MORNINGSTAR Wealth

Custodial Integrator – Bulk Insert

Custodial Integrator Version:	3.4
Document Version:	1
Document Issue Date:	October 29, 2013

Technical Support:	(866) 856-4951
Telephone:	(781) 376-0801
Fax:	(781) 376-8040
Web:	byallaccounts.morningstar.com

Table of Contents

About this Guide

Purpose of this guide

This guide describes an optional capability in Custodial Integrator called "Bulk Insert" that enables improved performance of the Import step.

Intended audience

This document is intended for use by firms that intend to use the Custodial Integrator (CI) Bulk Insert.

Overview of Bulk Insert

Bulk Insert is an optional capability for inserting data "in bulk" into the CI database during the CI Import process.

Bulk Insert is intended for firms that have a large number of positions. Bulk Insert can improve the running time of CI's Import step by 50% or more for firms that have 5,000 positions or more to download. There are specific SQL Server requirements as well as additional parameter settings for this feature.

For firms with fewer positions, Bulk Insert is not recommended because the performance improvement may not be significant enough to justify the overhead of meeting the additional requirements. (See Operational Requirements for details.)

Operational Requirements

The following items are requirements for CI Bulk Insert. More details are included in Enabling Bulk Insert.

- **The firm has 5,000 positions or more to download**. While not strictly a requirement, firms with fewer positions to download will not typically see significant CI performance improvements when implementing Bulk Insert.
- **SQL 2005 or later** is required to perform the full range of bulk operations.
- **SQL Server Recovery Model for the CI database set to Simple or Bulk Logged**. The SQL Server Recovery Model affects the efficiency of Bulk Insert. Use either the Simple or Bulk Logged recovery model for the CI database.
- **SQL Server and CI must share access to XML files.** Create a shared folder for the bulk insert data files, and ensure that CI can read, write, and delete files in the folder, and that SQL Server can read them.
- **The CI SQL Server User must have "ADMINISTER BULK OPERATIONS" permission.** When configured to perform Bulk Insert, CI will do so on every CI Import operation. The SQL User that is used by CI (whether by SQL Login or by Windows Authentication) must be granted the server-level permission "ADMINISTER BULK OPERATIONS".
- **Two CI.ini parameters must be set.** The `useBulkInsert` and `bulkInsertDataFolder` parameters must be set in the CI.ini file.

How the CI Import Process Works

This section describes the CI Import process, how Bulk Insert affects the way data is inserted into the database, and the location of the XML data files.

Import operations

The CI import process downloads financial data gathered by the ByAllAccounts server every day. It then inserts the downloaded data into the CI database. The CI database tables that hold daily

financial data are transient; at the beginning of each CI run these tables are emptied of their contents and then populated again with fresh data brought in from the ByAllAccounts service.

The following CI database tables store transient financial data:

`baaSecurity`	Contains imported securities.
`baaSecurityHx`	Contains security prices for the previous business day and for additional dates if requested.
`baaHolding`	Contains imported positions.
`baaHoldingHx`	Contains position prices for the previous business day and for additional dates if requested.
`baaTransaction`	Contains imported transactions.
`baaHoldingLot`	Contains imported position lots.

XML data files

Financial data is sent from the ByAllAccounts server to CI in XML format. As soon as the data is received, CI writes it to a temporary XML file. The differences between Bulk Insert and single-row are in the mechanism used to extract the data from the temporary XML file and insert it to the appropriate database tables, and in the location of the XML data files.

XML insert process

- Single-row insert: Single-row insert refers to the mechanism by which insertion into the database is done at the row level (one SQL INSERT operation for each table row). When using single-row insert, CI parses each element (security, position, transaction, etc.) in the XML data file and then inserts it to the appropriate table using the simple SQL INSERT statement. Single-row insert is the default CI behavior.
- Bulk Insert: Bulk Insert refers to the mechanism by which insertion into the database is done in bulk. When using Bulk Insert, CI does not directly interpret the XML data file. Instead CI passes a pointer to the XML file to the SQL server together with a series of BULK INSERT statements. SQL server directly reads the data from the XML file and inserts the records in bulk.

Location of XML data files

- Single-row insert: Single-row insert mode is the default CI behavior. All response data files are written to the CI working folder configured by the user (`<working>\imported`).
- Bulk Insert: When in bulk-insert mode, the `<working>\imported` folder is still aused to store other imported data such as transaction translations, position/transaction filters, and descriptive data of Financial Institutions and accounts. However, CI will write the financial data files directly to a separate shared folder reserved for temporary storage of financial data. This special shared folder is configured by the user and must be accessible both to CI and to the SQL server. See details about the folder and its configuration in the Enabling Bulk Insert section.

Note: Regardless of the insert mode and data location, when CI is running in debug mode, the response data files are not automatically deleted from their folder. When using Bulk Insert with debug turned on, the files will be kept in the folder specified by the `bulkInsertDataFolder` parameter until the user manually deletes them.

ENABLING BULK INSERT

Use these instructions to enable Bulk Insert at your firm. The requirements are summarized in Operational Requirements, and spelled out in greater detail here.

1. **Ensure SQL 2005 or later is installed.**
2. **Ensure the SQL Server Recovery Model is either Simple or Bulk Logged.**

   In SQL Server a recovery model is specified for each database, and the recovery model affects the efficiency of Bulk Insert. To enable CI's Bulk Insert to work quickly, use either the Simple or Bulk Logged recovery model for the CI database.

   Note: If you were previously using the Full recovery model on the CI database, you must switch the database to the Bulk Logged recovery model.

   For descriptions of recovery models see:
   http://msdn.microsoft.com/en-us/library/ms189275%28v=sql.105%29.aspx

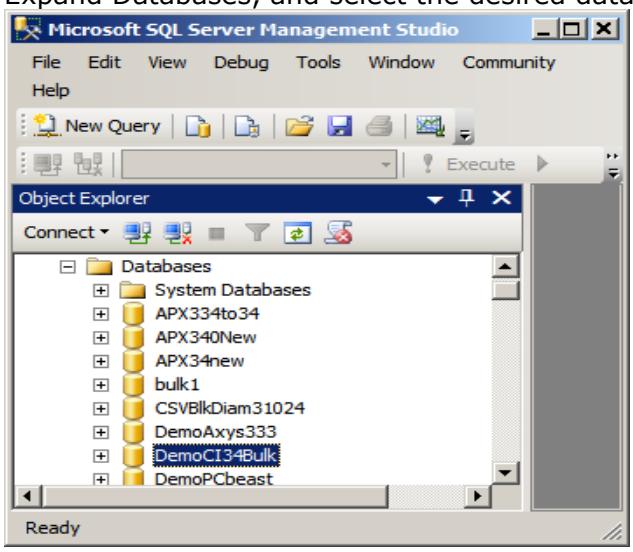
   Database backups
   The CI database holds very little persistent data, including things like CI configuration, account mappings, and security mappings. This persistent data is typically changed only when CI runs and performs periodic download. In a typical use model, CI is run once per day, or at most a small number of times per day.

   How to change the recovery model
   Use these steps if you need to view or change the recovery model of a database. Note that you need ALTER permission on the database. These instructions are based on Microsoft SQL Server 2008 R2.

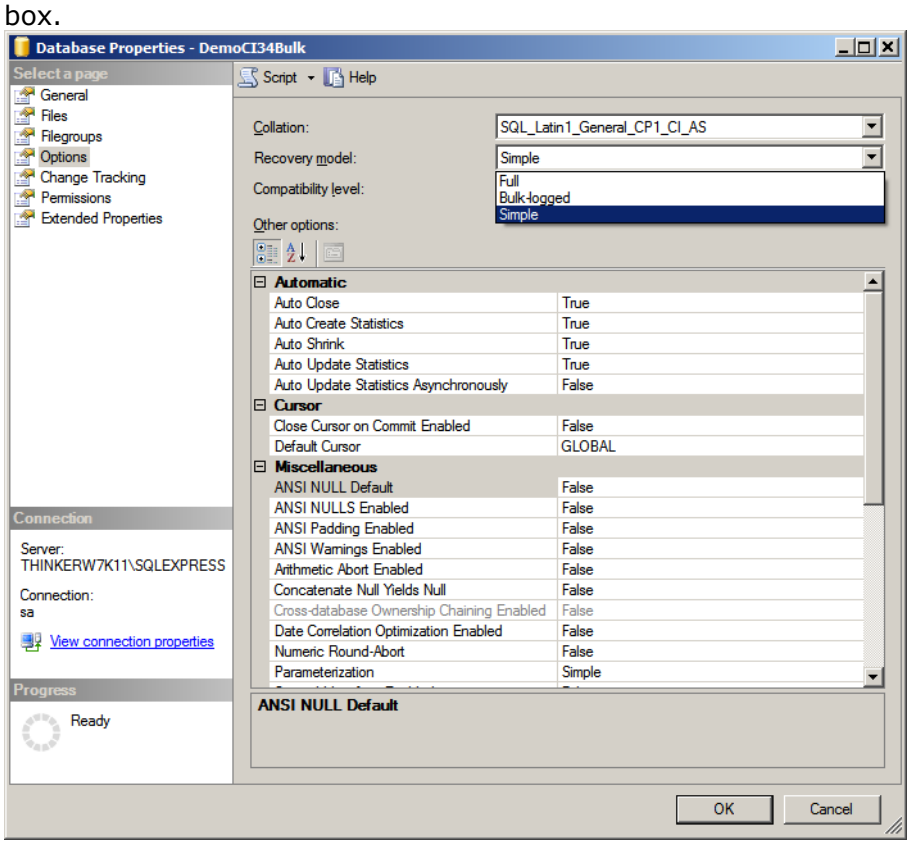
   Note: Refer to your SQL Server documentation for instructions before changing the recovery model, and back up your database before making any changes.

   a) From Microsoft SQL Server Management Studio, Object Explorer connect to the appropriate instance of the Microsoft SQL Server Database Engine, and then click the server name to expand the server tree.

   b) Expand Databases, and select the desired database.



c) Right-click the database, then click **Properties**, which opens the Database Properties dialog box.



d) In the Select a Page pane, click **Options**.
The current recovery model is displayed in the Recovery model list box.

e) To change the recovery model select a different model from the list. Select **Bulk Logged** or **Simple**.

3. Create the bulk insert data folder.

CI must be able to write the XML files to a folder that the SQL Server process can access. This folder can be on the CI computer, the SQL Server computer, or a third computer. It must be a shared folder and made accessible to both the CI computer and the SQL Server computer.

The `bulkInsertDataFolder` parameter in the CI.INI file specifies the absolute path of this "shared" folder. CI will write out the XML file to disk and so it is recommended that you enable optimal performance of this operation, which may mean that the folder to which the XML file is to be written is on a storage device that is local to the CI computer. We recommend that you evaluate your computing environment to determine the best performing location for this folder.

4. Ensure that CI has permission to read, write, and delete in the bulk insert data folder.

The CI application writes the XML response file to this folder, therefore the CI user must have Read and Write access to the bulk insert data folder and its files.

5. **Ensure that SQL Server has read permission for the XML files in the bulk insert data folder.**

   The SQL Server service reads the XML file in the bulk insert data folder, so it must have Read access to the files in the folder. The SQL User that is used in CI must have read access to the files in the XML data files folder. CI can authenticate to SQL Server using one of two modes, and the mode used affects how SQL server will attempt to access the XML files, which in turn dictates the settings you need to adjust.

   - If CI authenticates to SQL Server using *Windows Authentication*, then SQL Server will access the XML file via impersonation of the Windows user who is running CI. Therefore, when Windows Authentication is used, the Windows account of the user running CI needs Read permission to the folder. The access permissions for the CI applications are also usually good for the SQL server access to the folder. Ensure that Windows impersonation/delegation settings are configured appropriately to enable access to the folder containing the XML file
   - If CI authenticates to SQL Server using *SQL Server Authentication* (using a SQL Login such as **sa**), then the SQL Server will access the XML file as the Windows user that is configured to run the SQL Server service. Folder permissions for the bulk insert data folder must allow the SQL Server service user to read the XML file. The permission is implicit if SQL Server runs under one of the privileged built-in accounts: *Network Service* or *Local System*, which is usually the case. However, an additional permission may have to be granted if SQL server service runs under a domain user account that is different than the domain user account under which CI runs.

6. **Ensure that CI SQL User has "ADMINISTER BULK OPERATIONS" permissions.**

   When configured to perform Bulk Insert, CI will do so on every CI Import operation. The SQL User that is used by CI, whether by SQL Login or by Windows Authentication, must be granted the server level permission "ADMINISTER BULK OPERATIONS".

7. **Stop CI before editing the CI.ini file.**

8. **Set the two Bulk Insert parameters in CI.ini file.**

   Two CI.ini parameters manage Bulk Insert behavior:

   - `useBulkInsert=y`
     Defaults to 'n', which leaves the SQL insert mechanism as it is. Setting this parameter to 'y' enables Bulk Insert. For this parameter to work, the `bulkInsertDataFolder` parameter also must be set.
   - `bulkInsertDataFolder=`*<full pathname of the shared bulk-inset data folder>*
     Specifies the folder where CI will write the XML financial data files when running in Bulk Insert mode. There is no default. This parameter must be set for the `useBulkInsert=y` parameter to work, and is ignored when `useBulkInsert=n` or is not set.

     The folder must be specified as an absolute pathname in Universal Naming Convention (UNC) format, and must point to a network shared folder, as in:
     \\<*MachineName*>\<*shared folder name*>\[<*bulk insert data folder name*>]

     Note that the `bulkInsertDataFolder` parameter is set in addition to the CI working folder and is used for temporary storage of data when using Bulk Insert. See step 3 and Location of XML data files for more information.

     The Windows user running CI must have read/write/create/delete access to the folder and its files, and the SQL user used in CI must have read access to the files in the folder..

9. **Start CI.**

   During initialization, CI will verify that the `bulkInsertDataFolder` parameter specifies the absolute path to a shared folder in UNC format, that the specified folder exists, that the current

Windows user running CI has Read, Write, Create, and Delete access to the files in the folder and has been granted the sever-level permission Administer Bulk Operations.
If this validation fails, CI will exit after displaying an error message. The message and additional error details are logged in the CI log file. For log information, refer to CI Logging when using Bulk Insert, page 6.

Disabling Bulk Insert

When Bulk Insert is enabled, it can be disabled at any time by setting `useBulkInsert=n` in the CI.ini file and restarting CI. No other changes are needed.

CI LOGGING WHEN USING BULK INSERT

When in debug mode, CI writes all row-insert transactions to the CI log. When Bulk Insert is used, that granular information will not be available in the CI log.

Exceptions generated by SQL server during Bulk Insert will still be logged as fatal errors and will trigger a popup error message on the CI screen. The exceptions generated from Bulk Insert contain information about the type of error, the table where the insert error occurred, and other detailed information depending on the specific exception. For example, it would show the primary key that is in violation of the unique key constraint.

Case of successful validation

If all validations of the configured bulk-insert-data folder pass successfully, then CI logs an information message about the location of the bulk-insert data folder. See following CI log excerpt (in bold):

```
10-07 21:50:47.471: AWT-EventQueue-0: CI CSV V3.4.000 started with db conn:
'jdbc:sqlserver://localhost\SQLEXPRESS;DatabaseName=tmpCSV34JacobusBulkErrtest ;, log:1'
10-07 21:50:47.471: AWT-EventQueue-0: Running on Windows Vista, Java 1.6.0_13-b03
10-07 21:50:47.471: AWT-EventQueue-0: Running in time zone America/New_York
10-07 21:50:48.204: AWT-EventQueue-0: Database schema version: 51
10-07 21:50:48.360: AWT-EventQueue-0: Bulk-Insert data folder set to:
\\thinkerw7k11\bulkinsertdata1
10-07 21:50:48.610: AWT-EventQueue-0: Requesting TX translations from DC
10-07 21:50:49.530: AWT-EventQueue-0: Received requested TX translations.
```

Cases of validation failure

This describes some possible failure cases and their related error messages.

Error Message	Possible Cause
Invalid parameter value: 'bulkInsertDataFolder=null' Parameter 'bulkInsetDataFolder' must be set when using bulk insert.	The bulkInsertDataFolder parameter is not specified or specified with an empty value
Invalid parameter value: 'bulkInsertDataFolder=c:\<directory>' Must denote the absolute path of an existing shared folder and be specified in UNC format: \\<machine name>\<shared folder name>[\<data folder>].	The bulkInsertDataFolder is specified but not correctly

Invalid parameter value: 'bulkInsertDataFolder=<bad path>' Must denote the absolute path of an existing shared folder and be specified in UNC format: \\<machine name>\<shared folder name>[\<data folder>].	The bulkInsertDataFolder is specified but not in UNC format. It might have been set as a relative path or as path to a folder on a drive.
Invalid parameter value: 'bulkInsertDataFolder=<bad path>' The specified folder does not exist.	The pathname is not in UNC format and it does not exist. The check for existence is run first.
Invalid parameter value: 'bulkInsertDataFolder=<good or bad path>' The specified folder does not exist.	The check for whether the file exits happens first. This error message shows that the folder does not exist. The path might be correctly specified (absolute pathname in UNC), but it could also be incorrect.
Invalid parameter value: 'bulkInsertDataFolder=<good path>' You don't have permission to read files in this folder.	CI user does not have the required permissions to the folder The pathname is specified correctly and points to an existing shared folder. However the Windows user running CI does not have all read permissions to the folder.
Invalid parameter value: 'bulkInsertDataFolder=<good or bad path>' You don't have permission to create/write files in this folder.	CI user does not have the required permissions to the folder The pathname is specified correctly and points to an existing shared folder. However the Windows user running CI does not have all the create/write permissions to the folder.

Unable to bulk read test file 'text.hml' in folder: \\<machine name>\<shared folder name>[\<data folder>].
 	
The current CI SQL user may have no permission to read files or perform bulk operations in this folder.
	There may be one of two causes for this error, each with a different log message.
	
	1) The SQL Server Service is denied access to the folder. The CI log contains the full SQLServerException that was thrown. This example shows a excerpt from a log with the SQLServerException:
	
	`10-08 13:02:30.840: AWT-EventQueue-0: Error: An internal error occurred. Info: select CONVERT(xml,BulkColumn, 2) FROM OPENROWSET(BULK '\\win7test\bulkinsertdata\test.xml', SINGLE_BLOB) as BulkColumn`
	`10-08 13:02:30.840: AWT-EventQueue-0:` **`com.microsoft.sqlserver.jdbc.SQLServerException: Cannot bulk load because the file "\\win7test\bulkinsertdata\test.xml" could not be opened.`**
	
	2) CI SQL user has not been granted the "bulkadmin" server role. Log excerpt with SQLServerException:
	
	`10-08 13:26:39.605: AWT-EventQueue-0: Error: An internal error occurred. Info: select CONVERT(xml,BulkColumn, 2) FROM OPENROWSET(BULK '\\thinkerw7k11\bulkinsertdata1\test.xml', SINGLE_BLOB) as BulkColumn`
	
	`10-08 13:26:39.605: AWT-EventQueue-0:` **`com.microsoft.sqlserver.jdbc.SQLServerException: You do not have permission to use the bulk load statement.`**

CI AUTOMATION AND BULK INSERT

Bulk Insert requirements, configuration, and behavior are identical when CI is run in interactive mode and when it is run in automation mode.