such as if it failed to connect to the database). |
| _LogDataBit | Read / Write | When set True, all server items are logged. The _LogDataBit tag is reset to False automatically. |
| _SessionInsertionCountHi | Read Only | The upper 32 bits of the insertion count. Insertion count is the number of data insertions since the log group was last enabled. |
| _SessionInsertionCountLo | Read Only | The lower 32 bits of the insertion count. |
| _TimeToOpenRecordsetMS | Read Only | The time it took to initially open a connection to the database in milliseconds. |
| _Triggered | Read Only | The log group is triggered, that is, at least one of the log group's triggers is true. When a log group is triggered, the triggers that are true insert data on static interval, data change, and / or on transition from a start condition to a stop condition, as determined by the logging settings in Add Trigger: Logging Conditions, **Triggers: Condition Based**, or **Triggers: Time Based**. |

* This tag may be True if the database is disconnected and the log group is currently logging directly to the store file. It may also be True if the database has recently been reconnected and the log group is still in the process of forwarding its contents to the database.

# Event Log Messages

The following information concerns messages posted to the Event Log pane in the main user interface. Consult the OPC server help on filtering and sorting the Event Log detail view. Server help contains many common messages, so should also be searched. Generally, the type of message (informational, warning) and troubleshooting information is provided whenever possible.

## Log group '<name>' - <error>

**Error Type:**

Error

## Log group '<name>' failed to validate table '<name>'. Reason: <reason>

**Error Type:**

Error

**Possible Cause:**

1. The log group's Table Selection property is set to 'Log to an Existing Table' and the table specified by the Table Name property does not exist.

2. The log group's Table Selection property is set to 'Create a new table once and always append to this table' and the table's columns do not match the configured Column Mapping.

**Possible Solution:**

1. Create the table using the database management system.

2. Delete the existing table, and then let DataLogger create a new table once and append to it thereafter. | Edit the log group's column mapping to match the table's columns.

## Log group '<name>' failed to create table '<name>' on DSN '<name>'. Reason: <reason>

**Error Type:**

Error

**Possible Cause:**

DataLogger log group specified was unable to create the table for the reason provided by the ODBC driver.

**Possible Solution:**

Correct the conditions that cause the error reported.

## Unable to open recordset on log group '<name>'. Reason: <reason>

**Error Type:**

Error

**Possible Cause:**

DataLogger was unable to create the mechanism that would allow it to perform an insertion because of the reason provided by the ODBC driver.

**Possible Solution:**
If the database is on a remote computer, verify the user permissions are correct and that the network connection is active.

## Unable to open recordset on log group '<name>'. Reason: Memory exception.

**Error Type:**
Error

**Possible Cause:**
The ODBC driver was unable to make the connection because of a memory exception.

**Possible Solution:**
Use a different DSN with a different ODBC source.

## Unknown error opening recordset.

**Error Type:**
Error

**Possible Cause:**
DataLogger encountered an unknown error while opening the recordset.

**Possible Solution:**
If the database is on a remote computer, verify the user permissions are correct and that the network connection is active.

## The query failed because the data source is not appendable. Please double check your user permissions.

**Error Type:**
Error

**Possible Cause:**
The database table is not appendable.

**Possible Solution:**
Often the user must explicitly set an 'append' permission on the table using the database software.

## Unknown error while executing query in table '<name>' for log group '<name>'.

**Error Type:**
Error

**Possible Cause:**

DataLogger encountered an unknown error while executing an insertion on the opened recordset.

**Possible Solution:**

If the database is on a remote computer, verify the user permissions are correct and that the network connection is active.

## Unable to query recordset on Log Group '<name>'. Reason: <reason>

**Error Type:**

Error

**Possible Cause:**

DataLogger encountered an unknown error while attempting to query the opened recordset.

**Possible Solution:**

If the database is on a remote computer, verify the user permissions are correct and that the network connection is active.

## Unable to query recordset on Log Group '<name>'. Reason: Memory exception.

**Error Type:**

Error

**Possible Cause:**

DataLogger was unable to perform the insertion because of a memory exception.

**Possible Solution:**

Use a different DSN with a different ODBC source.

## Unable to query recordset on Log Group '<name>'. <class>: <reason>

**Error Type:**

Error

**Possible Cause:**

1. DataLogger was unable to perform the insertion for the reason provided by the ODBC driver.

2. The data includes an array that is too large or improperly defined.

**Possible Solution:**

1. Correct the conditions that cause the error reported.

2. Address the data type and character limit issues (see Large Arrays).

## Unsupported SQL type '<type>' for column '<name>'.

**Error Type:**

Error

**Possible Cause:**

The SQL type specified is not supported by the DSN.

**Possible Solution:**

Specify a supported data type for the column given.

# Oracle maximum column width of 30 has been exceeded for column '<name>'.

**Error Type:**

Error

**Possible Cause:**

The specified column name is greater than 30 characters.

**Possible Solution:**

Reduce the specified column name to less than or equal to 30 characters.

# The trigger '<name>' must have at least one logging condition set.

**Error Type:**

Error

**Possible Cause:**

The specified trigger is not correctly configured.

**Possible Solution:**

Correct the trigger configuration (see Triggers).

# Failed to load log item '<name>'.

**Error Type:**

Error

**Possible Cause:**

DataLogger was unable to load the specified log item from the project file.

**Possible Solution:**

Edit the project file to correct the log item definition.

# Failed to load trigger expression '<name>'.

**Error Type:**

Error

**Possible Cause:**

DataLogger was unable to load the specified trigger from the project file.

**Possible Solution:**

Edit the project file to correct the trigger definition.

## The log items in log group '<name>' need to be mapped to the columns in table '<name>' before logging occurs. Please setup this column mapping.

**Error Type:**

Warning

**Possible Cause:**

The column mapping for the specified log group is incomplete.

**Possible Solution:**

Edit the column mapping for the specified log group.

## Failed to register log item '<name>' for log group '<name>'.

**Error Type:**

Warning

**Possible Cause:**

The specified item does not exist in the server configuration or has an invalid address format.

**Possible Solution:**

Check the server item and verify that it is a valid address.

## Failed to register trigger item '<name>' for log group '<name>'.

**Error Type:**

Warning

**Possible Cause:**

1. A start or stop item of a defined trigger expression does not exist in the server configuration or has an invalid address format.

2. The defined log on data change monitor item does not exist in the server configuration or has an invalid address format.

**Possible Solution:**

1. Verify the start and stop items for each trigger configuration.

2. Verify that each trigger configuration's log on data change monitor item address exists in the server configuration.

## Failed to register batch ID '<name>' for log group '<name>'.

**Error Type:**

Warning

**Possible Cause:**

The specified batch ID item does not exist in the server configuration or has an invalid address format.

**Possible Solution:**

Check the batch ID item and verify that it is a valid address.

## Failed to open Store and Forward file '<name>'. Reason: <reason>

**Error Type:**

Warning

**Possible Cause:**

1. The file has already been opened by another process.

2. The configured directory is invalid or Read Only.

**Possible Solution:**

1. Verify that the Store and Forward file is not already open by another process.

2. Verify that the configured directory is valid and contains proper permissions.

## Store and Forward file '<name>' has reached maximum size of <size> MB.

**Error Type:**

Warning

**Possible Cause:**

DataLogger cannot add more data to the store file because doing so would exceed the maximum file size.

**Possible Solution:**

1. Disable the log group and then move or rename the store file. Once complete, re-enable the log group.

2. Increase the value of the Maximum Storage Size property.

## Failure writing to Store and Forward file '<name>'. Reason: <reason>

**Error Type:**

Warning

**Possible Cause:**

1. The hard disk does not have enough space remaining for data to be written to the file.

2. If located remotely, access to the file may have been interrupted.

**Possible Solution:**

1. Increase the available space or verify that the hard disk has adequate space remaining.

2. Verify that the remote file's location is accessible.

## Invalid Store and Forward file '<name>'.

**Error Type:**
Warning

**Possible Cause:**
The specified file located in the configured Store and Forward directory is not a valid store file.

**Possible Solution:**
Move or delete the existing file.

## Store and Forward file '<name>' is incompatible with current log item list.

**Error Type:**
Warning

**Possible Cause:**
The log group's configuration does not match the configuration under which the store file was populated. Either the log item list or the table format has changed.

**Possible Solution:**
The old file is archived in an "_archive" sub-directory of the configured storage directory. To use the file, rename it to "<log group name>.bin" and then move it into the configured storage directory. Once complete, revert the project to the DataLogger configuration with which the file was created.

## Failure reading from Store and Forward file '<name>'. Reason: <reason>

**Error Type:**
Warning

**Possible Cause:**
DataLogger cannot read from the store file. If located remotely, access to the file may have been interrupted.

**Possible Solution:**
Verify that the remote file's location is accessible.

## Storage directory '<name>' for log group '<name>' is invalid. Setting storage directory to '<name>'.

**Error Type:**
Warning

## Unable to access Store and Forward file '<name>' from its current system directory. Please move this file to a non-system directory and update the Store and Forward Storage directory accordingly.

**Error Type:**
Warning

## Log group '<name>' exceeds the item count of <count> for the MySQL DSN using the 'Wide' table format. Logging activity will be rejected until this count is reduced.

**Error Type:**
Warning

**Possible Cause:**
Logging activity is rejected until this count is reduced – The MySQL ODBC driver.

**Possible Solution:**
Use a different DBMS or use multiple log groups (logging to multiple tables) to 'break up' the total number of items to be logged.

## CreateMSSQLTable: <query>

**Error Type:**
Informational

## CreateMySqlTable: <query>

**Error Type:**
Informational

## CreatePostGreSQLTable: <query>

**Error Type:**
Informational

## CreateAccessTable: <query>

**Error Type:**
Informational

## CreateOracleTable: <query>

**Error Type:**
Informational

## Log group '<name>' is connected to DSN '<name>' and is logging data.

**Error Type:**
Informational

## Log group '<name>' is disconnected and is no longer logging data.

**Error Type:**
Informational

### Saving Store and Forward file '<name>' as '<name>'.

**Error Type:**

Informational

### Log group '<name>' has requested creation of at least one TIMESTAMP column for table '<name>'. This has been adjusted to the MS-SQL required DATETIME.

**Error Type:**

Informational

### Created table named '<name>' for log group '<name>' on DSN '<name>'.

**Error Type:**

Informational

### Table name '<name>' exists for log group '<name>' on DSN '<name>', which is in 'create table once and append' mode.

**Error Type:**

Informational

### No table name provided for the database.

**Error Type:**

Informational

### Table validation failed. The table does not exist.

**Error Type:**

Informational

### The deadband range is invalid. The high value must be higher than or equal to the low value.

**Error Type:**

Informational

## Reason Codes

Some event log messages append a "Reason:" code. The reason is the result of a CFileException or an internal exception code of 0x1000. The internal exception code 0x1000 occurs when a partial record is encountered. Since this would corrupt the data, the transaction is discarded. The possible CFileException codes and descriptions are shown in the table below.

| Error | Value | Description |
|---|---|---|
| CFileException::none | 0 | No error occurred |
| CFileException::genericException | 1 | An unspecified error occurred |
| CFileException::fileNotFound | 2 | The file could not be located |
| CFileException::badPath | 3 | All or part of the path is invalid |
| CFileException::tooManyOpenFiles | 4 | The permitted number of open files was exceeded |
| CFileException::accessDenied | 5 | The file could not be accessed |
| CFileException::invalidFile | 6 | There was an attempt to use an invalid file handle |
| CFileException::removeCurrentDir | 7 | The current working directory cannot be removed |
| CFileException::directoryFull | 8 | There are no more directory entries |
| CFileException::badSeek | 9 | There was an error trying to set the file pointer |
| CFileException::hardIO | 10 | There was a hardware error |
| CFileException::sharingViolation | 11 | SHARE.EXE was not loaded or a shared region was locked |
| CFileException::lockViolation | 12 | There was an attempt to lock a region that was already locked |
| CFileException::diskFull | 13 | The disk is full |
| CFileException::endOfFile | 14 | The end of file was reached |

# Index