

# CGI Advantage<sup>®</sup> 4

---

## System Processing User Guide



This guide contains information proprietary to CGI Technologies and Solutions Inc. Unauthorized reproduction or disclosure of this information in whole or in part is prohibited.

CGI Advantage is a registered trademark of CGI Technologies and Solutions Inc.

Due to the nature of this material, numerous hardware and software products are mentioned by name. In most, if not all, cases, the companies that manufacture the products claim these product names as trademarks. It is not our intention to claim these names or trademarks as our own.

Copyright © 2001, 2024, CGI Technologies and Solutions Inc. All Rights Reserved.

## Table of Contents

System Processing Overview .....	7
End User Processing .....	8
Compose Email Message.....	8
Transaction Schedule.....	8
Job Manager .....	10
Job Summary .....	10
Schedule New Job.....	11
Job Parameters .....	12
Job Steps for New Chain Job.....	12
Setup Custom Parameters .....	13
Inquiries.....	14
Archive Facilitator Inquiry .....	14
Chain Jobs List .....	14
Chain Job Steps .....	15
Edit Job.....	15
Job Inquiry .....	16
Job Log Entries.....	16
Pending Jobs.....	17
Print Facilitator Inquiry.....	17
Reports .....	17
Reports and Forms.....	17
View Forms.....	18
Advanced - Setup.....	19
Application Parameter .....	20
Application Servers.....	20
Assembly Configuration.....	20
Batch Interface Event .....	21
Encrypted Batch Parameters.....	21

File Transfer .....	22
File Transfer Application .....	22
Tenant .....	23
Application URL Mapping .....	23
File Transfer Folder .....	23
File Transfer Role .....	24
File Transfer Maintenance .....	25
File Transfer Audit Log .....	27
Form Definition .....	28
Form Transformation Definition .....	28
Form Transformation Group Order .....	28
Form Transformation Group Relationships .....	29
Forms Access Control .....	29
Forms Printing Access Control .....	29
Forms Server Definition .....	30
Interface Request Configuration .....	30
Job Server Control .....	31
Job Updater .....	32
Job Updater Parameter Values .....	34
Manage Reports and Forms Roles .....	34
Messages .....	35
Print Form Transformation Fields .....	35
Print Job Setup .....	36
Print Resource Setup .....	36
Print Server Setup .....	36
Setup Audit Log and External Notification .....	36
Setup Batch Job .....	37
System Level Process Parameters .....	38
Catalog Folder Details .....	38
Item Details .....	39

Chain Job .....	40
Setup for Chain Job Steps .....	41
Edit Catalog Entry .....	41
Site Specific Parameters .....	42
System Processes .....	42
Associating Batch Process with System Process .....	43
Transaction Print Action .....	44
Advanced - Batch Processing .....	45
Batch Jobs .....	45
Working with System Maintenance and Utility .....	47
Maintenance of Tables .....	48
Maintenance of Transactions .....	49
Setting up Parameters .....	52
Input Parameter File .....	52
Job Framework .....	55
Custom Parameter Screen .....	55
System Maintenance Utility Parameters .....	55
Exporting Records from a Table .....	56
Insert Records in a Table .....	59
Update Records in a Table .....	62
Overlay Records in a Table .....	64
Overlay Records in a Table from CSV .....	66
Delete Records from a Table .....	68
Purge a Table .....	70
Archiving Transactions .....	71
Archiving Historical Transactions .....	73
Unarchiving Transactions .....	74
Exporting Transactions .....	75
Transaction Import .....	78
Performing Transaction Custom Actions .....	82

Mark Transactions for Processing.....	84
Hold Transactions from Processing .....	86
Discarding Transactions.....	89
Editing Transactions.....	92
Activating Transactions .....	94
Deactivating Transactions .....	96
Submitting Transactions.....	98
Validating Transactions.....	102
Printing Transactions.....	105
Rejecting All Pending Transactions .....	108
Approve Transaction .....	111
System Maintenance Utility Listener.....	113
Listener.....	113
Application Parameter .....	113
ADV Job Interaction Client.....	115
Setup .....	115
Driver Data File.....	116
Control Block.....	118
Control Type: Properties .....	118
Control Type: Action.....	119
Control Type: Variable.....	123
Job Block .....	123
Chain Job Block .....	125
Logging Configuration File.....	127
Suggested Usage .....	127
DAT Files .....	127
JIC Execution.....	127
Running JIC Jobs as a Specific User.....	128
Application Stop/Start .....	129

## System Processing Overview

The System Processing area of Advantage includes setup and inquiries related to various types of system processing.

## End User Processing

Most associate system processing with just batch, chain, and report jobs. However, there are other forms of system processing that are on demand by end users such as generating a form, sending an email, and setting up a transaction for later processing. Triggering these events is done through the transaction menu with the corresponding action.

### Compose Email Message

The **Send Page** action is used to define information for the system to send an email to one or more email addresses with reference to a transaction. Recipients can be entered manually or by using the pick available to system users. The email could just be a 'for your information' or a request to complete, apply an override, any other system action, or for research purposes.

### Transaction Schedule

The **Schedule** action is used to define an action for the system to take on a transaction in the future such as submitting it on a future date. Choosing the **Add** icon creates a record for you with all the transaction information pre-populated and the Schedule Date/Time set to the current date and time. The default allows you to know the correct format for both components of that field.

The Action field has to then be selected with the action you wish the system to take on the transaction. As a user scheduling one of these actions, you must have authority to perform that action or the batch processes created for the scheduled event will fail. Also, there are system rules about when the actions can be taken, so pay careful attention to the Description.

Action	Description
Deactivate	Not a likely choice, but will deactivate an active transaction with a Transaction Phase of <i>Final</i> at a future date so that it cannot be cancelled, modified or referenced
Activate	Not a likely choice, but will activate a deactivated transaction with a Transaction Phase of <i>Final</i> at a future date so that is can be cancelled, modified or referenced.
Archive	If all transaction-specific rules for archiving have been met, this action will archive the transaction as long as it has a Transaction Phase of <i>Final</i> or <i>Historical Final</i> .
Unarchive	Not applicable when scheduling a transaction.
Edit	This choice will create a modification draft on a future date if the Transaction Phase is <i>Final</i> and the type of transaction allows modifications.



Discard	Not a likely choice, but will discard the transaction, if the Transaction Phase is <i>Draft</i> or create a modification draft if the Transaction Phase is <i>Final</i> .
Validate	This choice will fire all system edits at a future date for a transaction with the Transaction Phase of <i>Draft</i> .
Submit	This choice will fire all system edits and attempt to submit a transaction with a Transaction Phase of <i>Draft</i> to either <i>Pending</i> or <i>Final</i> , depending on workflow configuration.
Mark Transaction Ready	This choice will set the Transaction Status to <i>Ready</i> for a transaction with the Transaction Phase of <i>Draft</i> so that any system process configured to attempt to submit all draft and ready transactions will select the transaction.
Hold Transaction	This choice will set the Transaction Status to <i>Held</i> for a transaction with the Transaction Phase of <i>Draft</i> so that any system process configured to attempt to submit all draft and ready transactions will not select the transaction. This is not a likely choice to schedule but one to be done immediately with the <b>Mark Hold</b> action found in the transaction menu.
Other Action	Not a likely choice, but is used when a transaction custom action is desired instead of the common transaction items discussed above. That custom action is specified in the Sub Action field.

## Job Manager

The Job Manager (also known as Run Batch Job) lists all of the batch, chain, and report jobs available in the Advantage application. You can use the search fields to narrow the list by Application, Catalog Label, or by the Item Type. The Item Type determines the row-level action.

› Row-level actions

- **Open** - This action is displayed if the Item Type is *Folder*. Selecting this actions transitions you to a page that lists the contents of the folder, which can include additional folders or a list of jobs included in the selected folder.
- **View** - This action is displayed if the Item Type is *System Batch*, *Chain*, or *Report*. Selecting this action transitions you to a [summary page](#) for the selected batch, chain, or report job.

## Job Summary

A Job Summary page exists for batch, chain, and report jobs. This page is accessed after selecting **View** from the row-level menu of a batch, chain or report job on the Job Manager page.

› Page-level actions

- **Schedule New Job** - This action is available for batch jobs and report jobs. This action transitions you to the [New Job](#) page, which allows you to schedule the batch/report job.
- **Schedule New Chain Job** - This action is available for chain jobs. This action transitions you to the [Job Steps for New Chain Job](#) page, which allows you to disable steps, if needed, before scheduling the chain job.
- **View Pending Jobs** - This action is available for batch and report job. This action transitions you to the [Pending Jobs](#) page for the selected batch or report job, which allows you to view/edit any jobs for the associated batch or report job that are currently pending in Advantage.
- **View Pending Chain Jobs** - This action is available for chain jobs. This action transitions you to the [Pending Chain Jobs](#) page for the selected chain job, which allows you to view/edit any jobs for the associated chain job that are currently pending in Advantage.
- **Restart Job** - This action allows you to restart a job that can be restarted, based on the job's characteristics and run status.

› Row-level actions

- **View Job Steps** - This Related Page action is available for chain jobs and transitions you to a [Chain Job Steps](#) page, which lists each job included in the chain job for the selected Job ID, and includes additional information about each job, such as the Return Code.
- **View Log** - This Related Page action is available for batch and report jobs and transitions you to the [Job Log Entries](#) page for the selected Job ID.

- **Report(s)** - This Related Page action is available for report jobs and transitions you to the [Reports](#) page, which lists any reports that were generated for the selected Job ID.
- **Restart** - This action allows you to restart a job that can be restarted, based on the job's characteristics and run status.

## Schedule New Job

The New Job and New Chain Job pages allow you to enter required information and schedule the new batch, chain, or report job. Selecting Save on this page enables the page-level menu, which includes a link to set up specific parameters for the batch, chain or report job. After required fields on this page have been populated and all parameters have been specified via the Job Parameters page, select **Submit Request**.

› Field Information

Field Name	Description
Job ID	System maintained. Displays the job identifier.
Job	System maintained. Displays the Catalog name for the job.
Job Name	Specifies the name for this job. User can provide a name for this particular run.
Run Option	Indicates the frequency for this run. The available options are: Once, Run Immediate, Hourly, Weekly, Daily, Monthly, and Custom. If the Run Option is not Run Immediate, a valid schedule time (field name-Schedule Time) and expire time (field name- Don't Schedule After) should be provided. If the option, Custom, is selected an additional field, Run Interval (minutes) appears on the screen.
Application Server ID	Identifies the server on which to run this job. You can use the pick list to see a list of the available servers.
Application Server Name	System maintained. Indicates the name of the server you specified.
Scheduled Date/Time	Specifies the date and time to run this job.
Don't Schedule After	Specifies the date and time after which the job expires.

Number of Runs	Indicates the number of iterations of the job you are requesting to be executed. If left blank, the job will run (if a periodic job) as long as the job does not expire.
Pre-Condition Return Code	Indicates the condition that must be met by the previous job before this job can be run. You choose from the pick list: Successful, Failed, Terminated, System Failure, Warning, or non-fatal error.
Run After Completing Job	Indicates the job that must be completed before the current job being scheduled can be run. You can use the pick list to see a list of available jobs to choose from (jobs already scheduled).
Viewable by All Users?	This flag is only available for report jobs. When checked the rendered report instance is viewable by all users.

› Page-Level Action

- **Setup Parameters** - This link transitions you to the [Job Parameters](#) page for the selected batch or report job.
- **Setup Parameters for Job Steps** - This link transitions you to the [Job Steps](#) page for the selected chain job.

## Job Parameters

The Job Parameters page lists all parameters for the selected batch or report job. After entering the required information for each of the parameters, select **Save & Close**, which returns you to the [New Job](#) page. After entering the required information on the New Job page, select **Submit Request**.

If you are scheduling a chain job, then this page lists each job in the chain along with an **Edit Job Parameters** row-level action that transitions you to the [Job Parameters](#) page for the selected job. Select **Save & Close**, which returns you to the [Job Steps](#) page for the chain job. Continue selecting the Edit Job Parameters row-level action for each job in the chain until you have verified/specific all parameter information. Select the **Back** button on the Job Steps page to return to the [New Chain Job](#) page. After entering the required information on the New Chain Job page, select **Submit Request**.

The **Setup Custom Parameters** page-level action is provided on the Job Parameters page, if the job includes custom parameters. The action transitions you to the [Setup Custom Parameters](#) page.

Refer to the associated run sheet for details on the parameters provided with each batch, chain, and report job. Run Sheet guides can be accessed/downloaded from the "User Guide Downloads" topic file in the online help.

## Job Steps for New Chain Job

The Job Steps for New Chain Job page allows you to disable any jobs that you do not want included in the current run of the chain job. You can disable a job by selecting *No* for the corresponding Disable Job

field. You are transitioned to this page after selecting Schedule New Chain Job from the page-level menu on the [Chain Job Summary](#) page.

› Page-level actions

- **Continue Scheduling** - This action transitions you to the [New Chain Job](#) page, which allows you to schedule the chain job with the jobs that are not disabled.

## Setup Custom Parameters

Some jobs in Advantage include custom parameters that can be set up, in addition to the standard set of parameters that are provided with the job. The **Setup Custom Parameters** page-level action is provided on the Job Parameters page, if the job includes custom parameters. Refer to the associated run sheet for details on the custom parameters provided with each a job. Run Sheet guides can be accessed/downloaded from the "User Guide Downloads" topic file in the online help.

## Inquiries

The System Processing area of Advantage includes the following inquiries (listed alphabetically):

- [Archive Facilitator Inquiry](#)
- [Chain Jobs List](#)
- [Chain Job Steps](#)
- [Edit Job](#)
- [Job Inquiry](#)
- [Job Log Entries](#)
- [Pending Jobs](#)
- [Print Facilitator Inquiry](#)
- [Reports](#)
- [Reports and Forms](#)
- [View Forms](#)

### Archive Facilitator Inquiry

The Archive Facilitator Inquiry allows users to view the status of archiving and restoring processes run by the Facilitator. The Facilitator is a process that allows archiving/restoring of data, be it for Checks, Leave, or transactions, to be executed as a multi-step process chained together such that users only have to kick-off the first job in the series.

When the Facilitator job is executed, this page is updated with a status of *Transaction Ready for Archiving*, *Transaction Archiving*, *Transaction Archiving History*, *Transaction Unarchiving*, *Transaction Archive Complete*, *Table Ready for Archiving*, *Table Archiving*, *Table Restoring*, or *Table Archive Complete*, depending upon the action specified in the Facilitator parameter file. The inquiry excludes all records that have a Status of *Transaction Ready For Printing*, *Transaction Printing*, or *Transactions Printed*.

- › Row-level action
  - **View Log** - This action transitions you to the [Job Log Entries](#) page

### Chain Jobs List

This page lists all jobs that have been executed as part of the chain job that was selected on the [Job Inquiry](#) page.

- › Field Information

Each job displays the following fields (listed below alphabetically):

- **Catalog ID** – The ID of the catalog for the job submitted.
  - **Catalog Name** – The name of the catalog for the job submitted.
  - **End Time** – The time at which the job finished the run if complete.
  - **Item Type** – The type of the job submitted, that is, whether it is a report, system batch or a chain job.
  - **Job ID** – The ID of the job submitted.
  - **Job Name** – The name of the job submitted.
  - **Return Code** - This field indicates the Return Code of the job, for example, Successful, Failed, System Failure, or Warning. Refer to the associated run sheet for the job for additional details about the Return Code for the selected job.
  - **Run Status** – The current status of the job.
  - **Start Time** – The time at which the job started to run if complete.
  - **User ID** – The User ID that has submitted the job.
- › Row-level Related Page
- **View Log** - This link transitions you to the [Job Log Entries](#) page, which allows you to view all messages logged during execution of the job.
- › Row-level Related Action
- **Verify Job Running** - This action verifies whether the job instance is in fact still running or the status was incorrect because of a system event stopping the job.

## Chain Job Steps

This page is accessed by selecting View Job Steps from the row-level menu for a previously executed job on the [Chain Job Summary](#) page.

- › Row-level actions
- **View Log** - This Related Page action is available for each job that was executed as part of the selected chain and transitions you to the [Job Log Entries](#) page for the selected Job ID.
  - **Restart** - This action allows you to restart a job that can be restarted, based on the job's characteristics and run status.

## Edit Job

The Edit Job page allows you to edit information for the selected Job ID, which was previously provided when scheduling the job. Refer to the "[Schedule New Job](#)" topic for information about the fields on this page. You can edit parameters for the job by selecting the **Edit Parameters** page-level action, which transition you to the [Job Parameters](#) page.

The Edit Job page is accessed by selecting Edit Job from the row-level menu on the [Pending Jobs](#) page. This action is only applicable for jobs that have not started running.

## Job Inquiry

This page lists all jobs in the system. Use this inquiry page to search for and view all jobs by any user.

### > Field Information

Each job displays the following fields (listed below alphabetically):

- **Catalog ID** – The ID of the catalog for the job submitted.
- **Catalog Name** – The name of the catalog for the job submitted.
- **End Time** – The time at which the job finished the run if complete.
- **Item Type** – The type of the job submitted, that is, whether it is a report, system batch or a chain job.
- **Job ID** – The ID of the job submitted.
- **Job Name** – The name of the job submitted.
- **Return Code** - This field indicates the Return Code of the job, for example, Successful, Failed, System Failure, or Warning. Refer to the associated run sheet for the job for additional details about the Return Code for the selected job.
- **Run Status** – The current status of the job.
- **Start Time** – The time at which the job started to run if complete.
- **User ID** – The User ID that has submitted the job.

### > Row-level Related Page

- **View Log** - This link is available for batch jobs. This link transitions you to the [Job Log Entries](#) page, which allows you to view all messages logged during execution of the job.
- **View Job Steps** - This link is available for chain jobs. This link transitions you to the Chain Jobs List page, which allows you to view all jobs in the chain and provides links to the job log for each job in the chain. You can also verify if a job is running.

### > Row-level Related Action

- **Verify Job Running** - This action verifies whether the job instance is in fact still running or the status was incorrect because of a system event stopping the job.

## Job Log Entries

The Job Log Entries page lists all messages that are logged by the batch or report job. The Job Log Entries page is accessed by selecting View Log from the row-level menu on the [Job Summary](#) page for



batch and report jobs. The Job Log Entries page can also be accessed by selecting View Log from the row-level menu on the [Job Steps for New Chain Job](#) page. Select Back to return to the previous page.

## Pending Jobs

The Pending Jobs page allows you to view/edit any jobs for the associated batch, chain, or report job that are currently pending in Advantage.

### > Row-level Related Pages

- **Edit Job** - This action transitions you to the [Edit Job](#) page for the selected Job ID. This action is only applicable for jobs that have not started running.
- **View Log** - This action transitions you to the [Job Log Entries](#) page for the selected Job ID.

### > Row-level Related Actions

- **Delete Job** - Removes the job from the queue before the Job Manager picks it up. Jobs picked up to be run cannot be deleted.
- **Approve Job** - Approves a job in pending approval status. This can only be done by users with appropriate access (for example, users assigned to the BATCHADM security role).
- **Submit Job** - Submits a job to the Job Manager for processing. Only applicable for jobs that have a *To Be Submitted* Run Status.
- **Kill Job** - Stops the processing of a job that has already been picked up by the Job Manager (note: the job may not stop immediately).
- **Verify Job Running** - Verifies whether the job is already assigned to a job manager.

## Print Facilitator Inquiry

The Print Facilitator Inquiry provides a filtered view of the Facilitator table. All records on the Facilitator table that have a status of *Transaction Ready For Printing*, *Transaction Printing* or *Transaction Printed* are displayed.

## Reports

The Reports page lists all reports generated by the selected Job ID. You can view the report in PDF format by selecting the **View PDF** row-level action. You can return to the [Job Summary](#) page by selecting the **Back** button.

## Reports and Forms

The Reports and Forms page allows authorized users to view forms and reports based on their assigned security role. The [Manage Reports and Forms Roles](#) setup page determines which reports and forms a security role can view.

> Field Information

Field Name	Description
Job ID	The ID of the submitted job.
Catalog Name	Displays the Catalog name for the job.
Job Name	Specifies the user provided name for the job.
User Id	Displays the User ID of the user who submitted the job.
End Date	The date and time when the job was ended.
File Name	Name of the report/form pdf file.

## View Forms

The View Forms page allows users to easily access PDF forms generated by the Advantage Forms Printing Solution from within CGI Advantage. This is a good means to preview a print or to create a duplicate print. Forms may be printed to the View Forms page from either online or batch printing.

Forms appear on this page, only if both of the following are true:

- The form was printed using a PDF output
- The user selected the View Forms check box before printing.

Printed forms will stay on this page until the Online Forms Cleanup process purges them. The process purges forms that have been accessed or have expired.

## Advanced - Setup

The System Processing area of Advantage includes setup on the following pages (listed alphabetically):

- [Application Parameter](#)
- [Application Servers](#)
- [Assembly Configuration](#)
- [Batch Interface Event](#)
- [Encrypted Batch Parameters](#)
- [File Transfer](#)
- [Form Definition](#)
- [Form Transformation Definition](#)
- [Form Transformation Group Order](#)
- [Form Transformation Group Relationships](#)
- [Forms Access Control](#)
- [Forms Printing Access Control](#)
- [Forms Server Definition](#)
- [Interface Request Configuration](#)
- [Job Server Control](#)
- [Job Updater](#)
- [Job Updater Parameter Values](#)
- [Manage Reports and Forms Roles](#)
- [Messages](#)
- [Print Form Transformation Fields](#)
- [Print Job Setup](#)
- [Print Resource Setup](#)
- [Print Server Setup](#)
- [Setup Audit Log and External Notification](#)
- [Setup Batch Job](#)
- [Site Specific Parameters](#)

- [System Processes](#)
- [Transaction Print Action](#)

## Application Parameter

Application Parameter (APPCTRL) is one that is part of general system configuration. Options are set once and do not vary by year, department, transaction, or other factors. The page is very generic in its design so that it can host many different types of options with the Parameter Name and Parameter Value fields. Details on the various parameters are given in various user guides where the parameter controls system processing.

Field Name	Description
Parameter Name	The name, often in the form resembling a database attribute name, is the name by which application logic knows the parameter. For this reason, values in this field should not be changed.
Parameter Short Description	A more descriptive name given to explain the often abbreviated Parameter Name.
Parameter Value	The setting for a parameter expressed in various formats, depending on the type of parameter. For valid values see the Parameter Description field or further system documentation on the parameter for more details.
Parameter Description	A long description of what functionality the parameter provides along with information on valid values.

Note: Any changes to records should be followed by a restart of all servers used for CGI Advantage applications. Records delivered should never be deleted, even if the functionality they control is not used.

## Application Servers

The Application Servers page is used to setup and maintain the list of application servers in the system. A record will be maintained in this page for each application server with the relevant information such as Application Server Host Name, Job Manager Name, Port, and so forth.

## Assembly Configuration

All CGI Advantage transactions that support the Assembly Process must first be defined and configured on the Assembly Configuration (ASCG) table. This table is only accessible from the CGI Advantage Administration Application.

### Configuration Points:

- The SO, PO, and MA Transaction Types support the Assembly Process. For each transaction code, the system utilizes existing print setup tables to define the Print Resource, Application Resource, and Print Job for the Assembly Process. The Assembly Process references this table for information required to execute to the job.
- In addition, the Assembly Configuration table allows you to indicate the maximum wait time and ping frequency to the Print Resource Server. The **Max Wait Time for Assembly Job (in min)** field specifies the maximum period of time that the system will attempt to connect to the Print Resource Server. The system will fail the job if the system cannot connect to the Print Resource Server within the indicated time. The **Ping Frequency to Print Resource (in sec)** field indicates the frequency at which the system will attempt to connect to the Print Resource Server. For example, if the **Max Wait Time for Assembly Job** field is 1 minute and the **Ping Frequency to Print Resource** is 30 seconds, then the system will attempt to connect every 30 seconds until the minute is up.

## Batch Interface Event

The Batch Interface Event (BIEVNT) page is used to specify information required to run the Batch Interface Pre-Processor job. The BIEVNT table stores information used to match against the input files to the job. The table can also be used to track how many table records or transactions are imported or submitted using the System Maintenance Utility (SysManUtil) or Multi Process Submit job against a specific interface run.

Before running the Batch Interface Pre-Processor job, you need to create the corresponding BIEVNT record to indicate the expected count and/or total that will be processed for the Batch ID, Department, Unit, Import Date, and Transaction or Table combination. If pre-approval is required, you need to approve the BIEVNT record before running the job. You cannot edit the Expected Received Date, Expected Transactions, or Expected Total after approving the record.

When the Batch Interface Pre-Processor job is run, the BIEVNT record is updated to reflect the results of the matching. You can run the Interface Batch Demand Report process to see the processed BIEVNT records and their corresponding processed interfaced transaction counts on a given day. Refer to the Batch Interface Pre-Processor and Interface Batch Demand Report run sheets in the *CGI Advantage - Financial Utilities Run Sheets* guide for more information.

When a specific interface run involves importing table records, overlaying table records, importing transactions, or submitting transactions, and an action is performed using the System Maintenance Utility (SysManUtil) or Multi Process Submit job with the BiEvtStatsListener and relevant parameter information, the BIEVNT record is updated to reflect the import or submit counts. Refer to the “[System Maintenance Utility Listener](#)” topic and the “Multi Process Submit” run sheet in the *CGI Advantage - Financial Utilities Run Sheets* guide for more information.

## Encrypted Batch Parameters

The Encrypted Batch Parameters page provides a central location for the system to store secure batch parameter information. Access to this table is configurable for only permitted users. Encrypted information is displayed in plain text on the table. Each FTP Password value in the Batch Catalog must be a Parameter ID from the Encrypted Batch Parameters page that is associated with the FTP password. The following fields are required on this table: Parameter ID and Parameter Value.

- › Field Information

Field Name	Description
Parameter ID	The name to associate with the encrypted parameter.
Parameter Value	Indicates the value of the parameter to encrypt.
Department	Indicates the Department to associate with the encrypted parameter.
Unit	Indicates the Unit to associate with the encrypted parameter.
Description	Provides a useful description of the encrypted parameter.

## File Transfer

The File Transfer functionality allows authorized users to view and access predefined file directories on the application servers for the purposes of downloading and uploading of files from/to file locations accessible from a user's workstation. Multiple pages allow for administrators to:

- Define file directories that are accessible for file transfer
- Define roles that can access the directories
- Define allowable file actions (upload and download) for each role

Access to most of the File Transfer pages is from the Advantage Administration application. The File Transfer Maintenance (FUD) page can be accessed from the Advantage Administration application as well as the related unified applications.

## File Transfer Application

The File Transfer Application (FTAPPL) page allows you to define the application name and the associated application codes that are used in file transfers.

> Field Information

Field Name	Description
Application	Specifies the application identifier.
Name	Specifies the name of the application associated with the Application code.

## Tenant

(For CGI internal use only) A Tenant (TENANT) page allows you to define different Advantage tenants in a multiple Advantage instance site. The concept of a “tenant” is used by CGI when hosting sites to identify the individual sites. A default “Advantage” tenant entry is provided for used by non-hosted sites. At least one entry is required on the TENANT page.

> Field Information

Field Name	Description
Tenant	Specifies the identifier tenant.
Name	Specifies the name associated with the tenant.

## Application URL Mapping

The Application URL Mapping (APPURL) page allows you to define the URL for the application so that REST Web Services can communicate to the application and request files/folders from target application file system.

> Field Information

Field Name	Description
Application	Specifies the name of the application associated with the URL defined on the record.
Tenant	Specifies the tenant associated with the application.
URL	Specifies the URL of the application to derive the files and folders on the File Transfer Maintenance page.

## File Transfer Folder

The File Transfer Folder (FOLDTL) page allows you to define the file directories that are accessible from the file transfer page.

> Field Information

Field Name	Description
------------	-------------

Folder	Identifier to be associated with the file folder entry.
Tenant	Specifies the Tenant associated with the File Transfer Folder entry.
Folder Path	Full directory path, including server name, of the directory to be defined.
Folder Display Name	Alias name associated with the folder. This will be the name of the folder displayed on the File Transfer page instead of the actual folder name.

## File Transfer Role

The File Transfer Role (FTROLE) page allow you to define security role access to the specified file directory. On the File Transfer Role page, you will associate the security role with the file directory and the authorized actions that users assigned to that security role will be allowed to perform on the file directory.

› Field Information

Field Name	Description
Security Role ID	Security Role Identifier that will be authorized to access to the file directory.
Application	Specifies the application that is associated with the file directory.
Tenant	Specifies the Tenant associated with the File Transfer Role entry.
Folder	The name associated with the Folder Transfer folder entry that specifies the file directory that will be accessible for the security role.
Filter By Name	<p>Indicates the value to be used as a filter for the File Transfer Maintenance (FUD) page. Based on the selected security role, users can only view/update files/folders on the FUD page that include the value in the File/Folder name field. If no value is provided in this field, then a filter is not applied based on the file or folder name. The value entered in the field is case sensitive and a case sensitive search will be performed.</p> <p>More than one value can be entered by separating the values with " ". As an example, Filter By Name value of "adv auto"</p>



	would only return files and folders that contain “adv” or “auto” in the name.
Filter By Extension	<p>Indicates the file extension to be used as a filter for the File Transfer Maintenance (FUD) page. Based on the selected security role, users can only view/update files on the FUD page that have the indicated extension. If no value is provided in this field, then a filter is not applied based on the file extension. The value entered in the field is case sensitive and a case sensitive search will be performed.</p> <p>More than one value can be entered by separating the values with “ ”. As an example, Filter By Extension value of “pdf txt” would only return files with a file extension of “pdf” or “txt”.</p>
Allow Download	Specifies whether the security role is authorized to perform the Download action to download files from the specified folder.
Allow Upload	Specifies whether the security role is authorized to perform an Upload action to upload files from the specified folder.
Allow Create Folder	Specifies whether the security role is authorized to perform the Create Folder action to create a new sub-folder within the selected file directory on the tree component.
Allow Rename	Specifies whether the security role is authorized to perform the Rename action to rename one or more selected files on the grid component.
Allow Delete	Specifies whether the security role is authorized to perform the Delete action to delete one or more selected files on the grid component.

## File Transfer Maintenance

The File Transfer Maintenance (FUD) page displays file directories in a tree/grid format that users are allowed to access. On the FUD page, the user is able to perform authorized actions on the list of folders or files. Note: Folders cannot be selected for the Download action. Setup on the File Transfer Role (FTROLE) page determines what users can view/update/download. Refer to the “[File Transfer](#)” and “[File Transfer Role](#)” topics for more information.

The File Transfer Maintenance (FUD) page displays the following CGI disclaimer, “*Please note that using this application may require you to upload sensitive personal information. It is recommended that you upload such information in accordance with your organization’s privacy policy and use fields as designed to prevent unauthorized access to sensitive information.*” This text cannot be altered or hidden through extensibility.

Sites can append additional information to the CGI disclaimer on the Home Page and File Transfer Maintenance (FUD) page by specifying the text to be appended using the Configurable Text (CTEXT) record, where the CTEXT code equals FUDDCLM. The appended text will appear as a separate paragraph following the CGI disclaimer.

If the ENABLE\_FT\_AUDIT\_LOG parameter is set to *true* on ERP Application Parameters (ERPCTRL), file transfer requests through the FUD page are logged to File Transfer Audit Log (FTAUDLOG).

This page also supports Sorting and Pagination functions for grid records. Sorting allows users to view Folders/Files in the order they want and it is available on all the grid (Folders/Files, Date, and Size) fields. The pagination options help view a set of (20, 50, 100, 500) records per page based on the option selected.

> Tree Component

- Displays authorized file directories and their sub-directories in a tree format. The user can expand individual file directories to view the associated sub-directories.

> Grid Component

- Displays files and sub-directories associated with the selected file directory on the tree component of the page.

> Grid Actions

- Displays allowable grid level actions. Individual actions will be visible if the action is allowed on the Security Role assigned to the user.
  - Create Folder – Action creates a sub-folder in the selected directory on the Tree Component.
  - Upload – Action allows user to upload one or more files to the selected directory on the Tree Component. The number of files that can be uploaded at a time and the maximum file size permitted for each file are limited by the MaxUploadFiles and MaxAttachmentSize configuration parameters defined in ADV30Params.ini.
  - Download – Action downloads selected files on the grid component to the file location specified by the user.
  - Delete – Action deletes selected files on the grid component.

> Grid Field Information

Field Name	Description
File/Folder	Displays a file name or folder name associated with sub-directories of the selected folder in the tree component.
Date	Last modified date of file or folder.
Size	Indicates the size of the file or folder.

## File Transfer Audit Log

The File Transfer Audit Log (FTAUDLOG) page allows you to view logged File Transfer API actions, which can come from various sources such as the File Transfer Maintenance (FUD) page, Advantage Connect, or PowerShell scripts. When the ENABLE\_FT\_AUDIT\_LOG parameter is set to *true* on ERP Application Parameters (ERPCTRL), file transfer requests are logged to FTAUDLOG.

> Field Information

Field Name	Description
User ID	User ID used to perform the action.
Application	Application against which the action was performed.  Determined from the Folder Path and corresponds to an Application set up on the File Transfer Application (FTAPPL) page.
Action	File transfer action performed.
Folder Path	The path to the folder where the action was performed. The folder path is relative to the application and does not contain absolute path references.
File / Folder	For the Create action, identifies the folder created.  For the Rename action, identifies the new name for the file.  For all other actions (Delete, Upload, Download), identifies the file involved in the action.
Action Date	Time and date when the action was performed.  This is a required search field with support for conditional operators.
Request ID	Autogenerated ID for the request.  When you delete or download multiple files, one log record is created for each file but they are assigned to the same Request ID.
Original File / Folder	For the Rename Action, identifies the original name for the file.  Not used for other actions.

Source of Request	<p>Identifies where the File Transfer API request came from:</p> <ul style="list-style-type: none"> <li>• Online – File Transfer Maintenance (FUD)</li> <li>• Offline – request from outside the application (for example, Advantage Connect using the File Transfer API for uploading/downloading files)</li> <li>• Script – request from script with a header parameter identifying it as a Script (for example, PowerShell script)</li> </ul>
Tenant	<p>(For CGI internal use only)</p> <p>Tenant against which the action was performed. By default, this is not displayed on the page.</p>

## Form Definition

The Form Definition page specifies the physical characteristics of the database files on which the control and application tables are defined. This table also indicates how tables are stored on the database files and defines the structure of the application database.

## Form Transformation Definition

The Form Transformation Definition page displays the setup for printing forms in the system. A record is maintained for each form to be printed with the form name and a unique identifier. This page enables the user to transform a form before printing in the following ways:

- By specifying the sequence of fields, as they would appear in the form.
- By defining relationships for the form based on form structure.
- By reordering of data from the original XML of the transaction.
- › Row-level actions
  - **Fields** - This action transitions you to the [Print Form Transformation Fields](#) page.
  - **Group Orders** - This action transitions you to the [Form Transformation Group Order](#) page.
  - **Group Relationships** - This action transitions you to the [Form Transformation Group Relationships](#) page.

## Form Transformation Group Order

The Form Transformation Group Order page displays the setup for printing forms in the system. A record is maintained for each form to be printed with the form name and a unique identifier. This page enables the user to transform a form before printing by reordering data from the original XML of the document.

- › Page-level action

- **Form Transformation Definition** - This action transitions you to the [Form Transformation Definition](#) page.
- › Row-level action
  - **View** - This action allows you to view/edit all group order definitions that are set up for the selected form.

## Form Transformation Group Relationships

The Form Transformation Group Relationships page displays the setup for printing forms in the system. A record is maintained for each form to be printed with the form name and a unique identifier. This page enables the user to transform a form before printing by defining relationships for the form based on form structure.

- › Page-level action
  - **Form Transformation Definition** - This action transitions you to the [Form Transformation Definition](#) page.
- › Row-level action
  - **View** - This action allows you to view/edit all group relationship definitions that are set up for the selected form.

## Forms Access Control

The Forms Access Control (FAC) page helps to map a form definition file (.xdp) with a schema translation (.xsl) and an Advantage Application Resource. This page defines an Output Pro form to the Advantage System and it also includes the elements of the IN\_JF\_TRANS table, which maps an Advantage resource with a data translation schema.

## Forms Printing Access Control

The Forms Printing Access Control page allows a forms printing administrator to define and control the access to combinations of print jobs (forms) and printers for a user or group of users. This is done by setting up a combination of User ID, Department and/or Unit. The forms printing administrator could be a centralized administrator or a decentralized administrator. In the later case, row level security can be used to secure the access control records shown on the page to only those that belong to the decentralized administrator's assigned Departments and/or Units. A value of "\*ALL" can be specified for the user id to specify that the access record is defined for a group of users. All users, by default, will then have access to that access record. Each access record contains:

- User ID – User ID or group of users, which have access
- Application Resource – Resource ID of the Application
- Print Job – Code for the Print Job
- Print Job Name – Name of the Print Job
- Print Resource – ID for the Print Resource

- Print Resource Name – Name of the Print Resource
- Dept. – Department to which the user/group of users belong
- Unit – Unit to which the user/group of users belong
- Preferred – Flag to identify that this is the preferred combination of Print Job and Output Resource (printer)

## Forms Server Definition

The Forms Server Definition page defines the forms server used for dynamic form integration.

## Interface Request Configuration

The Interface Request Configuration (IRC) page is used to configure interface request for a specific Department, Unit, and Transaction Code combination. IRC is not used by any existing Advantage process, but can be used by custom process to create and submit jobs.

The following fields are required on this table: Department, Unit, Transaction Code, User ID, and Password. Note that the system does not validate the Department, Unit, or Transaction Code because the Department or Unit may not be set up within the Advantage application and the Transaction Code field can be used for transactions or tables.

> Field Information

Field Name	Description
Department	Indicates the Department associated with the interface request.
Unit	Indicates the Unit associated with the interface request.
Transaction Code	Indicates the Transaction Code associated with the interface request. Can also be used for tables.
User ID	Specifies the value to use as User ID.
Password	Specifies the value to use as Password.
Job Label	Specifies the value to use as Job Label.
Process Immediate	Selects to process the job immediately.
Transaction Status On Import	Specifies the Transaction Status when importing transactions:

	<ul style="list-style-type: none"> <li>• 2 – Ready</li> <li>• 1 – Held</li> </ul>
Process Load Balance	Select to use load balancing.
Job Manager	Specifies which job manager to use.
Pre-Processor	Specifies which job to use for pre-processing.
SysManUtil Import Job	Specifies which job to use for importing transactions.
SysManUtil Submit Job	Specifies which job to use for submitting transactions.

## Job Server Control

The Job Server Control page is used to setup and maintain the list of job servers in the system. Each job server is identified by the application server's Host Name, Application Server name and the Job Manager Name. This page allows the system administrators to start and stop registering jobs and also update each job manager instance settings such as Current Job Count, Maximum Jobs Allowed, Job Poll Rate, and so forth.

› **Job Server Parameters**

The **Job Server Parameters** row-level action transitions you to the Job Server Parameters page, which allows you to update instance settings for the select job manager. Select Update Instance Settings after you have made your updates or select Back to return to the Job Server Control page without making any changes.

Field Name	Description
Application	Read-only field that displays the name of the application.
Application Date Job Queue Active?	Jobs assigned to the application date job queue become due based on the application date and not the system date. Shows if these jobs are being polled.
Current Job Count	Read-only field that shows the number of jobs being currently run by the Job Manager.
Default Job Queue Active?	Indicates if the jobs belonging to the default job queue are being polled.

Job Poll Rate (seconds)	Shows the frequency with which the Job Manager accesses the job queue table to pick up jobs that become due.
Maximum Jobs Allowed	Shows the current value for the maximum number of jobs that can be run concurrently by the Job Manager. This value may be updated by the System Administrator after evaluating memory and processor resources available for the associated Job Manager VLSs.
Nightly Job Queue Active?	Shows if the jobs assigned to the nightly cycle job queue are being polled.
Polling?	Read-only field that shows if the Job Manager is currently polling for jobs.
Process Assigned Jobs Only	Read-only field that indicates whether the Job Manager only processes assigned jobs.

> Page-level Related Actions

- **Start Registering Jobs** - Select this action to start registering jobs for the selected application server.
- **Stop Registering Jobs** - Select this action to stop registering jobs for the selected application server.

Note: To start or to stop registering jobs, the user must be a member of the JOBCTRL security role or the ADMN security role.

> Page-level Related Pages

- **Edit AppServer List** - This action transitions you to the Application Servers page, which allows you to edit the current list of Application Servers.

## Job Updater

The Job Updater (JOBUPD) page can be used to set up data for the Job Parameter Updater process. The Job Parameters Updater process updates batch parameter values for jobs run as part of the nightly cycle. This helps in reducing the manual effort in setting up the jobs for execution. The data on this table is a set of table row columns to be updated for specific jobs, which run regularly on a schedule.

> Field Information

Field Name	Description
------------	-------------



Catalog Id	The Catalog ID of the batch job whose batch parameters need to be updated by the Job Parameter Updater process.
Job Name	The name of the batch job whose batch parameters need to be updated as part of the Job Parameter Updater run.
Table Name	The name of the table whose specific cell needs to be updated. Typically, this is BS_CATALOG_PARAM although it can even update any other reference table, if the where-clause for the row to be updated is defined.
Parameter Name	If the table name is BS_CATALOG_PARAM then Parameter Name is used as name of the batch catalog parameter to be updated. In case of any other table name, this is the name of the column within that table which needs to be updated.
Parameter Value	Parameter Value keeps the reference to the actual value that is defined in the Job Updater Parameters table. The value is resolved at job execution time based on the expression defined for that value.
Frequency	This drop-down field is currently used to perform sort order for the updates to the table rows. Valid values are Daily, Weekly, Monthly, and Specific.
Where Clause	<p>This field is used when table name is anything other than BS_CATALOG_PARAM. It is an expression that follows a specific syntax. This expression is resolved at runtime to form the where-clause while updating table values. If this field is left blank, then all rows from the mentioned table name will be updated with specified value.</p> <p>Example:</p> <pre>`USER_ID = '\${USER_ID}' AND EMP_CD = \${EMP_CD}`</pre> <p>The where-clause should always be enclosed in the ` (ASCII backquote) character. String type column values should always be enclosed in ' (ASCII single quote) characters.</p>
Active Flag	The Job Parameters Updater job will pick up only those records from this table that have Active Flag set to true or checked.

## Job Updater Parameter Values

The Job Updater Parameter Values (JOBUPDPV) page can be used to set up values for job parameters and other reference table column updates. These values are referenced on the Job Updater page, which are picked up by the Job Parameter Updater batch job. Values can either be a value literal (constant) or a function or expression of some other column value.

For example:

PREV\_MONTH\_FY is defined as  $\{ \{ \text{CURR\_APD} \} == 1 \} ? \{ \text{CURR\_FY} \} - 1 : \{ \text{CURR\_FY} \}$

Previous month fiscal year value will be evaluated at runtime based on the value of **CURR\_APD** (Current Accounting Period) and **CURR\_FY** (Current Fiscal Year). Assume that **CURR\_APD** is calculated as 1 (January) and **CURR\_FY** is calculated as 2023 then **PREV\_MONTH\_FY** will be calculated as **2022**.

The system by default calculates values for the following entries in the Job Parameter Values table.

- APPL\_SYS\_DT - Application System Date from the Application Parameter (APPCTRL) table.
- CURR\_FY - FY from the R\_CLDT table based on the Application System Date from APPCRL.
- CURR\_APD - APD from R\_CLDT table based on the Application System Date from APPCTRL.
- AD\_CHAIN\_SPEC\_RANGE\_RUN\_ID - Minimum agent id/batch run id of the previous month AD chain run from BS\_AGENT table.
- AD\_CHAIN\_MAX\_RUN\_ID - Maximum Run Id from the AD\_DOC\_HDR table.

> Field Information

Field Name	Description
Parameter Value	Name of the parameter value that can be referenced in the Job Updater's Parameter Value column or can be even referenced in the expressions for other parameter values.
Expression	Expression field follows a specific syntax. It can directly take in literal values, which gets set as values on the specified Job Updater table record. It can also take in specific syntax expression that gets evaluated at runtime based on other parameter values.  Refer to the example given above.
Parameter Description	Description of the Parameter Value.

## Manage Reports and Forms Roles

The Manage Reports and Forms Role page allows you to indicate which reports and forms a Security Role ID can view on the [Reports and Forms](#) page.

› Field Information

Field Name	Description
Security Role ID	Security Role ID that will be authorized to access the indicated report or form.
Application	The application where the report or form is located.
Functional Module	The functional module associated with the report or form that the Security Role can view.
Catalog Name	Allows you to select a valid catalog entry for a report or form that the security role can view.

## Messages

This page allows you to set up and maintain messages in the application. Each message can be identified with a unique Code along with a short description of the error in the Text field, and a Severity. An extended description of the error message can be provided in the Explanation field and the override level can be specified as well. Messages can be error messages, warnings, not displayed or just informational messages. When the message Severity is *Not Displayed*, the system does not display the message to users and the message does not impact the normal processing of the application. It is recommended to set this option with caution especially when the original error Severity is *Error* or *Severe*.

Refer to the "Transaction Feedback" topic in the *Transactions User Guide* for information about messages displayed on transactions after selecting Save, Validate, or Submit actions. For information about messages displayed on inquiry pages or reference tables, refer to the "System Feedback" topic in the *Page/Table User Guide*.

## Print Form Transformation Fields

The Print Form Transformation Fields page displays the setup for printing forms in the system. A record is maintained for each form to be printed with the form name and a unique identifier. This page enables the user to transform a form before printing by specifying the sequence of fields, as they would appear on the form.

- › Page-level action
  - **Form Transformation Definition** - This action transitions you to the [Form Transformation Definition](#) page.
- › Row-level action
  - **View** - This action allows you to view/edit all field definitions that are set up for the selected form.

- **Form Transformation Definition** - This action transitions you to the [Form Transformation Definition](#) page and selects the form that was selected on this page.

## Print Job Setup

The Print Job Setup page is used to configure the print jobs for a particular print server. A print job tells the underlying forms printing software being used what to do when handling the forms printing request. This table is keyed by an application resource ID and a print job code.

For print jobs used to print documents, only the print jobs permitted to print for a given document based on the document phase are displayed in the Print Job drop-down selection.

## Print Resource Setup

The Print Resource Setup page is used to configure print resources that are valid targets in the application for print jobs. A print resource is the destination of a print job. It can be a printer, PDF file, Fax, or any other output device allowed by the underlying forms printing software. The IPRS table allows for multiple entries per Type. However, you can only specify one default record per Type. For example, you can have multiple resource entries where the Type is PDF; however, only one of the entries can have the Default flag selected.

Please refer to the "Print Resource Setup" topic in *CGI\_Advantage\_4\_BIRT\_Migration\_Guide.pdf* for information about day zero print resources and print resource parameters.

Please refer to the "Using Attachments" topic in *the CGI Advantage Transactions User Guide* for information about sending a separate email containing the attachments associated to the version of the transaction being printed to Email output.

Please refer to the "Print Resource Setup" topic in *the CGI Advantage Procurement User Guide* for information about configuring an Email print resource for certain procurement transactions to default the recipient, subject, message text, and sender information.

## Print Server Setup

The Print Server Setup page is used to configure the forms printing server that is being used in the current Advantage application. Each print server record represents a physical forms printing server. It is keyed by the print server ID.

## Setup Audit Log and External Notification

Audit logging for a particular table can be set up using the Setup Audit Log and External Notification page (AUDCTRL).

To enable audit logging on a table, the following information can be set up on this page:

- **Resource Name** – Enter the table name for which activity is being monitored. Note: The system will infer the Process Type as *Tables*.
- **Audit Enabled** – Select the check box if audit logging needs to be turned on for the particular table; otherwise, leave the check box unselected.

The rest of the fields are only relevant to external notification setup.

To enable external notification on a table or on a transaction, the following information can be set up on this page:

- **Resource Name** – Enter the table name for which events will be logged for external notification. If external notification is desired on a transaction then enter the Transaction Code in the Resource Name field. Depending upon the Resource entered, the system will infer the Process Type as *Tables* or *Transactions*. Note: Resource Name is case-sensitive.
- **External Notification** – Select the check box if external notification needs to be turned on for the particular resource (table or transaction); otherwise, leave the check box unselected. Refer to the “Enable Audit Logging of IN\_OBJ\_ATT\_STOR table” topic in the *System Administration Guide* for more information about this check box.

The following setup is relevant for setting up external notification for transactions only and does not apply to tables:

- If transactions belonging to a particular Department are to be selected for external notifications then enter that Department in the Transaction Department field; otherwise, leave Transaction Department blank to indicate transactions of any Department. The System will infer Transaction Action Code as *Submit* since logging of external notification is only done for the transaction action of Submit.
- Select the target Transaction phases (*Draft*, *Pending* and/or *Final*) for which external notifications needs to be enabled.

Note: It is possible to turn both Audit logging and External Notification on for a table.

For any changes on this page to take effect, the application server should be shut down and then restarted.

## Setup Batch Job

Using the Batch Setup pages, you can add/remove/modify the following items on the Batch Catalog:

- **Folder** – A logical grouping of jobs.
- **System Batch** – Individual jobs that are identified and scheduled.
- **Report** – Any job where the main purpose of the job is to generate a report.
- **Chain Job** – A sequence of jobs that is intended to run as a group, in a particular order. You can also use logic in the chain to indicate how to proceed if jobs are successful or not (based on the job return code). One or more jobs defined in the chain can also be disabled for a run.
- › **Row-level actions**
  - **Edit** - This action allows you to edit the selected item.
  - This action transitions you to the [System Level Process Parameters](#) page, if the Catalog Label is *Batch Catalog* and the Item Type is *Folder*.
  - This action transitions you to the [Catalog Folder Details](#) page if the Item Type is *Folder* and the Catalog Label is any value other than *Batch Catalog*.
  - This action transitions you to the [Item Details](#) page, if the Item Type is *System Batch* or *Report*.

- This action transitions you to the [Chain Job](#) page, if the Item Type is *Chain Job*.
- **View Items** - This action is available if the Item Type is *Folder*. Selecting this action displays any folders, batch jobs, chain jobs, or report jobs that exist directly beneath the folder.
- **Delete** - This action allows you to delete the select folder, batch job, chain job or report job. Deleting a folder results in the deletion of all items contained within that folder.

## System Level Process Parameters

System wide batch parameters are defined at the System Level on the System Level Process Parameters page. This page is accessed by selecting **Edit** from the row-level menu for the Batch Catalog record on the Setup Batch Job (that is, Item Type is *Folder* and Catalog Label is *Batch Catalog*).

- **AMSROOT** - Root directory of the batch files (for example, C:\AMSADV30\RTFiles)
- **AMSEXPORT** - For files that are created by the program and need to remain after the job is completed (that is, cannot be temporary files). This could include interface files that come from/go to third party sources (for example, \$AMSROOT\ExportImport).
- **AMSIMPORT** - For files that are used by the program and need to remain after the job is completed (that is, cannot be temporary files). This could include interface files that come from/go to third party sources (for example, \$AMSROOT\ExportImport).
- **AMSLOGS** - For batch framework log files. If the job requires its own log files, this is where it is put (for example, \$AMSROOT\Logs).
- **AMSPARM** - Batch job parameter files specific to a single job instance only (for example, \$AMSROOT\Parms).
- **AMSTEMP** - For temporary files, usually stamped with process ID (for example, C:\TEMP).
- **AMSSPOOL** - Batch job report files, statistic files, exception reports, and so forth. These files may be sent to an OS print queue. File name is usually date and time stamped (for example, \$AMSROOT\Spool).

Note: Assumptions while implementing system wide batch parameters: It is assumed that wherever in the Job processes system wide batch parameter variables (that is, AMSEXPORT, AMSIMPORT, AMSROOT, AMSLOGS, AMSPARM, AMSTEMP, AMSSPOOL) are declared as input parameters, care should be taken to set the overrideable flag for that variable to *true*, otherwise the process may fail.

## Catalog Folder Details

The Catalog Folder Details page allows you to create/modify a folder that appears within the Batch Catalog. Each item must have a Catalog Label, which is then displayed on the Batch Catalog. A Resource ID can also be assigned to the selected folder. The parent folder's Resource ID is defaulted to this field; however, it can be changed to a different value or the field can contain a blank value. The Folder's Resource ID is carried down to all of its subsets which include folders, chains, individual jobs, and individual reports within the folder. Note: If entered, the Resource ID must be valid on the Application Resources table with a Resource Type of *Job*.

The Item Parameters section allows you to add parameters with a default value that can then be inferred to child jobs within the folder that use the same parameter. When scheduling a job that exists within the selected folder, if a valid value does not exist for the parameter at the job level, then Advantage will infer the value of the parameter from the "closest" parent folder that includes the same parameter. The parameter value can be changed when scheduling a job, if the **Override** check box is selected for the parameter on this page.

Note: This page is accessed by selecting Edit for an entry on the [Setup Batch Job](#) page, where the Item Type is *Folder* and the Catalog Label is not *Batch Catalog*.

- > Page-level actions
  - **Add Item** - This action allows you to add a report job, batch job, chain job, or folder within the currently selected folder.

## Item Details

The Item Details page allows you to add/edit information about a batch or report job.

- > Field Information

Field Name	Description
Application	Choose the application for which this job is designated. Users with access to this application can schedule and run this job (provided they have access to the parent folder).
Job Queue Id	All jobs are assigned a job queue Id. Choose one from the three options.
Catalog Label	Specifies a name for this job that is used for display purposes in the catalog.
Class Name	Indicates the Java class for this job. This does not include the package name or the file extension ".class", for example "AMSBatchCleanUp".
Package Name	Indicates the Java package name for this job, for example "com.amsinc.common.batch".
Resource ID	This field assigns a Resource ID to the selected job. The parent folder's Resource ID is defaulted to this field; however, it can be changed to a different value or the field can contain a blank value. Note: If the defaulted value is not changed and the parent folder's Resource ID changes or is deleted, then the value in this field will also be changed or deleted.

	Note: The Resource ID entered must be valid on the Application Resources (SCRSRC) table with a Resource Type of <i>Job</i> .
Requires Approval	If checked, then a user with a "Batch Administrator" role must manually approve the job before it can be run.
Single Instance	If checked, indicates that only a single instance of the job can be running at any given time.
Cust. Param. UI Package	Specifies the name of the user interface Java package where custom pages belong, for example "Reports_Sys_Admin_App".
Cust. Param. Page	Specifies the name of a custom page Java class (without the .class extension) where specific parameters can be setup for this particular job, for example "PIN_SYS_MAN_PARM".

The Item Parameters section allows you to add parameters with a default value that can then be inferred as a default value for the job. When scheduling the job, the value of the parameter is inferred from the job level, if defined at the job level. If a valid value does not exist for the parameter at the job level, then Advantage will infer the value of the parameter from the "closest" parent folder that includes the same parameter. The parameter value can be changed when scheduling a job, if the **Override** check box is selected for the parameter.

## Chain Job

The Chain Job page allows you to modify a chain job's Catalog Label that appears within the Batch Catalog. A Resource ID can also be assigned to the selected chain job. The parent folder's Resource ID is defaulted to this field; however, it can be changed to a different value or the field can contain a blank value. If the defaulted value is not changed and the parent folder's Resource ID changes or is deleted, then the value in this field will also be changed or deleted. Note: If entered, the Resource ID must be valid on the Application Resources table with a Resource Type of *Job*. All jobs associated with this Chain Job must contain the same Resource ID entered here.

The Item Parameters section allows you to add parameters with a default value that can then be inferred to child jobs within the chain that use the same parameter. If a valid value does not exist for a parameter at the chain/job level, then Advantage will infer the value of the parameter from the "closest" parent folder/parent chain record that includes the same parameter. The parameter value can be changed when scheduling a job, if the **Override** check box is selected for the parameter on this page.

Note: This page is accessed by selecting Edit for an entry on the [Setup Batch Job](#) page, where the Item Type is *Chain Job*.

> Page-level actions

- **Configure Chain Job** - This action transitions you to the [Setup for Chain Job Steps](#) page, which lists all jobs that are part of the selected chain job.



## Setup for Chain Job Steps

The Setup for Chain Job Steps page allows you to view a listing of all jobs that are associated with the chain job that was selected on the Chain Job page. This page can only be accessed by selecting **Configure Chain Jobs** from the page-level menu on the [Chain Job](#) page.

You can specify the Pre Condition Return Code for each job in the chain. The next job in the chain will not be executed until the Return Code for the previous job matches the value in the Pre-Condition Return Code field of the previous job in the chain.

- > Page-level actions
  - **Add New Job** - This action transitions you to the [Item Details](#) page, which allows you to add a new job for the selected chain job. The job will be added to the end of the chain, but can be moved to a different position.
- > Row-level actions
  - **Edit Job** - This action transitions you to the [Edit Catalog Entry](#) page, which allows you to modify information about the individual job.
  - **Top** - Moves the selected batch job to the top of the selected chain job, so that it is executed first.
  - **Up** - Moves the selected batch job up one position for the selected chain job.
  - **Down** - Moves the selected batch job down one position for the selected chain job.
  - **Bottom** - Moves the selected batch job to the end of the list of jobs for the selected chain job, so that it is executed last.

## Edit Catalog Entry

The Edit Catalog Entry page, allows you to modify information about an individual job that belongs to a chain job. This page is accessed by selecting Edit Job from the row-level menu on the [Setup for Chain Job Steps](#) page.

- > Field Information

Field Name	Description
Application	Choose the application for which this job is designated. Users with access to this application can schedule and run this job (provided they have access to the parent folder).
Item Type	Specifies the type of the catalog item.
Catalog Label	Specifies a name for this job that is used for display purposes in the catalog.

Class Name	Indicates the Java class for this job. This does not include the package name or the file extension “.class”, for example “AMSBatchCleanUp”.
Package Name	Indicates the Java package name for this job, for example “com.amsinc.common.batch”.
Resource ID	This field assigns a Resource ID to the selected job. All jobs associated with a Chain Job must contain the same Resource ID.
Requires Approval	If checked, then a user with a "Batch Administrator role must manually approve the job before it can be run.
Cust. Param. UI Package	Specifies the name of the user interface Java package where custom pages belong, for example “Reports_Sys_Admin_App”.
Cust. Param. Page	Specifies the name of a custom page Java class (without the .class extension) where specific parameters can be setup for this particular job, for example “PIN_SYS_MAN_PARM”.

## Site Specific Parameters

The Site Specific Parameters (SPAR) page affects the processing of various functional areas and supports individual site needs in the Advantage HRM application.

Refer to the CGI Advantage HRM Site Specific Parameters PDF file for additional information regarding the setup of the Site Specific Parameters.

## System Processes

Certain CGI Advantage processes require exclusive access to identified application database tables while they are running. One or more of the table update actions (Insert, Update, Delete) may be protected for the identified table(s). At the same time one or more of the transaction actions (Create, Validate, Submit, Approve/Bypass Approval, Reject, Reject All) may be protected for identified transactions. The System Processes table allows you to configure a process to have exclusive access to one or more tables and transactions.

The System Processes table has three tabs in Edit or Create Mode:

> System Processes

- **Process ID** – This field is a unique identification for each process to be configured.
- **Short Description** – This is a required field that captures a short description for the process.

- **Long Description** – This is an optional field to capture a long description for the process.
  - **In Progress** – This check box ‘locks’ the reference tables and the specified action. When done manually by the System Administrator, it prevents any updates made through any CGI Advantage application code. This mode is used to reserve the tables and transactions for external processes as the ‘locking’ does not involve database locks. This secures the tables for updates and/or transactions for specified actions by that job only.
  - **Lock Resources on Process Fail** – When a system process is associated with a batch process and if the batch process fails, this check box will prevent ‘releasing’ of the resources for online users for actions defined to be ‘locked’ in situations when the batch process fails. When this check box along with the Process Failed check box is checked the dependent resources will be ‘locked’ as defined by the In Progress check box.
  - **Process Failed** – This check box denotes that the job process failed. This along with the Lock Resources on Process Fail check box will lock the resources when the batch process fails. The system will set and reset this check box when the batch process fails processing or completes successfully.
- › Reference Table Resources
- This tab allows you to provide a list of tables for each process along with associated actions to be protected/allowed for the duration of the process run. All tables chosen here must already be registered as application resources of type *Reference Table*. For example, you may choose to allow updates on a specific table while the Payroll Process is running, but not allow deletions or inserts while the Payroll Process is running.
- › Transaction Resources
- This tab allows you to provide a list of transactions for each process along with associated actions to be protected/allowed for the duration of the process run. All transactions chosen here must already be registered as application resources of type *Transaction*. For example, you may choose to allow creation of Timesheet transactions while the Payroll Process is running, but not allow Submission while the Payroll Process is running.

## Associating Batch Process with System Process

The System Process can be associated with a batch process using batch parameters. While scheduling a batch job, the batch administrator can define a batch parameter, `SYSTEM_PROCESS_ID`, with the value as the System Process Id. If the System Process needs to be associated with an entire chain job then the parameter should be defined in the chain job parameter list. Since the batch parameters are hierarchical, if the parameter is provided at both the chain and job step level then the job step level parameter will be used (chain job level parameter will be overridden). Only one System Process Id can be associated with a batch process.

Associating System Processes to Batch Processes using batch parameters and adding Transaction Type resources to the System Process is a feature flag driven feature. The `onlineBatchProcesses` feature flag in the `feature.conf` file must be enabled to turn on this feature.

## Transaction Print Action

The Transaction Print Action page is used to specify specific special print actions that can be taken during transaction printing. The Transaction Print Action page setup is done in the target environment as opposed to the Administration environment used for other forms setup tables. Currently, only Financial supports the use of the Transaction Print Action page, and the only print action available is *Hide Inactive Procurement Lines*.

By creating an entry on the Transaction Print Action page, the user tells the system that for the specified Transaction Type and Transaction Code, the associated print action is allowed, and what its default value will be.

## Advanced - Batch Processing

Please refer to the following topic for a list of all system processing jobs.

- [Batch Jobs](#)

### Batch Jobs

The batch jobs for System Processing are listed alphabetically in the below table. For detailed information on the jobs (such as when to run, input, output, and process parameters) refer to the respective Run Sheets guide.

Job Name	Description
Batch Interface Pre-Processor	This job performs a three way match among the Batch Interface Event (BIEVNT) record, details provided in the INF file, and transaction/reference table records in the XML file.
Job Parameters Updater	This job updates batch parameter values for jobs run as part of the nightly cycle. Please refer to the <i>CGI Advantage - Administration Utilities Run Sheets</i> guide for more information on this job.
Multi Process Submit	This job selects draft transactions based on the selection criteria and spawns System Maintenance Utility (SMU) jobs to submit them.
Online Forms Cleanup	This job loops through online View Forms records selecting those that have been previously accessed, or those that have expired, and deletes those records along with the corresponding PDF files. Generated PDF files will be purged nightly. Please refer to the <i>CGI Advantage - Financial Utilities Run Sheets</i> guide for more information on this job.
Submit Ready Transactions	This job submits any transactions that are in <i>Ready</i> Status based on the Transaction Code and Create User ID values included as batch parameters. If any transactions are selected for the criteria, it schedules a System Maintenance Utility (SMU) batch job for each Transaction Code and Create User ID combination to break up the work load. Please refer to the <i>CGI Advantage - Financial Utilities Run Sheets</i> guide for more information on this job.
Sync Transaction ECM Reference	The job searches attachments in the Enterprise Content Management (ECM) system and adds an attachment reference on the Advantage transaction.
System Maintenance Utility	This job is the primary utility that can be used to maintain table data and transactions in CGI Advantage. When submitting an SMU instance, you must first choose an action, which will determine all additional options you may

	<p>need to complete the action. Refer to the "<a href="#">Working with System Maintenance and Utility</a>" topic in this user guide for more information. Also, you can refer to the <i>CGI Advantage - Administration Utilities Run Sheets</i> guide for more information on this job.</p>
Output File Maintenance	<p>This job moves and archives files from where they were originally written by system processing. Common use is to use it to move files to an external entity's location such as to a Treasury for check printing. It can be used as a standalone batch job or can be configured as a job step within a chain job. Please refer to the <i>CGI Advantage - Financial Utilities Run Sheets</i> guide for more information on this job.</p>

## Working with System Maintenance and Utility

System Maintenance Utility (SysManUtil or more commonly abbreviated as SMU) is the primary utility that can be used to maintain table data and transactions in CGI Advantage. When submitting an SMU instance, you must first choose an action (all of which are listed in the next sections), which will determine all additional options you may need to complete the action.

The following features are common to most SMU actions:

Feature	Description
Statistics	An option exists to generating statistics for the actions performed. For example, if transactions are being imported into the system from an XML file, the utility writes to the log, information about the total number of transactions that were imported, how many transactions were successfully imported into the system and how many failed.
Restart Capability	The capability exists to store restart information so that if a job fails for any reason the utility can pick up from where it left off and complete the rest of the job. For example, while importing an XML file that has 100 transactions the job fails after importing 45 transactions due to an error in the XML file. The job can be restarted after fixing the XML file and can continue to load the rest of the transactions into the system.
Chunking Database Commits	A provision exists to specify a commit block size for certain actions like importing transactions. This is very important for performance while processing a large job so that all the resources are not being used up. For example, while importing an XML file with 10,000 transactions the commit block size can be set to 100 so that the resources that have been used to import 100 transactions will be released once a commit is issued.
Maximum Errors Allowed	A provision exists to specify the maximum number of errors permitted while importing transactions or updating/inserting/deleting records from a table. If a value of 0 is specified then SysManUtil stops processing when the first error is encountered. If a value greater than 0 is specified then SysManUtil continues to process even though there are errors until the "maximum errors allowed" limit is reached. The data objects with errors are written to a separate error XML file. If a value of -1 is

specified then SysManUtil continues to process until it reaches the end of the input XML file irrespective of the number of errors encountered.

SysManUtil utility provides the following capabilities:

- [Maintenance of Tables](#)
- [Maintenance of Transactions](#)

## Maintenance of Tables

SysManUtil can be used to maintain table data by providing support for the following:

Action	Description
Delete	Existing records can be deleted based on data from an XML file. This data must contain the full primary key value of the record(s) to be deleted. If the record to be deleted does not exist, an error is issued. The user can optionally turn off the processing of business rules while deleting records.
Export	Data can be exported to an XML file. The user can optionally specify a range of records to be exported instead of all data. Records can be exported with or without object attachments, and the attachments may be exported to a zip file or to a directory.
Import	New records can be inserted from an XML file. If the record already exists in the table, the record will fail to be inserted. The user can optionally turn off the processing of business rules while inserting records into the table. If object attachments are present in the import XML file, then they are automatically imported
Overlay (from CSV)	Existing records can be updated and new ones inserted with data from a CSV file. The user can optionally turn off the processing of business rules while overlaying.
Overlay (from XML)	Existing records can be updated and new ones inserted with data from an XML file. The user can optionally turn off the processing of business rules while overlaying. For records that are updated, if new object attachments are present in the import XML file, then they are automatically imported, and if deleted object attachments are present, then they are deleted. For records that are inserted, if object attachments are present in the import XML file, then they are automatically imported.
Purge	All existing records can be purged without using an XML file. Please note that for this action SysManUtil will issue a direct SQL to the



	database hence no business rules will be fired to validate if the purge is allowed or not.
Update	Existing records can be updated with data from an XML file. If the record to be updated does not exist an error is issued. The user can optionally turn off the processing of business rules while updating records in a table. If new object attachments are present in the import XML file, then they are automatically imported, and if deleted object attachments are present, then they are deleted.

## Maintenance of Transactions

SysManUtil can be used to maintain transactions by providing support for the following:

Action	Description
Archive	<p>As a method of database maintenance, the archiving of transactions is a function that removes one or more transactions from the regular transaction catalog and moves it to the Archived Transaction Catalog. The data of the transaction is archived off either to the file system or to the database. When a transaction is archived, so are all of its associated attachments and transaction comments, regardless if the comment(s) has been suppressed.</p> <p>As there are system rules to prevent the archiving of certain types of transactions until they are 'complete', a separate Archive Historical feature exists to archive off transactions that have the Transaction Status of <i>Final Historical</i> where those edits for being complete do not apply.</p> <p>To facilitate more robust archiving, there is separate system processes for that action.</p>
Unarchive	<p>When one or more archived transactions need to be restored, this action returns the transaction to the regular transaction catalog and makes it available on all locations with transaction drill down capability.</p> <p>To facilitate more robust unarchiving, there is separate system processes for that action.</p>
Export	<p>One or more transactions can be exported to an XML file. The export can be done with or without object attachments, and the attachments may be exported to a zip file or to a directory. When a transaction is exported, so are all of its associated attachments and transaction comments. However if the comment(s) has been suppressed, then the comment(s) is not exported with the transaction.</p>

<p>Import</p>	<p>One or more transactions can be imported from an XML file with a specific Transaction Phase based on the import mode specified in the XML file. When a transaction is imported, so are all of its associated transaction comments provided the comments do not include transaction version numbers. All comments with version numbers are ignored (not imported), so as to prevent replication of comments.</p> <ul style="list-style-type: none"> <li>• Modification Drafts are created when the Mode is set to <i>MOD</i> (Modification). If new object attachments are present in the import XML file, then they are automatically imported, and if deleted object attachments are present, then they are deleted.</li> <li>• Final transactions are created when the Mode is set to <i>SE</i> (Snapshot Entry)</li> <li>• Template transactions are created when the Mode is set to <i>TE</i> (Template Entry)</li> </ul> <p>The option exists to applying overrides at import time so that when this transaction is submitted later, overrideable errors that are encountered will be converted to warnings and the transaction can be processed successfully.</p> <p>The option exists to bypassing approvals at import time so that when the transaction is submitted later, approvals are bypassed and the transaction is submitted directly to <i>Final</i>.</p> <p>The option exists to specify a Transaction Status of <i>Hold</i> or <i>Ready</i>. A status of <i>Hold</i> will cause any automatic transaction processing to skip the imported transaction until that status is changed to <i>Ready</i>.</p>
<p>Transaction Custom Actions</p>	<p>Custom actions can be performed on one or more transactions based on the selection criteria specified to select transactions. Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction.</p>
<p>Mark for Processing</p>	<p>One or more transactions can be marked for automatic system processing. This action changes the Transaction Status of the selected transactions to <i>Ready</i>. Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction.</p>
<p>Hold for Processing</p>	<p>One or more transactions can be held from automatic system processing. This action changes the Transaction Status of the selected transactions to <i>Held</i>. Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction.</p>
<p>Discard (Cancel)</p>	<p>One or more transactions can be discarded from the system if the Transaction Status is <i>Draft</i> or a Cancellation Draft created if the</p>

	<p>Transaction Status is <i>Final</i>. Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction.</p>
Edit	<p>When this action is performed on one or more transactions, a modification draft version of each transaction is created from the existing final version. Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction. Also, not all types of transactions allow a modification.</p>
Deactivate	<p>This action deactivates active transactions with a Transaction Phase of <i>Final</i> so that they cannot be cancelled, modified or referenced. Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction.</p>
Activate	<p>This action activates active transactions with a Transaction Phase of <i>Final</i> so that they can be cancelled, modified or referenced. Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction.</p>
Validate Submit	<p>These actions will validate or submit one or more transactions. If the Transaction Status is not specified, only <i>Ready</i> transactions are validated. If the Transaction Phase is not specified, only <i>Draft</i> transactions are validated.</p> <p>The option of triggering an exception report exists with the following formats for exception reports are supported.</p> <ul style="list-style-type: none"> <li>• DETAILED – lists detailed information of processing of transactions including error messages</li> <li>• FAILED_DOCS – list of transactions that failed processing</li> <li>• PROCESSED_DOCS – list of transactions that were successful</li> <li>• FAILED_LINES – transaction lines that failed processing</li> <li>• DOCUMENT_STATUS - list of transactions that were successful, and details about the transaction lines and the business errors reported</li> </ul> <p>Please note that due to the nature of this action a commit is issued after performing the action on each individual transaction.</p>
Reject Pending	<p>This action reject one or more pending transactions. If the Transaction Status of the transactions to be rejected is not specified, then by default, only <i>Submitted</i> transactions are selected for rejection. The Transaction Phase of transactions to be rejected can only be <i>Pending</i> by default. The user also has the option of specifying if an exception report must be generated while rejecting the transactions. This action</p>

	<p>would reject the transaction completely out of workflow regardless of approver, approval level or routing sequence level.</p> <p>The action works only for Internal Workflow.</p>
Approve	<p>This action approves one or more transactions. The major use of this action is to hold all transactions that meet one or more criteria in hold for a period of time while users manually approve or reject the transactions. Then at an appointed time, all transactions have a final approval placed on them by this batch job and presumably move to a Final state. Before that time, no transactions go past a Pending state.</p>

## Setting up Parameters

The System Maintenance Utility is driven by the parameters that are passed to it. There are three primary ways in which parameters can be supplied to this utility:

- Input parameter file
- Regular job parameters
- Custom parameter screen

Parameters are read using the following precedence:

- If an input parameter file has been specified, then the parameter values from this file are used
- If parameters have been specified using the custom screen, then these parameters are used
- If neither of the above has been specified, the regular job parameters are used

Please note that to use a parameter file with SysManUtil, the name of the parameter file has to be passed as a parameter to SysManUtil. This can be done either by using the custom parameter screen or by specifying regular job parameters. If the parameter file is being specified as a regular job parameter then the parameter name is *PARAM\_FILE*.

Each method of specifying parameters has its own advantages and no particular method is recommended over another. The administrator can decide his/her preferred approach based on the needs at hand. For example, for repetitive tasks that are performed often, using the parameter file is a good choice as the parameter file needs to be set up only once and all the tasks can be grouped together as one job. Using the regular job parameters is a good choice for SysManUtil to perform a specific step in a chain job, while the custom parameter screen is a good choice to perform ad hoc jobs.

## Input Parameter File

An input parameter file can be used to specify the parameters that are required to run a SysManUtil process. The use of the input parameter file has the following advantages:

- Greater flexibility in providing parameters
- Can perform more than one action in a single run

SysManUtil expects the parameter file to be present in the directory specified by the configuration parameter *XMLExportFileLocation* defined in "ADV30Params.ini". The basic format for a parameter file is as follows:

```
#A Comment
**ACTN_CD=<Action to be Performed>
PARAM_LINE_
<Parameter Name>=<Parameter Value>
<Parameter Name>=<Parameter Value>
...
PARAM_LINE_
<Parameter Name>=<Parameter Value>
<Parameter Name>=<Parameter Value>
...
**ACTN_CD=<Action to be Performed>
PARAM_LINE_
<Parameter Name>=<Parameter Value>
<Parameter Name>=<Parameter Value>
...
```

Every parameter file consists of one or more sections where each section is used to represent an action that can be performed. The start of a new section is indicated using "\*\*\*". Each section is made up of the following:

- Action or task to be performed at the beginning of the section using the format:  
**\*\*ACTN\_CD=<Action to be Performed>**
- Name-value pairs of the parameters required to perform the action in each new line, with the added flexibility of specifying multiple sets of parameters in different lines using the format:

```
PARAM_LINE_
<Parameter Name>=<Parameter Value>
<Parameter Name>=<Parameter Value>
...
```

- Comments – any line starting with a “#” can be used to indicate a comment. If the beginning of the section is commented then all the lines until the beginning of an uncommented section are ignored.

The example below gives a sample parameter file that is used to import transactions from multiple XML files. As indicated below, each XML file is specified in a separate parameter line.

```
#Import the transactions from these files
```

```
**ACTN_CD=DOCIMPORT
```

```
_PARAM_LINE_
```

```
FILE_NM=ImportFile1.xml
```

```
APPLY_OV=true
```

```
BYPS_APRV=true
```

```
DOC_STA=READY
```

```
COMMIT_BLOCK=25
```

```
MAX_ERRORS=10
```

```
ERROR_FILE_NM=ErrorFile1.xml
```

```
RESTART_FL=true
```

```
STATS=true
```

```
_PARAM_LINE_
```

```
FILE_NM=ImportFile2.xml
```

```
APPLY_OV=true
```

```
BYPS_APRV=true
```

```
DOC_STA=READY
```

```
COMMIT_BLOCK=25
```

```
MAX_ERRORS=10
```

```
MAX_ERRORS=10
```

```
ERROR_FILE_NM=ErrorFile2.xml
```

```
RESTART_FL=true
```

```
STATS=true
```

## Job Framework

SysManUtil can be set up to run as a standalone job in the job framework or it can be set up as one of the steps in a chain job. Chain job steps that use SysManUtil can be set up so that they are geared more towards performing a single task rather than trying to cater for all the actions that are supported by SysManUtil at once. The advantage of setting up jobs this way is that the number of parameters that must be set up for each step is limited. For example, a SysManUtil job can be setup that only submits AD transactions. This could be setup as part of the Automatic Disbursement Chain Job,

For more information about how to set up jobs please refer to the chapter on “Batch Processing: Using the Batch Administration Pages”. For a complete list of parameters that are required for each action that is supported, please refer to the later section on “System Maintenance Utility Parameters”.

## Custom Parameter Screen

SysManUtil is an all-purpose utility to maintain tables and transactions in the system. Due to the various actions that can be performed by SysManUtil, if a single batch setup is used to handle all actions, trying to manage the parameter setup for this job can be very cumbersome. For this reason, a custom parameter screen has been developed to aid in the setting up of parameters for SysManUtil. While setting up parameters for a job that is being scheduled, the custom parameter screen can be accessed by clicking the **Edit Parameters** button found at the top of the Edit Job page and then choosing the **Setup Custom Parameters** action from the page menu on the Job Parameters for System Maintenance Utility page.

The custom parameter screen is basically made up of a main section and the following subsections:

- **System Maintenance Utility Parameters** – A section that is always displayed no matter the action where either the Action Code or the Parameter File Name must be specified. The page will then show only the relevant sections based on the action selected. Even within a section, only parameters that are valid for the action are enabled and the rest are disabled.
- **Transaction** – section in which information pertaining to the selection criteria is entered.
- **Table** – section in which information pertaining to the selection criteria for tables is entered.
- **Import/Export/Archive** – section in which information pertaining to importing/exporting of transactions or tables and archiving/unarchiving of transactions is entered.
- **Exception** – section in which information pertaining to generating exception reports for transactions is entered.
- **Restart** – section in which information pertaining to storing restart information is entered.

## System Maintenance Utility Parameters

This section lists all of the actions that can be performed by SysManUtil and the parameters that can be set for each action.

- [Exporting Records from a Table](#)
- [Insert Records in a Table](#)
- [Update Records in a Table](#)

- [Overlay Records in a Table](#)
- [Overlay Records in a Table from CSV](#)
- [Delete Records from a Table](#)
- [Purge a Table](#)
- [Archiving Transactions](#)
- [Archiving Historical Transactions](#)
- [Unarchiving Transactions](#)
- [Exporting Transactions](#)
- [Transaction Import](#)
- [Performing Transaction Custom Actions](#)
- [Mark Transactions for Processing](#)
- [Hold Transactions from Processing](#)
- [Discarding Transactions](#)
- [Editing Transactions](#)
- [Activating Transactions](#)
- [Deactivating Transactions](#)
- [Submitting Transactions](#)
- [Validating Transactions](#)
- [Printing Transactions](#)
- [Rejecting All Pending Transactions](#)
- [Approve Transaction](#)

## Exporting Records from a Table

This action is used to export records from a table to an XML file, using the parameter range you specify. If the action code is to be specified using the parameter file, the value is *TBEXPORT*; otherwise, it is *202*.

Parameter Name Internal Parameter Name	Description
File Name FILE_NM	Name of the XML file to which table data is being exported from the system. SysManUtil writes the file to the directory specified by the configuration parameter <i>XMLExportFileLocation</i> defined in <i>ADV30Params.ini</i> . If attachments are exported, the attachments are either written to zip file or to a directory in the same location. The zip file or directory is named similarly to the XML file. The zip file has a “.zip” extension, and the directory has no extension.



<p>Table Name TBL_NM</p>	<p>Name of the table that is exported to the XML file.</p>
<p>From Key FM_KEY</p>	<p>This is an optional parameter that you can specify to restrict the data being exported to the file. When you specify a value for this parameter, data is exported beginning from this record. The format for the key is a WHERE clause that identifies a record using the primary key. For example, if table x has two primary keys A and B, the format for the key is:</p> <p>A=1;B=2.</p> <p>Note that character values need not be enclosed in quotes (for example, <i>VEND_CUST_CD=DELL</i>) and the following formats must be used to specify data values:</p> <ul style="list-style-type: none"> <li>• Just the date – <i>MM/DD/YYYY</i> (for example, <i>06/28/2001</i>)</li> <li>• Just the time – <i>HH24:MI:SS</i> (for example, <i>17:06:03</i>)</li> <li>• Date and time – <i>YYYY-MM-DD HH24:MI:SS</i> (for example <i>2002-06-18 14:08:55</i>)</li> </ul>
<p>To Key TO_KEY</p>	<p>This is an optional parameter that you can specify to restrict the data being exported to the file. When you specify a value for this parameter, data is exported up to this record. The format for the key is a WHERE clause that identifies a record using the primary key. For example, if table x has two primary keys A and B, the format for the key is:</p> <p>A=1;B=2</p> <p>Note that character values need not be enclosed in quotes (for example, <i>VEND_CUST_CD=DELL</i>) and the following formats must be used to specify data values:</p> <ul style="list-style-type: none"> <li>• Just the date – <i>MM/DD/YYYY</i> (for example, <i>06/28/2001</i>)</li> <li>• Just the time – <i>HH24:MI:SS</i> (for example, <i>17:06:03</i>)</li> <li>• Date and time – <i>YYYY-MM-DD HH24:MI:SS</i> (for example <i>2002-06-18 14:08:55</i>)</li> </ul>
<p>Commit Block Size COMMIT_BLOCK</p>	<p>Indicates the number of records that can be exported before the changes must be flushed to the file.</p> <p>If a value of &lt;=1 or no value is specified, then each exported record is flushed to the file one at a time.</p>
<p>Save Restart Info?</p>	<p>Stores information about the job so that it can be restarted if required. With the action "Table Export," when the job is restarted, only those</p>

<p>RESTART_FL</p>	<p>tables satisfying the search criteria greater than the stored checkpoint are chosen and exported. New records that may have entered the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job. This parameter is set to <i>true</i> by default.</p>
<p>Generate Statistics? GENERATE_STATS?</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Table Export," the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of records exported</li> <li>• Number of records that were successfully exported</li> <li>• Number of records that failed to be exported</li> </ul>
<p>Export With Attachments EXP_ATT_FL</p>	<p>When this parameter is set to <i>true</i>, SysManUtil will export the object attachments along with each data object exported. This parameter is set to <i>false</i> by default.</p>
<p>Attachments Export Type EXP_ATT_TYP</p>	<p>When exporting with attachments, this parameter indicates to SysManUtil how to export the attachments. To export the attachments to a .zip file, named the same as the export XML file, set this value to 1. To export the attachments to a directory, named the same as the export XML file, set this value to 2. To export the attachment in a binary encoded format, named the same as the XML Stream, set this value to 3. The attachment .zip file or directory will be located in the directory specified by the configuration parameter <i>XMLExportFileLocation</i> defined in ADV30Params.ini This parameter is set to 1, zip file, by default</p>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used or not to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional. When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears, determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p>

	<p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job. USER_ID cannot be located within a _PARAM_LINE_ block.</p>
<p>Excluded Attributes EXCL_ATT</p>	<p>This parameter restricts exporting for the specified attributes from the Data Object. When the value for this field is specified, SysManUtil excludes those specified attributes from the Export action. The value of this parameter needs to be given in a specific format. The attributes that need to be excluded from the export action should be given in a semi colon delimited format. For example to exclude attributes TBL_LAST_DT, AMS_ROW_VERS_NO the format should be given as:</p> <p>TBL_LAST_DT;AMS_ROW_VERS_NO</p> <p>This parameter has no default value.</p>
<p>Exclude Non-Persistent Attributes EXCL_NON_PERSIST_ATT</p>	<p>This parameter restricts exporting the non-persistent fields from the Data Object. When this parameter is set to true, SysManUtil will export the Data Object without non persistent fields and vice versa.</p> <p>This parameter is set to false by default.</p>
<p>Use Hierarchical XML USE_HIERCL_XML</p>	<p>Instructs the SMU Job to produce hierarchical XML. If this flag is true (checked), then the XML components will be arranged in a hierarchical order.</p> <p>This parameter is set to false (not checked) by default.</p>

### Insert Records in a Table

This action is used to insert new records into a table from an XML file. If the action code is to be specified using the parameter file, the value is *TBLIMPORT*; otherwise, it is *201*.

Parameter Name	Description
----------------	-------------

Internal Parameter Name	
<p>File Name FILE_NM</p>	<p>The name of the XML file that has records to be inserted into the system. SysManUtil expects the file to be located in the directory specified by the configuration parameter <i>XMLImportFileLocation</i> defined in ADV30Params.ini.</p>
<p>Commit Block Size COMMIT_BLOCK</p>	<p>Indicates the number of records that can be inserted before the changes must be committed to the database.</p> <p>If an error is encountered while importing one of the records in a given commit block, first all the records in that block are rolled back. Next, SysManUtil attempts to load and commit the records in the failed block one record at a time so that the records that can be inserted in that block will be inserted. It is essential to specify an optimal commit block size so that there is not a significant degradation in performance.</p> <p>If a value of <math>\leq 1</math> or no value is specified, then each imported record is committed to the database one at a time.</p>
<p>Enforce Edits? EDITS_FL</p>	<p>Indicates whether the records from the XML file are inserted with or without business rules applied. By default, this flag is set to <i>true</i> to indicate that records are inserted with business rules enabled.</p>
<p>Max. Errors Allowed MAX_ERRORS</p>	<p>Indicates the maximum number of logical errors that can occur while inserting records before the process is terminated.</p> <p>If a value of <math>-1</math> is specified, the processing does not stop irrespective of the number of errors encountered.</p>
<p>Detail Import Message Reporting DTL_IMP_MSG_FL</p>	<p>Indicates whether error message details for each failed to be processed record should be recorded in the error file.</p>
<p>Error File Name ERROR_FILE_NM</p>	<p>Specifies the name of the file to which the records that failed to be inserted are written. The file containing the errors is located in the directory specified by the configuration parameter <i>XMLImportErrorFileLocation</i> defined in ADV30Params.ini.</p>
<p>Save Restart Info? RESTART_FL</p>	<p>Stores information about the job so that it can be restarted if required. In the case of inserting table records, when the job is restarted, SysManUtil attempts to insert all records in the XML file after the last checkpoint. This parameter is set to <i>true</i> by default.</p>

<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. In the case of inserting table records, the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of records inserted</li> <li>• Number of records that were successfully inserted</li> <li>• Number of records that failed to be inserted</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>

<p>Listener Name LISTENER_NAME</p>	<p>When Generate Status is set to <i>true</i>, this parameter can be used to specify a listener class that implements the SysManUtilStatsListener interface to perform additional logic during table import.</p> <p>This is an optional, action-specific parameter. Only one listener class can be specified.</p> <p>If a parameter file is used, place the LISTENER_NAME parameter after the <b>**ACTN_CD</b> parameter and prior to the action code's first <b>_PARAM_LINE_</b> parameter.</p> <p>If a parameter file is not used, the LISTENER_NAME parameter can be set up as one of the job's input parameters. Note that the System Maintenance Utility Parameters page used for configuring SysManUtil job parameters online does not support this.</p>
--	--

### Update Records in a Table

This action is used to update existing records from a table based on the information provided in the XML file. If the action code is to be specified using the parameter file, the value is *TBLUPDATE*; otherwise, it is *203*.

<p>Parameter Name Internal Parameter Name</p>	<p>Description</p>
<p>File Name FILE_NM</p>	<p>Name of the XML file that has records to be updated in the system. The file is located in the directory specified by the configuration parameter <i>XMLImportFileLocation</i> defined in <i>ADV30Params.ini</i>.</p>
<p>Commit Block Size COMMIT_BLOCK</p>	<p>Indicates the number of records that can be updated before the changes must be committed to the database.</p> <p>If an error is encountered while importing one of the records in a given commit block, first all the records in that block are rolled back. Next, SysManUtil attempts to load and commit the records in the failed block one record at a time so that the records that can be updated in that block will be updated. It is essential to specify an optimal commit block size so that there is not a significant degradation in performance.</p> <p>If a value of <math>\leq 1</math> or no value is specified, then each updated record is committed to the database one at a time.</p>
<p>Enforce Edits? EDITS_FL</p>	<p>Indicates whether the records from the XML file are updated with or without business rules applied. By default, this flag is set to <i>true</i> to indicate that records are updated with business rules enabled.</p>

<p>Max. Errors Allowed MAX_ERRORS</p>	<p>Indicates the maximum number of logical errors that can occur while updating records before the update process is terminated.</p> <p>If a value of <math>-1</math> is specified, the processing does not stop irrespective of the number of errors encountered.</p>
<p>Detail Import Message Reporting DTL_IMP_MSG_FL</p>	<p>Indicates whether error message details should be recorded in the error file.</p>
<p>Error File Name ERROR_FILE_NM</p>	<p>Specifies the name of the file to which the records that failed to update are written. The file containing the errors is located in the directory specified by the configuration parameter <i>XMLImportErrorFileLocation</i> defined in ADV30Params.ini.</p>
<p>Save Restart Info? RESTART_FL</p>	<p>Stores information about the job so that it can be restarted if required. In the case of a table update, when the job is restarted, SysManUtil attempts to update all records in the XML file after the last checkpoint. This parameter is set to <i>true</i> by default.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. In the case of a table update, the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of records updated</li> <li>• Number of records that were successfully updated</li> <li>• Number of records that failed to be updated</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-</p>

	<p>specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
--	--

### Overlay Records in a Table

This action is used to overlay records in a table based on the information provided in the XML file. If the record exists, its information is updated. Otherwise, if the record does not already exist, a new record is added to the table. If the action code is to be specified using the parameter file, the value is *TBLOVERLAY*; otherwise, it is *204*.

Parameter Name Internal Parameter Name	Description
File Name FILE_NM	Name of the XML file that has records to be overlaid in the system. The file is located in the directory specified by the configuration parameter <i>XMLImportFileLocation</i> defined in <i>ADV30Params.ini</i> .
Commit Block Size COMMIT_BLOCK	<p>Indicates the number of records that can be overlaid before the changes must be committed to the database.</p> <p>If an error is encountered while importing one of the records in a given commit block, first all the records in that block are rolled back. Next, SysManUtil attempts to load and commit the records in the failed block one record at a time so that the records that can be overlaid in that block will be overlaid. It is essential to specify an optimal commit block size so that there is not a significant degradation in performance.</p>



	<p>If a value of <math>\leq 1</math> or no value is specified, then each overlaid record is committed to the database one at a time.</p>
<p>Enforce Edits? EDITS_FL</p>	<p>Indicates whether the records from the XML file are overlaid with or without business rules applied. By default, this flag is set to <i>true</i> to indicate that records are overlaid with business rules enabled.</p>
<p>Max. Errors Allowed MAX_ERRORS</p>	<p>Indicates the maximum number of logical errors that can occur while overlaying records before the overlay process is terminated.</p> <p>If a value of <math>-1</math> is specified, the processing does not stop irrespective of the number of errors encountered.</p>
<p>Detail Import Message Reporting DTL_IMP_MSG_FL</p>	<p>Indicates whether error message details for each failed to be processed record should be recorded in the error file.</p>
<p>Error File Name ERROR_FILE_NM</p>	<p>Specifies the name of the file to which the records that failed to be processed are written. The file containing the errors is located in the directory specified by the configuration parameter XMLImportFileLocation defined in ADV30Params.ini.</p>
<p>Save Restart Info? RESTART_FL</p>	<p>Stores information about the job so that it can be restarted if required. In the case of a table overlay, when the job is restarted, SysManUtil attempts to overlay all records in the XML file after the last checkpoint. This parameter is set to <i>true</i> by default.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. In the case of a table overlay, the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of records overlaid</li> <li>• Number of records that were successfully overlaid</li> <li>• Number of records that failed to be overlaid</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will</p>

	<p>be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE_ parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
<p>Listener Name LISTENER_NAME</p>	<p>When Generate Status is set to <i>true</i>, this parameter can be used to specify a listener class that implements the SysManUtilStatsListener interface to perform additional logic during table overlay.</p> <p>This is an optional, action-specific parameter. Only one listener class can be specified.</p> <p>If a parameter file is used, place the LISTENER_NAME parameter after the **ACTN_CD parameter and prior to the action code's first _PARAM_LINE_ parameter.</p> <p>If a parameter file is not used, the LISTENER_NAME parameter can be set up as one of the job's input parameters. Note that the System Maintenance Utility Parameters page used for configuring SysManUtil job parameters online does not support this.</p>

## Overlay Records in a Table from CSV

This action is used to overlay records in a table based on the information provided in the CSV file. If the record exists, its information is updated. Otherwise, if the record does not already exist, a new record is added to the table. If the action code should be specified using the parameter file, the value is *CSVOVERLAY*; otherwise, it is *208*.

Parameter Name Internal Parameter Name	Description
Table Name TBL_NM	Name of the table that the data from the CSV file should be written to.
File Name FILE_NM	Name of the CSV file that has records to be overlaid in the system. The file is located in the directory specified by the configuration parameter <i>XMLImportFileLocation</i> defined in <i>ADV30Params.ini</i> .
Commit Block Size COMMIT_BLOCK	<p>Indicates the number of records that can be overlaid before the changes have to be committed to the database.</p> <p>If an error is encountered while importing one of the records in a given commit block, all the records in that block are rolled back. Next, SysManUtil attempts to load and commit the records in the failed block one record at a time so that the records that can be overlaid in that block will be overlaid. It is essential to specify an optimal commit block size so that there is not a significant degradation in performance.</p> <p>If a value of <math>\leq 1</math> or no value is specified, then the value is read from parameter <i>CSV_DEFAULT_COMMIT_BLOCK_SIZE</i> in table <i>IN_APP_CRTL</i>. If no valid value can be retrieved from this table, the default value of 1000 is used.</p>
Enforce Edits? EDITS_FL	Indicates whether the records from the CSV file are overlaid with or without business rules applied. By default, this flag is set to <i>true</i> to indicate that records are overlaid with business rules enabled.
Max. Errors Allowed MAX_ERRORS	<p>Indicates the maximum number of logical errors that can occur while overlaying records before the overlay process is terminated.</p> <p>If a value of <math>-1</math> is specified, the processing does not stop regardless of the number of errors encountered.</p>
Error File Name ERROR_FILE_NM	Specifies the name of the CSV file to which the records that failed to be processed are written. The file containing the errors is located in the directory specified by the configuration parameter <i>XMLImportFileLocation</i> defined in <i>ADV30Params.ini</i> .
Save Restart Info? RESTART_FL	Stores information about the job so that it can be restarted if required. In the case of a table overlay, when the job is restarted, SysManUtil attempts to overlay all records in the CSV file after the last checkpoint. This parameter is set to <i>true</i> by default.

<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. In the case of a CSV overlay, the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of records overlaid.</li> <li>• Number of records that were successfully overlaid.</li> <li>• Number of records that failed to be overlaid.</li> </ul>
--	--

### Delete Records from a Table

This action is used to delete records in a table based on the information provided in the XML file. If the action code is to be specified using the parameter file, the value is *TBLDELETE*; otherwise, it is *205*.

<p>Parameter Name Internal Parameter Name</p>	<p>Description</p>
<p>File Name FILE_NM</p>	<p>Name of the XML file that has records to be deleted from the system. The file is located in the directory specified by the configuration parameter <i>XMLImportFileLocation</i> defined in <i>ADV30Params.ini</i>.</p>
<p>Commit Block Size COMMIT_BLOCK</p>	<p>Indicates the number of records that can be deleted before the changes must be committed to the database.</p> <p>If an error is encountered while deleting one of the records in a given commit block, first all the records in that block are rolled back. Next, SysManUtil attempts to delete and commit the records in the failed block one record at a time so that the records that can be deleted in that block will be deleted. It is essential to specify an optimal commit block size so that there is not a significant degradation in performance.</p> <p>If a value of <math>\leq 1</math> or no value is specified, then each deleted record is committed to the database one at a time.</p>
<p>Enforce Edits? EDITS_FL</p>	<p>Indicates whether the records from the XML file are deleted with or without business rules applied. By default, this flag is set to <i>true</i> to indicate that records are deleted with business rules enabled.</p>
<p>Max. Errors Allowed MAX_ERRORS</p>	<p>Indicates the maximum number of logical errors that can occur while deleting records before the delete process is terminated.</p> <p>If a value of <math>-1</math> is specified, the processing does not stop irrespective of the number of errors encountered.</p>

<p>Detail Import Message Reporting DTL_IMP_MSG_FL</p>	<p>Indicates whether error message details for each failed to be processed record should be recorded in the error file.</p>
<p>Output Error File ERROR_FILE_NM</p>	<p>Specifies the name of the XML file to which the records that failed to be deleted are written. The file containing the errors is located in the directory specified by the configuration parameter <i>XMLImportErrorFileLocation</i> defined in ADV30Params.ini.</p>
<p>Save Restart Info? RESTART_FL</p>	<p>Stores information about the job so that it can be restarted if required. In the case of deleting records, when the job is restarted, SysManUtil attempts to delete all records in the XML file after the last checkpoint. This parameter is set to <i>true</i> by default.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. In the case of deleting records, the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of records deleted</li> <li>• Number of records that were successfully deleted</li> <li>• Number of records that failed to be deleted</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first <b>**ACTN_CD</b> parameter within the file.</p>

On the other hand, specifying an action-specific User ID requires the USER\_ID parameter to be located after the applicable \*\*ACTN\_CD parameter and prior to the action code's first \_PARAM\_LINE parameter. That is, the USER\_ID must be physically located in between \*\*ACTN\_CD and the next \_PARAM\_LINE\_. The USER\_ID cannot be located within a \_PARAM\_LINE\_ block.

If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER\_ID is the User ID that will be used in place of the execution User ID.

If the USER\_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.

### Purge a Table

This action is used to purge records in a table. If the action code is to be specified using the parameter file, the value is *TBLPURGE*; otherwise, it is *206*.

Parameter Name Internal Parameter Name	Description
Table Name TBL_NM	Name of the table whose data has to be purged from the system.
User ID USER_ID	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the</p>

parameter file. That is, the USER\_ID must be physically located just prior to the first \*\*ACTN\_CD parameter within the file.

On the other hand, specifying an action-specific User ID requires the USER\_ID parameter to be located after the applicable \*\*ACTN\_CD parameter and prior to the action code's first \_PARAM\_LINE parameter. That is, the USER\_ID must be physically located in between \*\*ACTN\_CD and the next \_PARAM\_LINE\_. The USER\_ID cannot be located within a \_PARAM\_LINE\_ block.

If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER\_ID is the User ID that will be used in place of the execution User ID.

If the USER\_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.

## Archiving Transactions

This action is used to archive one or more transactions based on a list that is provided in a flat file. If the action code is to be specified using the parameter file, the value is *DOCARCHIVE*; otherwise, it is *111*. Note that the format for the flat file that lists the Transaction ID identifiers that must be archived is as follows:

<DOC\_CD>,<DOC\_DEPT\_CD>,<DOC\_ID>[,<DOC\_VERS\_NO>]

<DOC\_CD>,<DOC\_DEPT\_CD>,<DOC\_ID>[,<DOC\_VERS\_NO>]

<DOC\_CD>,<DOC\_DEPT\_CD>,<DOC\_ID>[,<DOC\_VERS\_NO>]

Either a transaction collection (Transaction Code, Transaction Department Code and Transaction ID) can be specified or a particular version of the transaction can be specified in the flat file on a separate line. If just the transaction collection is specified then a line in the flat file is made up of three tokens separated by commas; otherwise, it is made up of 4 tokens separated by commas.

The example below gives a sample flat file that is used to archive a combination of transaction collections and individual transactions.

```
PO,060, 04230200000000000004
PO,080, 042302000000000000024
PO,060, 042302000000000000124,1
PO,060, 042302000000000000124,2
```

Parameter Name	Description
----------------	-------------

Internal Parameter Name	
<p>File Name FILE_NM</p>	<p>The fully qualified location of the flat file that contains the list of transactions that must be archived.</p>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>

Note: Please see the "Transaction Archiving" section of the *CGI Advantage Financial Utilities Run Sheets guide* for details on Application Parameters involved.



## Archiving Historical Transactions

This action is used to archive the historical transactions for a transaction collection specified in a flat file. If the action code is to be specified using the parameter file, the value is *DOCARCHIVEHIST*; otherwise, it is *112*. The format for the flat file to specify the list of transaction collections is the same as described in the preceding section for “Archiving Transactions” except that only transaction collections and not specific transaction versions can be specified.

Parameter Name Internal Parameter Name	Description
File Name FILE_NM	The fully qualified location of the flat file that contains the list of transaction collections whose historical transactions must be archived.
User ID USER_ID	<p>This parameter allows the specified User ID and not the submitter’s User ID or the submitter’s effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job’s input parameters. The submitter’s User ID or the submitter’s effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job’s execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first <b>**ACTN_CD</b> parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable <b>**ACTN_CD</b> parameter and prior to the action code’s first <b>_PARAM_LINE</b> parameter. That is, the USER_ID must be physically located in between <b>**ACTN_CD</b> and the next <b>_PARAM_LINE_</b>. The USER_ID cannot be located within a <b>_PARAM_LINE_</b> block.</p> <p>If a parameter file is not used, then the job’s input parameters are specified either as a regular job or on the custom parameter screen, as</p>

	<p>mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
--	---

Note: Please see the “Transaction Archiving” section of the *CGI Advantage Financial Utilities Run Sheets* guide for details on Application Parameters involved.

## Unarchiving Transactions

This action is used to unarchive one or more transactions based on a list that is provided in a flat file. If the action code is to be specified using the parameter file, the value is *DOCUNARCHIVE*; otherwise, it is *113*. The format for the flat file to specify the list of transaction collections is the same as described in the preceding section for “Archiving Transactions”.

Parameter Name Internal Parameter Name	Description
File Name FILE_NM	The fully qualified location of the flat file that contains the list of transactions that must be unarchived.
User ID USER_ID	<p>This parameter allows the specified User ID and not the submitter’s User ID or the submitter’s effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job’s input parameters. The submitter’s User ID or the submitter’s effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job’s execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically</p>

	<p>located just prior to the first <b>**ACTN_CD</b> parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the <b>USER_ID</b> parameter to be located after the applicable <b>**ACTN_CD</b> parameter and prior to the action code's first <b>_PARAM_LINE</b> parameter. That is, the <b>USER_ID</b> must be physically located in between <b>**ACTN_CD</b> and the next <b>_PARAM_LINE_</b>. The <b>USER_ID</b> cannot be located within a <b>_PARAM_LINE_</b> block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified <b>USER_ID</b> is the User ID that will be used in place of the execution User ID.</p> <p>If the <b>USER_ID</b> parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
--	--

Note: Please see the "Transaction Archiving" section of the *CGI Advantage Financial Utilities Run Sheets* guide for details on Application Parameters involved.

## Exporting Transactions

This action is used to export transactions from the system into an XML file based on the search parameters specified for the job. If the action code is to be specified using the parameter file, the value is *DOCEXP*; otherwise, it is *172*.

Parameter Name Internal Parameter Name	Description
File Name FILE_NM	The name of the XML file to which transactions will be exported from the system. The file will be written to the directory specified by the configuration parameter <i>XMLExportFileLocation</i> defined in <i>ADV30Params.ini</i> . If attachments are exported, the attachments are either written to zip file or to a directory in the same location. The zip file or directory will be named similarly to the XML file. The zip file will have a ".zip" extension, and the directory will have no extension.
Transaction Type DOC_TYP	The Transaction Type for the transactions to be exported. The parameter is optional. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions to be exported. The parameter is optional. The "*" notation can be used to represent wild cards.

Transaction Department Code DOC_DEPT_CD	The Transaction Department for the transactions to be exported. The parameter is optional. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions to be exported. The parameter is optional. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions to be exported. This parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Phase Code DOC_PHASE_CD	The phase of the transactions to be exported. This is an optional parameter.
Transaction Unit Code DOC_UNIT_CD	The Unit of the transactions to be exported. This is an optional parameter. The "*" notation can be used to represent wild cards.
Create User ID DOC_CREA_USID	The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Status DOC_STA_CD	The status of the transactions to be exported. This is an optional parameter.  The following transaction status values can be specified: <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
Commit Block Size COMMIT_BLOCK	Indicates the number of transactions that can be exported before the changes must be flushed to the file.  If a value of <=1 or no value is specified, then each exported transaction is flushed to the file one at a time.
Save Restart Info? RESTART_FL	Stores information about the job so that it can be restarted if required. With the action "Transaction Export," when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and exported. New transactions that may have entered the system between the time the job failed and was restarted

	are also chosen if they meet the selection criteria specified as parameters to the job. This parameter is set to <i>true</i> by default.
<p>Generate Statistics? GENERATE_STATS?</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Transaction Export," the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions exported</li> <li>• Number of transactions that were successfully exported</li> <li>• Number of transactions that failed to be exported</li> </ul>
<p>Export With Attachments EXP_ATT_FL</p>	<p>When this parameter is set to <i>true</i>, SysManUtil will export the object attachments along with each data object exported.</p>
<p>Attachments Export Type EXP_ATT_TYP</p>	<p>When exporting with attachments, this parameter indicates to SysManUtil how to export the attachments. To export the attachments to a .zip file, named the same as the export XML file, set this value to 1. To export the attachments to a directory, named the same as the export XML file, set this value to 2. The attachment .zip file or directory will be located in the directory specified by the configuration parameter <i>XMLExportFileLocation</i> defined in the ADV30Params.ini file.</p>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first <b>**ACTN_CD</b> parameter within the file.</p>

	<p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
<p>Excluded Attributes EXCL_ATT</p>	<p>This parameter restricts exporting for the given attributes from the Transaction. When the value for this field is specified, SysManUtil excludes those specified attributes from the Export action. The value of this parameter needs to be given in a specific format. The attributes which need to be excluded from the export action should be given in a semi colon delimited format. For example to exclude attributes DOC_LAST_DT, DOC_LAST_USID the format should be given as:</p> <p>DOC_LAST_DT;DOC_LAST_USID</p>
<p>Exclude Non-Persistent Attributes EXCL_NON_PERSIST_ATT</p>	<p>This parameter restricts exporting the non-persistent fields from the Transaction. When this parameter is set to true, SysManUtil will export the Transaction without non persistent fields and vice versa.</p>

## Transaction Import

This action is used to import one or more transactions from an XML file into the system. Various options can be specified while importing the transactions. If the action code is to be specified using the parameter file, the value is *DOCIMPORT*; otherwise, it is *171*.

Parameter Name Internal Parameter Name	Description
File Name FILE_NM	The name of the XML file to be imported into the system. The file to be imported is located in the directory specified by the configuration parameter <i>XMLImportFileLocation</i> defined in <i>ADV30Params.ini</i> .
Apply Overrides? APPLY_OVERRIDES	Indicates that when loading the transaction, the override level in the transaction needs to be set to the override level of the user importing the transactions into the system (in this case, "sa" or "system administrator"). The override level of the user is set to facilitate the

	<p>transaction being submitted without someone having to open the transaction manually and apply overrides from the online system. However, if this transaction is subsequently edited from the online system, the override level is removed.</p>
<p>Bypass Approvals? BYPASS_APPROVAL</p>	<p>Indicates that when loading the transaction, the Bypass Approval Indicator in the transaction has to be set. The Bypass Approval Indicator allows the transaction to be submitted without going through approvals processing. If this transaction is subsequently edited from the online system, the Bypass Approval Indicator is removed.</p>
<p>Transaction Status DOC_STA_CD</p>	<p>Indicates the status of the transaction after it is imported into CGI Advantage. If this parameter is not specified, transactions are kept in the <i>Held</i> status by default.</p> <p>The following transaction status values can be specified</p> <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> </ul>
<p>Commit Block Size COMMIT_BLOCK</p>	<p>Indicates the number of records that can be inserted before the changes must be committed to the database.</p> <p>If an error is encountered while importing one of the records in a given commit block, first all the records in that block are rolled back. Next, SysManUtil attempts to load and commit the records in the failed block one record at a time so that the records that can be inserted in that block will be inserted. It is essential to specify an optimal commit block size so that there is not a significant degradation in performance.</p> <p>If a value of <math>\leq 1</math> or no value is specified, then each imported transaction is committed to the database one at a time.</p>
<p>Max. Errors Allowed MAX_ERRORS</p>	<p>Indicates the number of logical errors after which the transaction import process is terminated. If a value of <math>-1</math> is specified, processing does not stop, irrespective of the number of errors encountered.</p>
<p>Detail Import Message Reporting DTL_IMP_MSG_FL</p>	<p>Indicates whether error message details for each failed to be processed record should be recorded in the error file.</p>
<p>Error File Name ERROR_FILE_NM</p>	<p>Specifies the name of the XML file to which the transactions that failed to be imported are written. The file is located in the directory indicated by the configuration parameter <i>XMLImportErrorLocation</i> defined in ADV30Params.ini.</p>

<p>Save Restart Info? RESTART_FL</p>	<p>Stores information about the job so that the job can be restarted if required. In the case of a "Transaction Import," when the job is restarted, SysManUtil attempts to import all transactions in the XML file after the last checkpoint. This parameter is set to <i>true</i> by default.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, statistics are written to the log table at the end of the run. With the action "Transaction Import," the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions imported</li> <li>• Number of transactions that were successfully imported</li> <li>• Number of transactions that failed to be imported</li> </ul>
<p>Override Level OVERRIDE_LVL</p>	<p>Indicates the specific override level that should be set when the transaction or transactions are imported. Use of this parameter will apply only if the existing Apply Overrides parameter is set to "True".</p> <p>Valid values are: 0 (displays blank in Advantage), 1, 2, 3, 4, 5, 6, 7, 8, 9 and 10.</p> <p>If the Apply Override parameter is set to true and the Override Level value is zero or the parameter is not specified in the parameter file, then the Override Level of the user who is importing the transactions will be used.</p> <p>The Override Error level of the User is derived as follows: When Use Role's Override Level is not checked on the Access Control record that grants the user the privilege to override, then the system uses the Override Error level from the User Information. If the Use Role's Override Level is checked, the assigned Override Error level for the role will be used. There is one exception: for users in the Admin role, the Override Error level from the User Information table is always used. To arrive at the Access Control record that grants the user the privilege to override, a User's security roles, sorted by Precedence, along with Resource Group ID are used for lookup.</p> <p>If the value of this parameter exceeds the override level value of the user who is importing the transactions, then the user's value will be used instead. The typical use of this parameter is to specify a value that is not greater than the value for the user who is importing the transactions.</p> <p>A value of 0 for the Override Level indicates that the override level assigned to the Execution User ID will be applied.</p>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is</p>



	<p>used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
<p>Use Hierarchical XML USE_HIERCL_XML</p>	<p>The Use Hierarchical XML flag instructs the SMU Job to consume hierarchical XML. If this flag is checked then the job will consider Hierarchical Input XML.</p>
<p>Listener Name LISTENER_NAME</p>	<p>When Generate Status is set to <i>true</i>, this parameter can be used to specify a listener class that implements the SysManUtilStatsListener interface in order to perform additional logic during transaction import.</p> <p>This is an optional, action-specific parameter. Only one listener class can be specified.</p> <p>If a parameter file is used, place the LISTENER_NAME parameter after the **ACTN_CD parameter and prior to the action code's first _PARAM_LINE_ parameter.</p>

If a parameter file is not used, the LISTENER\_NAME parameter can be set up as one of the job's input parameters. Note that the System Maintenance Utility Parameters page used for configuring SysManUtil job parameters online does not support this.

## Performing Transaction Custom Actions

This action is used to perform a transaction custom action for transactions in the system that satisfy the search criteria specified. If the action code is to be specified using the parameter file, the value is *DOCOTHER*; otherwise, it is *200*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	The Transaction Type for the transactions on which the transaction custom action has to be performed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions on which the transaction custom action has to be performed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions on which the transaction custom action has to be performed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions on which the transaction custom action has to be performed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions on which the transaction custom action has to be performed. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Phase Code DOC_PHASE_CD	The Phase Code for the transactions on which the transaction custom action has to be performed. This is an optional parameter.

Transaction Unit Code DOC_UNIT_CD	The Unit Code for the transactions on which the transaction custom action has to be performed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Create User ID DOC_CREA_USID	The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Status DOC_STA_CD	The status of the transactions on which the transaction custom action has to be performed. This is an optional parameter.  The following transaction status values can be specified <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
Transaction Sub Action DOC_S_ACTN_CD	The code for the transaction custom action that will be performed.
Save Restart Info? RESTART_FL	Stores information about the job so that it can be restarted if required. When the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and the transaction custom action is performed on these. New transactions that may have entered the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job. This parameter is set to <i>true</i> by default.
Generate Statistics? GENERATE_STATS?	When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With transaction custom actions the following statistics are written: <ul style="list-style-type: none"> <li>• Total number of transactions on which the action was performed</li> <li>• Number of transactions on which the action was successfully performed.</li> <li>• Number of transactions on which the action was unsuccessfully performed.</li> </ul>
User ID USER_ID	This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.

	<p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
--	--

### Mark Transactions for Processing

This action is used to change the transaction status to *Ready* for transactions in the system that satisfies the search criteria specified. Only transactions that are in the *Draft Phase* are picked up for this action. If the action code is to be specified using the parameter file, the value is *DOCREADY*; otherwise, it is *191*.

Parameter Name	Description
Internal Parameter Name	
Transaction Type DOC_TYP	The Transaction Type for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.

Transaction Code DOC_CD	The Transaction Code for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions whose status has to be changed. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Unit Code DOC_UNIT_CD	The Transaction Unit Code for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Status DOC_STA_CD	<p>The status of the transactions whose status has to be changed. This is an optional parameter.</p> <p>The following transaction status values can be specified</p> <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
Save Restart Info? RESTART_FL	Stores information about the job so that it can be restarted if required. With this action when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and updated. New transactions that may have entered in the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job. This parameter is set to <i>true</i> by default.
Generate Statistics? GENERATE_STATS	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With this action the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions chosen</li> <li>• Number of transactions whose status was successfully changed</li> </ul>

	<ul style="list-style-type: none"> <li>• Number of transactions whose status was not changed</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>

### Hold Transactions from Processing

This action is used to change the transaction **Status** to *Held* for transactions in the system that satisfies the search criteria specified. Only transactions that are in the **Draft Phase** are picked up for this action. If the action code is to be specified using the parameter file, the value is *DOCHOLD*; otherwise, it is *192*.

Parameter Name	Description
Internal Parameter Name	

Transaction Type DOC_TYP	The Transaction Type for the transactions whose status has to be changed. The parameter is optional. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions whose status has to be changed. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Unit Code DOC_UNIT_CD	The Transaction Unit Code for the transactions whose status has to be changed. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Status DOC_STA_CD	<p>The status of the transactions whose status has to be changed. This is an optional parameter.</p> <p>The following transaction status values can be specified</p> <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
Save Restart Info? RESTART_FL	Stores information about the job so that it can be restarted if required. With this action when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and updated. New transactions that may have entered in the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job.

<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With this action the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions chosen</li> <li>• Number of transactions whose status was successfully changed</li> <li>• Number of transactions whose status was not changed</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>



## Discarding Transactions

This action is used to discard transactions in the system that satisfies the search criteria specified. The primary purpose of the discard action is to create a Cancellation Draft transaction from a *Final* version of a transaction collection. It can also be used to delete (remove from the system) a *Draft* version of a transaction collection. If the action code is to be specified using the parameter file, the value is *DOCDISCARD*; otherwise, it is *153*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	The Transaction Type for the transactions to be discarded. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions to be discarded. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions to be discarded. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions to be discarded. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions to be discarded. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Phase Code DOC_PHASE_CD	The Phase Code of the transactions to be discarded. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Status DOC_STA_CD	The Status of the transactions that must be discarded. This is an optional parameter.
Transaction Unit Code DOC_UNIT_CD	The Transaction Unit Code of the transactions to be discarded. This is an optional parameter. The "*" notation can be used to represent wild cards.

<p>Create User ID DOC_CREA_USID</p>	<p>The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.</p>
<p>Fiscal Year or Lag Days CLEANUP_PERIOD</p>	<p>The Fiscal Year/ Lag Days works as selection criteria to perform the transaction actions of discard along with or without the existing selection criteria</p> <p>For the Financial application, transaction selection for discard would be based on the Fiscal Year. The system would select the records where the Fiscal Year provided in the parameter is equivalent to the Fiscal Year (Current Fiscal Year) from the Transaction Header (doc_hdr) table. This feature is very useful for Financial to remove unprocessed transactions from the year that at the end of the fiscal year.</p> <p>For the HRM, VSS and Admin application, transaction selection for discard would be based on the Lag Days. The system would select the records where the difference between the Current Date [APPCTRL date] and the Transaction last Modified date [doc_appl_last_dt] is greater than the number of Lag Days specified by the user.</p> <p>Relevant when Action Code is <i>Discard Transaction</i> and Transaction Phase Code is <i>Draft/Conflict Draft</i>.</p>
<p>Save Restart Info? RESTART_FL</p>	<p>Stores information about the job so that it can be restarted if required. With the action "Transaction Discard," when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and discarded. New transactions that may have entered the system between the time that the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Transaction Discard," the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions discarded</li> <li>• Number of transactions that were successfully discarded</li> <li>• Number of transactions that failed to be discarded</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p>

	<p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
<p>Fiscal Year/Lag Days FY / LAG_DAYS</p>	<p>Relevant when Action Code is Discard Transaction and Transaction Phase Code is Draft/Conflict Draft or Action Code is Reject All Pending Transaction.</p> <p>For Financial application this parameter is Fiscal Year. When Fiscal Year is specified then System selects transactions with that Fiscal Year. If parameter file is being used to specify this parameter in Advantage Financial application then this parameter should be listed as FY.</p> <p>For other applications this parameter is Lag Days. When Lag Days is specified then System selects transactions where difference between the Application Control date and the Transaction last Modified date is greater than Lag Days parameter value (while calculating Application Control date is excluded and Transaction last Modified date is included). If parameter file is being used to specify this parameter for applications other than Advantage Financial then this parameter should be listed as LAG_DAYS.</p>

## Editing Transactions

This action is used to edit the transactions in the system that satisfies the search criteria specified. The primary purpose of the edit action is to create a *Modification Draft* transaction version from a *Final* version of a transaction collection. If the action code is to be specified using the parameter file, the value is *DOCEDIT*; otherwise, it is *131*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	The Transaction Type for the transactions to be edited. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions to be edited. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions to be edited. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions to be edited. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions to be edited. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Phase Code DOC_PHASE_CD	The Phase Code of the transactions to be edited. The parameter is optional.
Transaction Unit Code DOC_UNIT_CD	The Transaction Unit Code of the transactions to be edited. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Status DOC_STA_CD	The status of the transactions to be edited. This is an optional parameter.  The following transaction status values can be specified <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> </ul>

	<ul style="list-style-type: none"> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
<p>Save Restart Info? RESTART_FL</p>	<p>Stores information about the job so that it can be restarted if required. With the action “Transaction Edit,” when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and edited. New transactions that may have entered the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action “Transaction Edit,” the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions edited</li> <li>• Number of transactions that were successfully edited</li> <li>• Number of transactions that failed to be edited</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter’s User ID or the submitter’s effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job’s input parameters. The submitter’s User ID or the submitter’s effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job’s execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code’s first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between</p>

	<p>**ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
--	---

## Activating Transactions

This action is used to activate transactions in the system that satisfies the search criteria specified. If the action code is to be specified using the parameter file, the value is *DOCACTIVATE*; otherwise, it is *103*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	Transaction Type for the transactions to be activated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	Transaction Code for the transactions to be activated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	Transaction Department Code for the transactions to be activated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	Transaction ID for the transactions to be activated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	Transaction Version Number for the transactions to be activated. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Unit Code DOC_VERS_NO	Transaction Unit Code for the transactions that must be activated. This is an optional parameter. The "*" notation can be used to represent wild cards.

<p>Create User ID DOC_CREA_USID</p>	<p>The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.</p>
<p>Transaction Status DOC_STA_CD</p>	<p>The status of the transactions to be activated. This is an optional parameter.</p> <p>The following transaction status values can be specified</p> <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
<p>Save Restart Info? RESTART_FL</p>	<p>Store information about the job so that it can be restarted if required. With the action "Transaction Activate," when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and activated. New transactions that may have entered the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Transaction Activate," the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions activated</li> <li>• Number of transactions that were successfully activated</li> <li>• Number of transactions that failed to be activated</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-</p>

	<p>specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
--	--

## Deactivating Transactions

This action is used to deactivate transactions in the system that satisfies the search criteria specified. If the action code is to be specified using the parameter file, the value is *DOCDEACTIVATE* else it is *102*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	The Transaction Type for the transactions to be deactivated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions to be deactivated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	Transaction Department Code for the transactions to be deactivated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions to be deactivated. This is an optional parameter. The "*" notation can be used to represent wild cards.



<p>Transaction Version Number DOC_VERS_NO</p>	<p>The Transaction Version Number for the transactions to be deactivated. The parameter is optional. Primitive search criteria using "&gt;" or "&lt;" can be used. For example, you can specify "&gt;2" to get transaction versions greater the second version.</p>
<p>Transaction Unit Code DOC_UNIT_CD</p>	<p>The Transaction Unit Code for the transactions that must be deactivated. This is an optional parameter. The "*" notation can be used to represent wild cards.</p>
<p>Create User ID DOC_CREA_USID</p>	<p>The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.</p>
<p>Transaction Status DOC_STA_CD</p>	<p>The status of the transactions to be deactivated. This is an optional parameter.</p> <p>The following transaction status values can be specified</p> <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
<p>RESTART_FL</p>	<p>Stores information about the job so that it can be restarted if required. With the action "Transaction Deactivate," when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and deactivated. New transactions that may have entered the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Transaction Deactivate," the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions deactivated</li> <li>• Number of transactions that were successfully deactivated</li> <li>• Number of transactions that failed to be deactivated.</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p>

	<p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
--	--

## Submitting Transactions

This action is used to submit one or more transactions in the system that satisfies the search criteria specified. If the action code is to be specified using the parameter file then the value is *DOCSUBMIT* else it is *162*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	The Transaction Type for the transactions to be submitted. This is an optional parameter. The "*" notation can be used to represent wild cards.

Transaction Code DOC_CD	The Transaction Code for the transactions to be submitted. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions to be submitted. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions to be submitted. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions to be submitted. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Unit Code DOC_UNIT_CD	The Transaction Unit Code for the transactions to be submitted. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Status DOC_STA_CD	<p>The status of the transactions to be submitted. If this parameter is not specified, SysManUtil automatically submits all transactions in the <i>Ready</i> status.</p> <p>The following transaction status values can be specified:</p> <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
Create User ID DOC_CREA_USID	The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.
Exception Report Indicator EXCEP_REP_IND	<p>Specifies the type of exception report to generate. Four types of exception reports can be generated:</p> <ul style="list-style-type: none"> <li>• DETAILED</li> <li>• FAILED_DOCS</li> <li>• PROCESSED_DOCS</li> <li>• FAILED_LINES</li> </ul>

	<ul style="list-style-type: none"> <li>DOCUMENT_STATUS</li> </ul>
Exception Report File Name EXCEP_REP_FILE_NM	The file name of the exception report. This parameter is required when the Exception Report Indicator (above) is specified. The file will be written to the directory specified by the configuration parameter <i>XMLExportFileLocation</i> defined in <i>ADV30Params.ini</i> .
Max. Number of Messages. DOC_STA_MAX_MSG_NO	The max number of detailed error messages for each transaction error in the Status Exception Report. This value is only required when DOCUMENT_STATUS report type is chosen.
Save Restart Info? RESTART_FL	Stores information about the job so that it can be restarted if required. With the action "Transaction Submit," when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and submitted. New transactions that may have entered the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job. This parameter is set to <i>true</i> by default.
Generate Statistics? GENERATE_STATS	When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Transaction Submit," the following statistics are written: <ul style="list-style-type: none"> <li>Total number of transactions submitted</li> <li>Number of transactions that were successfully submitted</li> <li>Number of transactions that failed to be submitted</li> </ul>
Transaction Phase Code DOC_PHASE_CD	The status of the transactions to be submitted. If this parameter is not specified, SysManUtil automatically submits all transactions in the <i>Draft</i> Code.  The following transaction Phase Code values can be specified: <ul style="list-style-type: none"> <li><i>Draft</i> (Value: 1)</li> <li><i>Pending</i> (Value: 2)</li> </ul>
User ID USER_ID	This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.  Whether a parameter file is used or not, use of this parameter is optional.  When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will

	<p>be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p> <p>This parameter has no default value.</p>
<p>Submit Progression Counter Size DOCSUB_PROG_CTR_SZ</p>	<p>Optional field and relevant for Action Code of Submit Transaction. Regardless of the setting of the Generate Statistics parameter, generates progression statistics of processed transactions after processing specified number of transactions. If not specified or if value is not a positive integer, then SMU will utilize the default setting of 250. This parameter is exceptional such that it can be specified outside of the parameter file and still be used.</p>
<p>Exception Severity Flag EXP_SEV_FL</p>	<p>Optional field. When selected, writes down the severity of error in exception report.</p>
<p>Listener Name LISTENER_NAME</p>	<p>When Generate Status is set to <i>true</i>, this parameter can be used to specify a listener class that implements the SysManUtilStatsListener interface to perform additional logic during transaction submission.</p> <p>This is an optional, action-specific parameter. Only one listener class can be specified.</p>

	<p>If a parameter file is used, place the LISTENER_NAME parameter after the **ACTN_CD parameter and prior to the action code's first _PARAM_LINE_ parameter.</p> <p>If a parameter file is not used, the LISTENER_NAME parameter can be set up as one of the job's input parameters. Note that the System Maintenance Utility Parameters page used for configuring SysManUtil job parameters online does not support this.</p>
--	--

## Validating Transactions

This action is used to validate one or more transactions in the system that satisfies the search criteria specified. If the action code is to be specified using the parameter file then the value is *DOCVALIDATE*; otherwise, it is *161*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	The Transaction Type for the transactions to be validated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions to be validated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions to be validated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction ID DOC_ID	The Transaction ID for the transactions to be validated. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions to be validated. The parameter is optional. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Status DOC_STA_CD	<p>The status of the transactions to be validated. If this parameter is not specified, SysManUtil automatically validates all transactions in the <i>Ready</i> status.</p> <p>The following transaction status values can be specified:</p>

	<ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
Transaction Unit Code DOC_UNIT_CD	The Transaction Unit Code for the transactions that must be submitted. This is an optional parameter. The "*" notation can be used to represent wild cards.
Create User ID DOC_CREA_USID	The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.
Exception Report Indicator EXCEP_REP_IND	Specifies the type of exception report to generate. Four types of exception reports can be generated: <ul style="list-style-type: none"> <li>• DETAILED</li> <li>• FAILED_DOCS</li> <li>• PROCESSED_DOCS</li> <li>• FAILED_LINES</li> </ul>
Exception Report File Name EXCEP_REP_FILE_NM	The file name of the exception report. This parameter is required when the Exception Report Indicator (above) is specified. The file will be written to the directory specified by the configuration parameter <i>XMLExportFileLocation</i> defined in <i>ADV30Params.ini</i> .
Max. Number of Messages. DOC_STA_MAX_MSG_NO	The max number of detailed error messages for each transaction error in the Exception Report. This value only appears when DOCUMENT_STATUS type report is chosen.
Save Restart Info? RESTART_FL	Stores information about the job so that it can be restarted if required. With the action "Transaction Validate," when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint are chosen and validated. New transactions that may have entered the system between the time the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job.
Generate Statistics? GENERATE_STATS	When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Transaction Validate," the following statistics are written: <ul style="list-style-type: none"> <li>• Total number of transactions validated</li> </ul>

	<ul style="list-style-type: none"> <li>• Number of transactions that were successfully validated</li> <li>• Number of transactions that failed to be validated</li> </ul>
<p>Transaction Phase Code DOC_PHASE_CD</p>	<p>The status of the transactions to be validated. If this parameter is not specified, SysManUtil automatically submits all transactions in the <i>Draft</i> Code.</p> <p>The following transaction Phase Code values can be specified:</p> <ul style="list-style-type: none"> <li>• <i>Draft</i> (Value: 1)</li> <li>• <i>Pending</i> (Value: 2)</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p>



If the USER\_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.

## Printing Transactions

The Print Transaction action is used to print multiple transactions on the Transaction Catalog via batch. The action code is to be specified using the parameter file with a value of *DOC\_ACTN\_PRINT*.

Parameter Name Internal Parameter Name	Description
Generate Statistics? GENERATE_STATS	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Transaction Validate," the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions printed</li> <li>• Number of transactions that were successfully printed</li> <li>• Number of transactions that failed to be printed</li> </ul>
User ID USER_ID	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>This parameter is optional.</p> <p>This parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between</p>

	<p>**ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>
<p>Verbose Messages for Parameter File</p> <p>PARM_FIL_VERBOSE</p>	<p>Optional field. Controls whether a parameter file's information is displayed in the job log. If set to "N" a message is not generated. If set to "Y" or any other value, parameter line information in the parameter file will be displayed in the job log. This parameter is exceptional such that it can be specified outside of the parameter file and still be used.</p>
<p>Print Job</p>	<p>This is a required parameter, used to retrieve the FPAC entry.</p>
<p>Number of Copies</p>	<p>This is a required parameter that defaults to 1. The value must be a positive integer. This parameter defines how many copies of each selected transaction to print.</p>
<p>Print Resource</p>	<p>This is a required parameter, which is used to retrieve the FPAC entry.</p>
<p>Print on Both Sides</p>	<p>This is a required parameter that defaults to False.</p>
<p>Application Resource</p>	<p>This is an optional parameter used to retrieve the FPAC entry when the Application Resource on FPAC is something other than the transaction code. If the parameter value is left blank the Transaction Code will be used for FPAC lookup.</p>
<p>Print User ID</p>	<p>This is an optional parameter. If not entered, the value defaults to the User ID of the user running the job. This parameter is used to retrieve the FPAC entry, not for security purposes.</p>
<p>View Form</p> <p>VIEW_FORM</p>	<p>This is an optional parameter. If provided as <i>true</i>, then it is used to populate details on the View Forms (FORMS) page to view the printed PDF file.</p>
<p>Download PDF</p> <p>DOWNLOAD_PDF</p>	<p>This is an optional parameter. This parameter is considered only if the View Form (VIEW_FORM) parameter is set to <i>true</i>.</p> <p>This parameter indicates if the generated PDF file from the Forms Printing Server will be downloaded to the Advantage Server. If the value is set to <i>true</i>, then it will download the file and if set to <i>false</i> or left blank, then the process will continue printing other transactions from the parameter file without downloading the file.</p> <p><b>Note:</b></p>

	<p>Setting to <i>true</i> will add a delay, because the process needs to wait for the PDF file to be generated and available to download from the Forms Printing Server.</p> <p>Setting to <i>false</i>, the user can still access the PDF file from the View Forms (FORMS) page, if the View Form (VIEW_FORM) parameter of the job is set to <i>true</i>.</p>
Form Description FORM_DSCR	This is an optional parameter. If provided, then it is used to populate details on the View Forms (FORMS) page to view the printed PDF file.
Transaction Type DOC_TYP	The <b>Transaction Type</b> for the transactions to be printed. This is an optional parameter.
Transaction Code DOC_CD	The <b>Transaction Code</b> defines the Transaction Code that will be selected for printing. Only one value can be entered in this field. This is a required parameter. Wildcards cannot be used.
Transaction Department Code DOC_DEPT_CD	The <b>Transaction Department Code</b> for the transactions to be printed. This is an optional parameter. The "*" notation can be used to represent wildcards. If left blank, all Departments are selected.
Transaction ID DOC_ID	The Transaction ID for the transactions to be printed. This is an optional parameter. The "*" notation can be used to represent wild cards. If left blank, all Transaction ID's are selected.
Transaction Version Number DOC_VERS_NO	The <b>Transaction Version Number</b> for the transactions to be printed. The parameter is optional, and must be a positive integer. Wildcard values are not allowed. If left blank, the most current version is selected.
Transaction Unit Code DOC_UNIT_CD	The <b>Transaction Unit Code</b> for the transactions that must be printed. This is an optional parameter. The "*" notation can be used to represent wildcards. If left blank, all unit codes for the selected department codes are selected.
Transaction Phase Code DOC_PHASE_CD	<p>The Phase of the transactions to be printed. If this parameter is not specified, all Transaction Phases are selected. Wildcard values are not allowed.</p> <p>The following transaction Phase Code values can be specified:</p> <ul style="list-style-type: none"> <li>• <i>Draft</i> (Value: 1)</li> <li>• <i>Pending</i> (Value: 2)</li> </ul>

	<ul style="list-style-type: none"> <li>• <i>Final</i> (Value: 3)</li> <li>• <i>Historical</i> (Value: 4)</li> <li>• <i>Conflict Draft</i> (Value: 5)</li> </ul>
Transaction Status DOC_STA_CD	<p>The status of the transactions to be printed. If this parameter is not specified all Transaction Statuses will be selected.</p> <p>The following transaction status values can be specified:</p> <ul style="list-style-type: none"> <li>• <i>Ready</i> (Value: 2)</li> <li>• <i>Held</i> (Value: 1)</li> <li>• <i>Rejected</i> (Value: 3)</li> </ul>
Create User ID DOC_CREA_USID	<p>The <b>ID</b> of the user who created the transaction. This is an optional parameter. The “*” notation can be used to represent wild cards.</p>

### Rejecting All Pending Transactions

This action is used to discard transactions in the system that satisfies the search criteria specified. The primary purpose of the discard action is to create a Cancellation Draft transaction from a *Final* version of a transaction collection. It can also be used to delete (remove from the system) a *Draft* version of a transaction collection. If the action code is to be specified using the parameter file, the value is *DOCDISCARD*; otherwise, it is *153*.

Parameter Name Internal Parameter Name	Description
Transaction Type DOC_TYP	The Transaction Type for the transactions to be rejected. This is an optional parameter. The “*” notation can be used to represent wild cards.
Transaction Code DOC_CD	The Transaction Code for the transactions to be rejected. This is an optional parameter. The “*” notation can be used to represent wild cards.
Transaction Department Code DOC_DEPT_CD	The Transaction Department Code for the transactions to be rejected. This is an optional parameter. The “*” notation can be used to represent wild cards.

Transaction ID DOC_ID	The Transaction ID for the transactions to be rejected. This is an optional parameter. The "*" notation can be used to represent wild cards.
Transaction Version Number DOC_VERS_NO	The Transaction Version Number for the transactions to be rejected. The parameter is rejected. Primitive search criteria using ">" or "<" can be used. For example, you can specify ">2" to get transaction versions greater the second version.
Transaction Phase Code DOC_PHASE_CD	The Phase Code of the transactions to be rejected. This is a non-editable parameter and has defaulted value as <i>Pending</i> .
Transaction Status DOC_STA_CD	The Status of the transactions that must be rejected. This is an optional parameter.
Transaction Unit Code DOC_UNIT_CD	The Transaction Unit Code of the transactions to be rejected. This is an optional parameter. The "*" notation can be used to represent wild cards.
Create User ID DOC_CREA_USID	The ID of the user who created the transaction. This is an optional parameter. The "*" notation can be used to represent wild cards.
Fiscal Year or Lag Days CLEANUP_PERIOD	<p>The Fiscal Year/ Lag Days works as selection criteria to perform the transaction actions of reject pending transactions along with or without the existing selection criteria</p> <p>For the Financial application transaction selection for rejection would be based on the Fiscal Year. The system would select the records where the Fiscal Year provided in the parameter is equivalent to the Fiscal Year (Current Fiscal Year) from the Transaction Header (doc_hdr) table. This feature is very useful for Financial to remove unprocessed transactions from the year that at the end of the fiscal year.</p> <p>For the HRM, VSS and Admin application transaction selection for rejection would be based on the Lag Days. The system would select the records where the difference between the Current Date [APPCTRL date] and the Transaction last Modified date [doc_appl_last_dt] is greater than the number of Lag Days specified by the user.</p> <p>Relevant when Action Code is <i>Reject All Pending Transaction</i>.</p>
Save Restart Info?	Stores information about the job so that it can be restarted if required. With the action 'Reject All Pending Transaction', when the job is restarted, only those transactions satisfying the search criteria greater than the stored checkpoint

<p>RESTART_FL</p>	<p>are chosen and rejected. New transactions that may have entered the system between the time that the job failed and was restarted are also chosen if they meet the selection criteria specified as parameters to the job.</p>
<p>Generate Statistics? GENERATE_STATS</p>	<p>When this parameter is specified, SysManUtil writes statistics to the log at the end of the run. With the action "Reject All Pending Transaction", the following statistics are written:</p> <ul style="list-style-type: none"> <li>• Total number of transactions rejected</li> <li>• Number of transactions that were successfully rejected</li> <li>• Number of transactions that failed to be rejected</li> </ul>
<p>User ID USER_ID</p>	<p>This parameter allows the specified User ID and not the submitter's User ID or the submitter's effective User ID to execute the submitted job. The use of this parameter depends on whether a parameter file is used to specify the job's input parameters. The submitter's User ID or the submitter's effective User ID is known as the execution User ID.</p> <p>Whether a parameter file is used or not, use of this parameter is optional.</p> <p>When a parameter file is used, this parameter can be specified zero or more times in the file. Where it appears determines if the User ID will be considered a global User ID or an action-specific User ID. Just as specifying a USER_ID overrides the job's execution User ID, it is possible that an action-specific User ID can override a global User ID. Thus, there can be a global User ID specified or not specified, and each separate action code can have an action-specific User ID specified or not specified. And if a particular action code does not have an action-specific User ID and if the parameter file does not have a global User ID, the execution User ID will be used for this particular action code.</p> <p>Within the parameter file, specifying a global User ID requires the USER_ID parameter to be the first parameter specified in the parameter file. That is, the USER_ID must be physically located just prior to the first **ACTN_CD parameter within the file.</p> <p>On the other hand, specifying an action-specific User ID requires the USER_ID parameter to be located after the applicable **ACTN_CD parameter and prior to the action code's first _PARAM_LINE parameter. That is, the USER_ID must be physically located in between **ACTN_CD and the next _PARAM_LINE_. The USER_ID cannot be located within a _PARAM_LINE_ block.</p> <p>If a parameter file is not used, then the job's input parameters are specified either as a regular job or on the custom parameter screen, as mentioned in a previous section. In either case the specified USER_ID is the User ID that will be used in place of the execution User ID.</p> <p>If the USER_ID parameter is not specified, then the execution User ID will continue to be used for the execution of this job.</p>

## Approve Transaction

This action is used to approve transactions in the system. The primary purpose of the approve action is to automatically approve pending records/transactions from the worklist through an offline batch process.

Parameter Name Internal Parameter Name	Description
Action ACTN_CD	The Action code which instructs the program to perform a certain action on its records. This field is required.
Approver ID APRV_ID	The User ID which will be used to select and approve the pending records from the worklist. This field is required.
Approval Role APRV_ROL	When entered, records from the specified Approval Role worklist are selected for approval by the job. If the parameter is left empty, the job selects the records from all the worklists assigned to the user. This is an optional field.
Transaction Code DOC_CD	When entered, the values are used in the selection logic for which transactions are selected for approval. This is an optional field.
Transaction Creation User ID DOC_CREA_USID	The ID of the user who created the transaction. This is an optional field.
Transaction Department DOC_DEPT_CD	When entered, the values entered are used in the selection logic for which transactions are selected for approval. This is an optional field.
Transaction ID DOC_ID	When entered, the values are used in the selection logic for which transactions are submitted. This is an optional field.
Transaction Status DOC_STA	When entered, the values entered are used in the selection logic for which transactions are submitted. This is an optional field.
Transaction Version Number DOC_VERS_NO	When entered, the values are used in the selection logic for which transactions are submitted. This is an optional field.

<p>Approval Level APRV_LVL</p>	<p>This parameter is used to select and approve the transaction at the specified Approval Level. This is a required field.</p>
<p>Apply Overrides APPLY_OVERRIDES</p>	<p>This parameter is used to decide if overrides have to be applied on the transaction before applying the approval. This is an optional field.</p>
<p>Exception Report Indicator EXCEP_REP_IND</p>	<p>This parameter defines the level of detail of the exception report. Valid values are:</p> <ul style="list-style-type: none"> <li>1 = Detailed</li> <li>2 = Failed Transactions</li> <li>3 = Processed Transactions</li> <li>4 = Failed Transaction Lines</li> <li>5 = Transaction Status</li> </ul>
<p>Exception Report File Name EXCEP_REP_FILE_NM</p>	<p>This parameter defines what file the program is to use to create an Exception Report. This is a required field if EXCEP_REP_IND is specified.</p>
<p>Save Restart Information? RESTART_FL</p>	<p>Stores information for the job such that the job can be restarted if required. In the case of a transaction approve action, if the job is being restarted for only those transactions that satisfy the search criteria greater than the checkpoint stored will be selected and approved. Please note that new transactions that enter the system between the time the job failed and was restarted will also be approved if they meet the selection criteria specified.</p>
<p>Progression Counter Size DOCSUB_PROG_CTR_SZ</p>	<p>The Progression Counter Size controls the interval at which progression messages are generated in the job log during transaction approval processing. These intermediate messages are displayed periodically based on the parameter's value until the action completes. The parameter's value should be positive if entered. If not specified or incorrectly specified, then processing uses a default value of 250.</p>



## System Maintenance Utility Listener

This section lists the available listeners that can be used with the System Maintenance Utility along with the related Application Parameters (APPCTRL).

- [Listener](#)
- [Application Parameter \(APPCTRL\)](#)

### Listener

This is a list of available listeners. Additional listener classes can be added, if they implement one of the supported listener interfaces (TBLARCHIVListener or SysManUtilStatsListener) and are used with the corresponding action.

Listener Class	Description
advantage.BiEvtStatsListener	<p>Can be used with table import, table overlay, transaction import, and transaction submission actions.</p> <p>Use this listener to update the import and submit counts on the Batch Interface Event (BIEVNT) record. When used, the following parameters must be specified:</p> <p>Job Name (JOB_NAME): The job name needs to follow this format so that the listener can identify the BIEVNT record to update:</p> <p>Department + Unit + date (yyyymmdd) + tablename/transcode + Batch ID + "-Import_/Submit_" + threadCount + "_" + CatalogID</p> <p>Also see TBL_IMP_CTLG_ID, TBL_OVLY_CTLG_ID, and INTF_USR_PREFIX under the APPCTRL listing.</p>

### Application Parameter

This is a list of Application Parameter (APPCTRL) records used with the above listeners.

Parameter Name	Description
INTF_USR_PREFIX	<p>Used by BiEvtStatsListener.</p> <p>Indicates the prefix of the interface User ID performing the SysManUtil action. If the SysManUtil job User ID does not begin with this prefix, then BiEvtStatsListener does not update the Batch Interface Event (BIEVNT) import or submit counts.</p>
TBL_IMP_CTLG_ID	Used by BiEvtStatsListener.

	<p>Indicates the Catalog ID of the SysManUtil job used to import records into a table.</p> <p>For example, a dedicated SysManUtil job can be set up that will perform table import actions with the LISTENER_NAME parameter and set the Catalog ID of the dedicated job here.</p>
TBL_OVLY_CTLG_ID	<p>Used by BiEvtStatsListener.</p> <p>Indicates the Catalog ID of the SysManUtil job used to overlay records in a table.</p> <p>For example, a dedicated SysManUtil job can be set up that will perform the table overlay action with the LISTENER_NAME parameter and set the Catalog ID of the dedicated job here.</p>

## ADV Job Interaction Client

This section describes the Job Interaction Client which is an interface to the job processing engine of the CGI Advantage application that:

- Can be used to submit jobs along with any required parameter values. This information is supplied to the job interaction client through a driver data file in a specific format.
- Does not require you to log in to the application through a web browser. All job framework commands are sent using the job interaction client from the command line.
- Can be used to control the job framework engine, such as starting and stopping regular job polling by the Job Manager. This periodic polling is used for picking up jobs that have been submitted by users through the online system and have met the scheduled criteria for job execution.

The Job Interaction Client is supported through an EJB client Java class, `AMSJobInteractionClient.class`, that accesses the Job Manager EJB on the target VLS to provide the functionality outlined above. A number of parameters (both required as well as optional) can be passed to the job using the Job Interaction Client.

The recommended way of invoking the Job Interaction Client is through either a batch file (Windows platform) or a shell script (AIX and Linux platform). The script file sets up the required Java classpath and facilitates the passing of parameters to the Job Interaction Client. The script file acts as a driver for executing the actual Job Interaction Client Java class. Template script files for both platforms are included within the Job Interaction Client folder and can be used as a starting point for creating a site specific script. These template script files, `runclient.bat` and `runclient.sh`, are described in more detail in the following sections.

Note: The flags under the Actions section on the Access Control page determine whether a user assigned to a Resource Group/Security Role combination can perform the following actions: Approve (Approve Job), Schedule (Schedule Job), Edit (Edit Job), Submit (Submit Job), Discard (Delete Job), Kill Job, Restart Job, View Completed Jobs, View Pending Jobs, View Others' Jobs, View Job Log, and View Job Report. Security is enforced for these actions irrespective of the medium through which they are executed (that is, through the Run Batch (BATRUN) page or through the Job Interaction Client). For the Administrator (ADMN) Role the system does not check the Access Control entries and there is no need to setup Access Control entries for this. If a job resource does not have a Resource ID defined on the Setup Batch Job (BATSETUP) page then only the users associated with the ADMN role can access it and perform actions on it.

## Setup

The location for JIC files is different depending on if you are within or outside the container. The JIC command is actually executed within the container and you can manipulate DAT files from outside the container.

- Within Container = `/apps/CGIADV/<ADV_name>/Custom/JICScripts`
- Persistent (Outside) Container = `<NFSLocation>/RTFiles/<Namespace>/<ADV_name>/Custom/JICScripts`

Where:

- `<NFSLocation>` = The NFS persistent storage area for the application.

- <Namespace> = The namespace for the Advantage applications. For example, prod.
- <ADV\_Name> = admin, fin, vss, pb, hr

Note: Ensure that all directories and files created under the JICScripts directory have the correct permissions and ownership so that the container's account (185) can access everything.

```
sudo chmod -R 775 ./*
```

```
sudo chown -R 185:<group for deploy account> ./*
```

## Driver Data File

The driver data file is a text file with a required “.dat” file extension that is ‘interpreted’ by the Job Interaction Client to submit specified jobs through a connection established to a specific VLS. The driver data file consists of a series of action blocks. Each action block is denoted by the “\*\*JOB” identifier tag followed by a value. The value is either used to submit a job or it identifies a control block that triggers an administrative function such as stopping online job polling or resetting the current *SleepInterval* or *MaxCheckLoop* value initialized from the INI file. A “\*\*JOB” value of 0 (zero) indicates a control block.

Each action block can have one or more parameters that are placed after the delimiter “\_PARAM\_LINE\_”. The delimiter tag is required regardless of whether or not the action block has any parameters. Parameters are supplied as name-value pairs.

The Job Interaction Client processes action blocks sequentially. By default, an error in the execution of any of the action blocks stops the Job Interaction Client from executing any further. However, this default behavior can be overridden as described later. The three types of action blocks are control blocks, job blocks and chain job blocks. A driver data file starts with at least one control block that identifies the specific Job Manager EJB and the target VLS followed by a series of job or chain job blocks. The driver data file can also end with a control block to restore the Job Manager properties so that it starts processing jobs submitted via normal online mode. A sample driver data file, NC.dat, is shown below:

```
**JOB=0
PARAM_LINE
CONTROL_TYPE=PROPERTIES
JOB_MANAGER_NAME=JobManager1
SLEEP_INTERVAL=5000
MAX_CHECK_LOOP=100
LOAD_BAL_MODE=true

**JOB=0
PARAM_LINE
CONTROL_TYPE=ACTION
ACTION=STOP_QUEUE

**JOB=3
PARAM_LINE
TBL_NM=IN_PAGES
ACTN_CD=202
COMMIT_BLOCK=1000
FILE_NM =IN_PAGES_DATA.xml

**JOB=385
CLIENT_NM = City of Greenville
SUCCESS_CODE = 12
PARAM_LINE
STEP_ID = 387
DOC_CD = 101

**JOB=0
PARAM_LINE
CONTROL_TYPE=ACTION
ACTION=START_QUEUE
```

*Sample data file NC.dat*

The sample driver data file shown here contains five action blocks that accomplish the following:

- Action Block 1: Opens a connection to the Job Manager EJB identified by the Job Manager Name and Id.
- Action Block 2: Stops the online job polling for the current Job Manager (the connection established in previous block). This sets up the associated VLS to process only jobs 'pushed' through the current driver data file.
- Action Block 3: Schedules a job (catalog Id = 3) and also provides values for the following parameters **TBL\_NM**, **ACTN\_CD**, **COMMIT\_BLOCK**, and **FILE\_NM**.
- Action Block 4: Schedules a chain job (catalog Id = 385). For one of the chain job elements (catalog Id = 387), the parameter **DOC\_CD** takes a value *101* for the run.
- Action Block 5: Starts the online job polling for the current Job Manager.

At the end, the resources associated with the current Job Interaction Client connection to the target Job Manager EJB are released and the connection is closed.

## Control Block

Control blocks are used to manage the target Job Manager properties or the local Job Interaction Client runtime settings. Each control block starts with the tag “\*\*JOB=0”. Additionally, the type of control action to be triggered is further specified by a required parameter **CONTROL\_TYPE** and its value. A sample control block is given below:

```

**JOB=0
_PARAM_LINE_

CONTROL_TYPE=PROPERTIES
JOB_MANAGER_NAME=JobManager2
SLEEP_INTERVAL=5000
MAX_CHECK_LOOP=100
    
```

Sample Control Block in Data File

The following **CONTROL\_TYPE** parameter values are supported:

- PROPERTIES
- ACTION
- VARIABLE

### Control Type: Properties

A Control block with the parameter **CONTROL\_TYPE** having the value *PROPERTIES* is used to initialize runtime settings for the Job Interaction Client. The following property names and values for “PROPERTIES” control blocks are supported:

Parameter Name	Acceptable Value	Comments
JOB_MANAGER_NAME	Text	Identifies JNDI name for the target Job Manager EJB. For WebSphere 8.x server clients, this connotes the fully qualified name of the form: cell/nodes/ <nodeName>/ servers/ <serverName>/ <relativeName>
SLEEP_INTERVAL	Number	Time interval, in milliseconds, between successive inquiries by the Job Interaction Client for a running job’s status.

MAX_CHECK_LOOP	Number	The maximum number of iterations (run status inquiry cycles) the Job Interaction Client will perform for any job.
LOAD_BAL_MODE	“true” or “false” (default false)	Specifies if the job should be submitted in load-balanced mode allowing all active eligible Job Managers a chance to pick up the job. If false, the job is run by the job manager connected.

*Property Names and Values for Control Blocks with parameter CONTROL\_TYPE=PROPERTIES*

A sample control block with CONTROL\_TYPE=PROPERTIES is given below:

```

**JOB=0
_PARAM_LINE_
CONTROL_TYPE=PROPERTIES
JOB_MANAGER_NAME=JobManager2

SLEEP_INTERVAL=5000
MAX_CHECK_LOOP=100
        
```

Specifies JNDI name of Job Manager to connect to.

*Sample Control Block with CONTROL\_TYPE=PROPERTIES*

### Control Type: Action

Control blocks with a parameter value of **ACTION** indicate that either a supported Job Manager action call is to be made or a local batch file or shell script is to be run. This is specified by providing a valid value for an additional parameter ACTION. For example, when load balancing is turned off, it may be desirable to stop job polling by the targeted Job Manager when executing jobs from the driver data file. This can be done by setting the parameter **ACTION** to have the value *STOP\_QUEUE*. Additionally, the value *START\_QUEUE* restarts the job polling usually specified at the end of the driver data file.

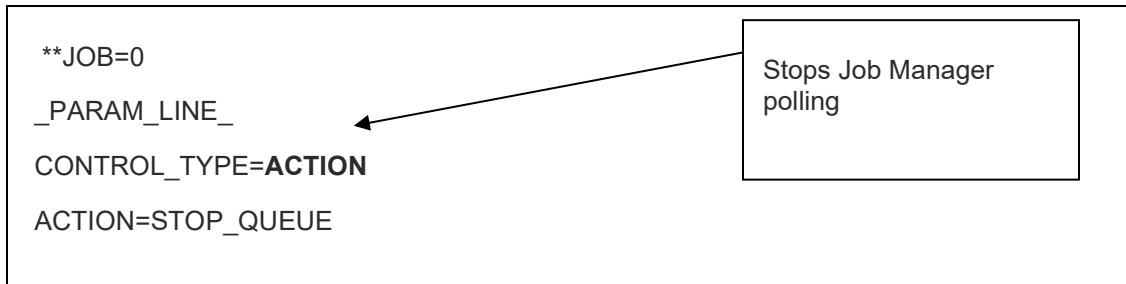
Batch files or shell scripts can be executed by setting the **ACTION** parameter value to *RUN\_SCRIPT*. The name of the shell script to run and the wait mode are parameters that must also be specified when using this type of action. The following table lists the supported parameters and their values for **ACTION** control blocks.

Parameter Name	Acceptable Value	Comments
ACTION	“STOP_QUEUE” “START_QUEUE” “RUN_SCRIPT”	STOP_QUEUE - stops job polling START_QUEUE - starts job polling RUN_SCRIPT - runs a batch file or a shell script

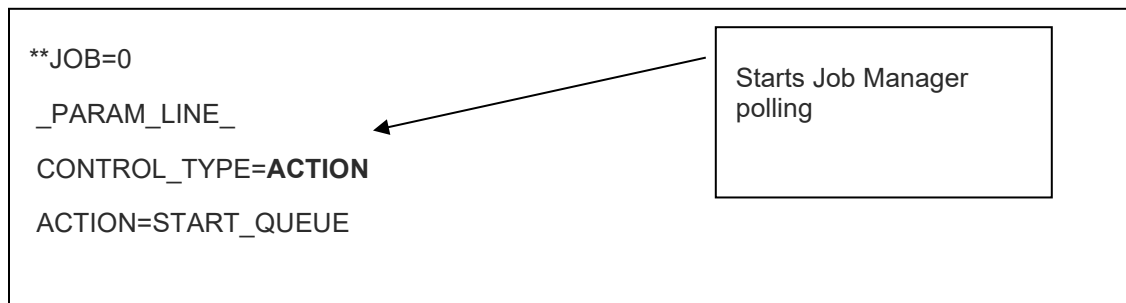
		Note that STOP_QUEUE and START_QUEUE require the user to be a member of the JOBCTRL security role or the ADMN security role.
SCRIPT_NAME	The path and name of the script file.	Name of the batch file or shell script to be run. Applicable only for ACTION = RUN_SCRIPT.
WAIT_MODE	"true" or "false"	Indicates if the Job Interaction client should wait for the script to finish running or proceed with processing the next block. Applicable only for ACTION = RUN_SCRIPT.

*Property Names and Values for Control Blocks with CONTROL\_TYPE=ACTION*

Three sample control blocks with CONTROL\_TYPE=ACTION are given below:

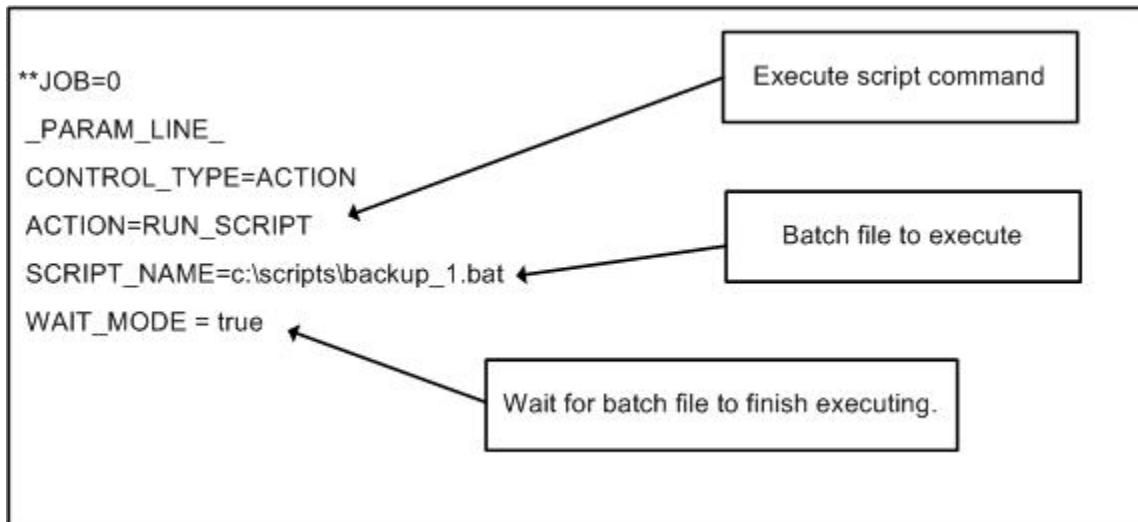


*Control Block (of type ACTION) that stops online job polling by the target Job Manager*



*Control Block (of type ACTION) that starts regular job polling by the target Job Manager*



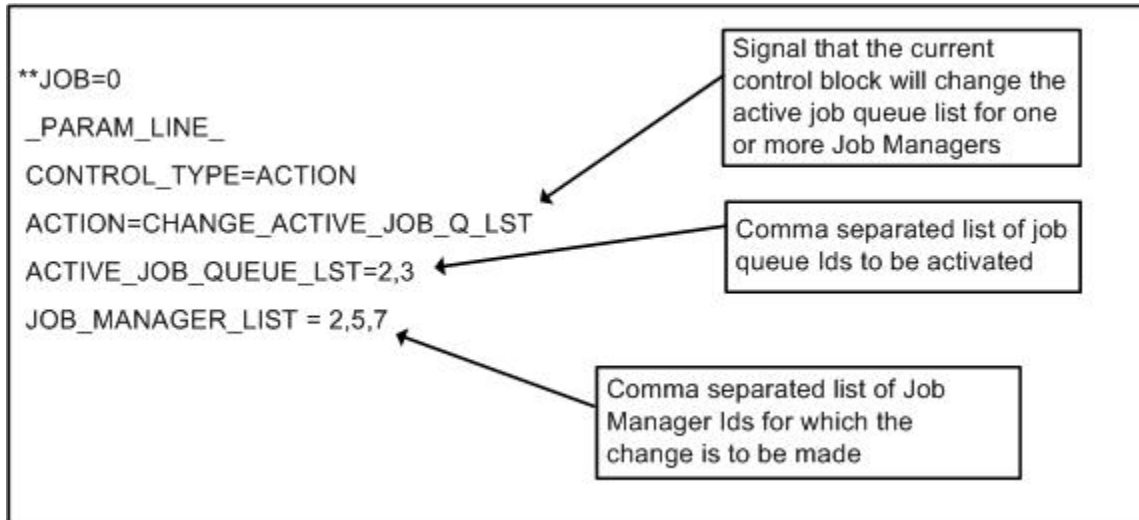


*Control Block (of type ACTION) that runs a local batch file*

Control blocks with a parameter value of **ACTION** are also used to manage active job queues on the server. The control block name value pair, ACTION = ACTIVE\_JOB\_Q\_LST is required for this purpose. Additional parameters are required to define the job queue Ids to activate as well as the list of Job Managers on which the change should be affected.

Parameter Name	Acceptable Value	Comments
ACTIVE_JOB_Q_LST	Comma separated list of job queues (Ids) to be activated (for example, 2,3).	Activating one or more of the job queues on the identified Job Manager is applicable only when the jobs are being submitted in the load-balancing mode and the job polling is on (START_QUEUE). Note that this parameter requires the user to be a member of the JOBCTRL security role or the ADMN security role.
JOB_MANAGER_LIST	Comma separated list of Job Manager Ids for which the active job queue list should be changed (for example, 2,5,7,8).	Each Job Manager in the system is assigned a unique Id. Note that this parameter requires the user to be a member of the JOBCTRL security role or the ADMN security role.

*Property Names and Values for Control Blocks with CONTROL\_TYPE=ACTION and ACTION=CHANGE\_ACTIVE\_JOB\_Q\_LST*

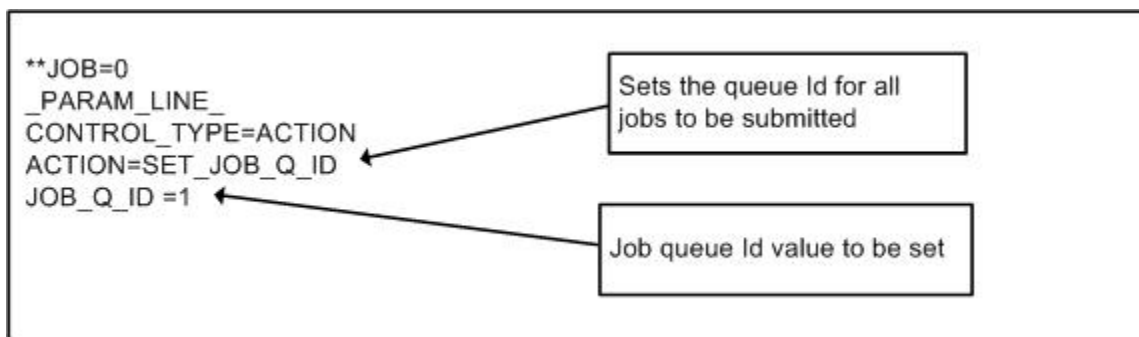


*Control Block (of type ACTION) that sets the activates one or more job queues on identified Job Managers*

By default, all jobs submitted through the job interaction client belong to the queue Id that corresponds to nightly cycle jobs. However, a control block described next can reset this value. The **ACTION** value for setting the queue Id for submitted jobs is *SET\_JOB\_Q\_ID*. An additional parameter, *JOB\_Q\_ID* is required that provides the value of the queue Id that is assigned to all subsequent jobs.

Parameter Name	Acceptable Value	Comments
JOB_Q_ID	Number	The value that identifies the job queue Id to be assigned to jobs submitted after this control block is processed.

*Property Names and Values for Control Blocks with CONTROL\_TYPE=ACTION and ACTION=SET\_JOB\_Q\_ID*

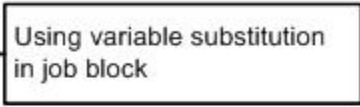


*Control Block (of type ACTION) that sets the job queue Id*

## Control Type: Variable

Control blocks with the parameter ACTION = VARIABLE are used to define constants at the beginning of a driver data file to be used throughout the file. These constants can be used to pass values to one or more job parameters in the data file. A sample data file section shows how this can be used. The required format is “%%CONSTANT\_NAME%% = value”.

```
**JOB=0
_PARAM_LINE_
CONTROL_TYPE=VARIABLE
%%CURRENT_DATE%% = 01/01/2002
%%CURRENT_FISCAL_YEAR%% = 2002
%%CURRENT_FISCAL_PER%% = 04
..
..
..
**JOB=85
_PARAM_LINE_
CLIENT_NM = City of Greenville
CURRENT_FY = %%CURRENT_FISCAL_YEAR%%
CURRENT_DT = %%CURRENT_DATE%%
```



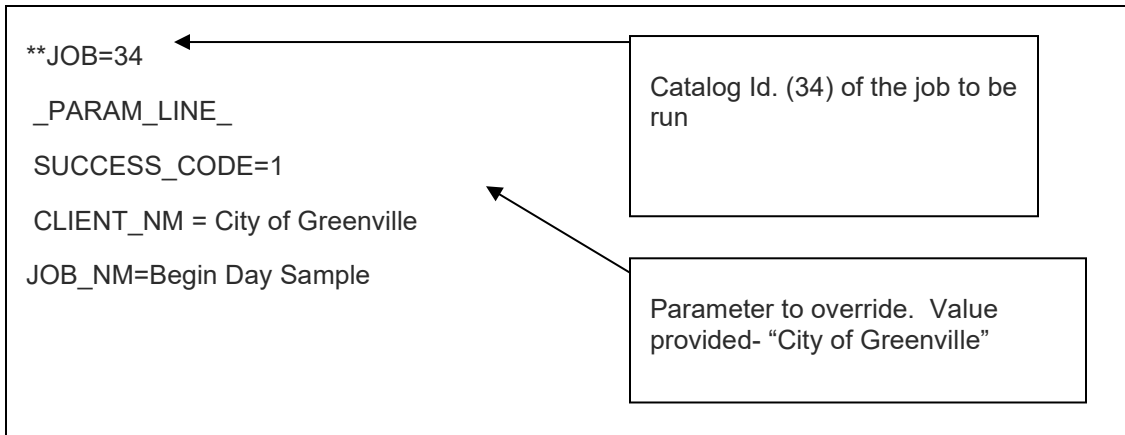
*Control Block (of type VARIABLE) that defines constants for a driver data file*

## Job Block

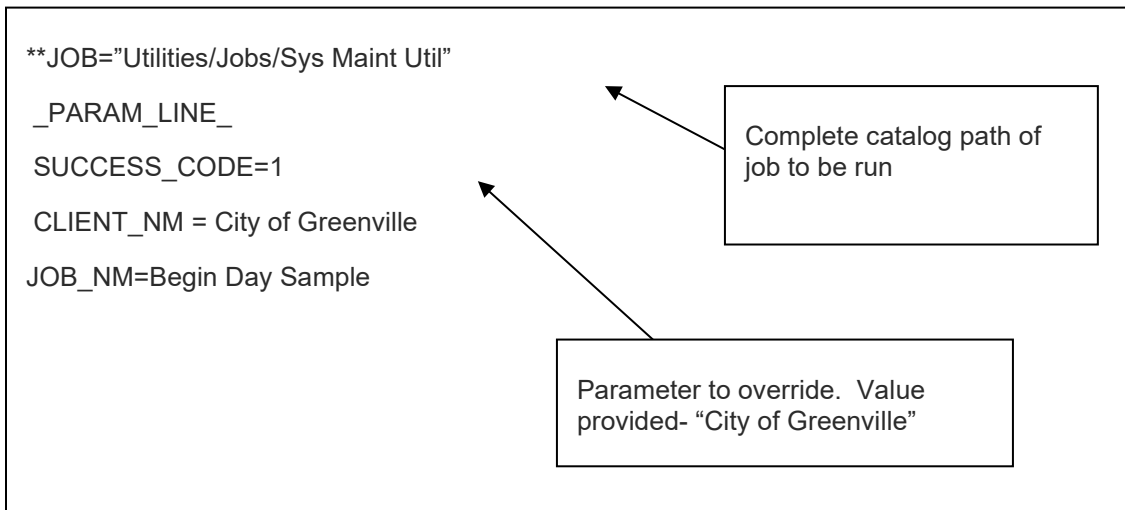
The Job block executes a CGI Advantage job already registered with the application job catalog. The catalog id of the job to be executed is specified after the “\*\*JOB” identifier. Alternately, the full catalog path name of the job may be specified for the “\*\*JOB” identifier. The catalog path name of a job consists of the full catalog names of all of the job’s parent folders, starting from the first parent under the root, and ending with the catalog name for the job. The separator character for folder names is ‘/’ and the entire path is enclosed within “”. The catalog name is case sensitive, so be careful when you enter in this information.

In addition to specifying the id or name of the job to execute, you can also override any of the parameters for the job in the job block as long as they can be overridden. If not overridden, the default parameter value from the job catalog is used. To override parameters, the parameter name must be specified exactly as defined in the catalog for the job.

Two sample job blocks are presented next to illustrate the two ways of specifying the job to be run and how parameter values must be supplied:



*Job Block with Job specified by a Catalog Id.*



*Job Block with Job specified by a Catalog Path.*

An additional Job Interaction Client specific parameter, **SUCCESS\_CODE**, can also be specified for the job block. It is the minimum acceptable return code from the current job that will allow the driver data file to continue being processed. An unacceptable return code value (actual job return code > SUCCESS\_CODE value specified) leads to a message being logged and the Job Interaction Client to stop processing the driver data file at the current block. By default, the SUCCESS\_CODE does not have to be defined. If not defined, the Job Interaction Client only continues if the job is executed successfully.

A complete list of job return codes is provided below:

**Successful = 1**

Return code 1 (Successful) is logged at the end of the batch job where no errors of type Warning, Validation or Severe were encountered.

**Warning = 4**

Return code 4 (Warning) is logged at the end of the batch job where errors of type Warning were encountered. These are usually business rule errors that allow business process to continue. Error log from the batch job should be brought to the attention of the party responsible for reviewing, examining, and resolving job output.

**Non Fatal Error = 8**

Return code 8 (Non Fatal Error) is logged at the end of the batch job where errors of type Validation were encountered. These are the errors that do not allow a current batch iteration to continue but are recoverable and allow continuance of the batch job. The batch job will continue when validation error is encountered but it will skip to the next iteration. In some cases this return code is also logged when no records to process are found by the batch job. Error log from the batch job should be brought to the attention of the party responsible for reviewing, examining, and resolving job output.

**Failed = 12**

Return code 12 (Failed) is logged at the end of the batch job where errors of type Severe were encountered. These are the errors that do not allow a business process or a batch job to continue. The batch job will stop upon encountering a severe error. Error log from the batch job should be brought to the attention of the party responsible for reviewing, examining, and resolving job output.

**Terminated = 16**

Return code 16 (Terminated) is logged when a batch job is terminated manually by the user.

**System Failure = 20**

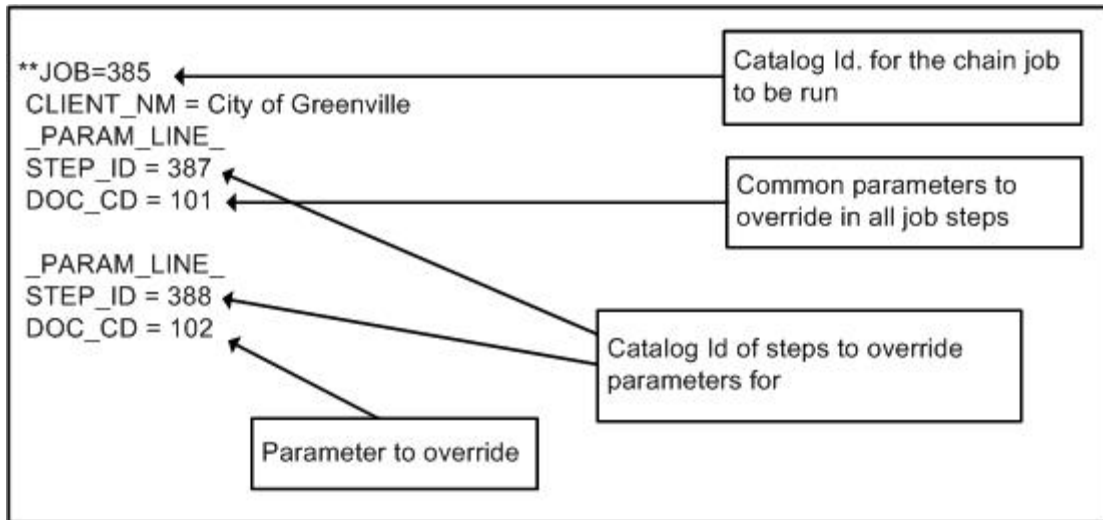
Return code 20 (System Failure) is logged when a batch job is terminated because of hardware failure that is, network issues, database server issues and so on.

Note: In the ADV 3.x runClient.bat|sh command, the Job Name could be specified with the '-I' parameter. This is not available in ADV 4 and a file parameter called 'JOB\_NM' can be added to any job to set the job name. Examples specifying a JOB\_NM are shown above.

## Chain Job Block

The Chain Job block executes a CGI Advantage chain job. A chain job can be specified in a manner similar to a job block in that either the chain job catalog id or the full catalog path name must be specified after the "\*\*\*JOB" identifier. Additional Job Interaction Client specific parameters are used to specify that individual chain job steps should be executed.

A parameter **STEP\_ID** is used to specify the chain job step (separate job) by providing the catalog id. Job steps must be separated by the "\_PARAM\_LINE\_" delimiter in the driver data file. Parameters for a job step can be specified after the **STEP\_ID** value has been specified. A sample chain job block is shown below:

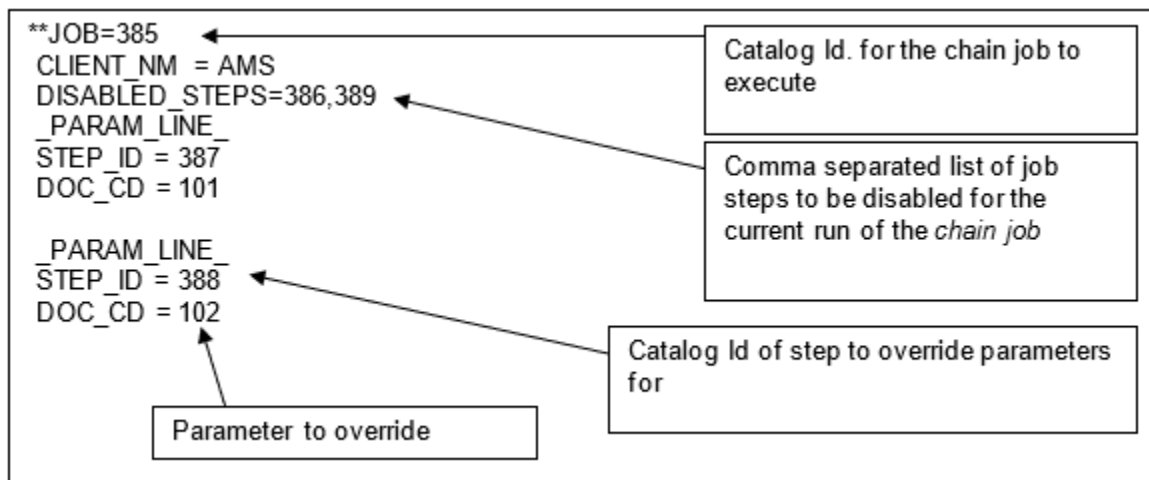


Sample Chain Job Block

In the sample chain job block shown above, a chain job gets scheduled in which the **CLIENT\_NM** parameter defined for every chain job element gets the value *City of Greenville*. The parameter **DOC\_CD** defined for job steps with catalog ids 387 and 388 get values 101 and 102, respectively. The default parameter values set up in the job catalog are used for the job steps not outlined in the chain job block and also for parameters not overridden for steps 387 and 388.

It must be noted that instead of the catalog id (such as 385), the full catalog path name can be used for identifying the chain job itself. However, the **STEP\_ID** parameter value can only accept valid catalog ids for the job steps.

It is sometimes required to disable certain job steps from the sequence defined for a chain job in the catalog. This is achieved by adding another optional parameter called **DISABLED\_STEPS** to the chain job parameters list that has a comma separated list of job step IDs to be disabled. A sample chain job block with disabled job steps is shown below:



Sample Chain Job Block with Disabled Job Step

The job steps with catalog ids 386, 389 are disabled for the current run. As shown in the two sample chain job blocks, common parameter values, such as **CLIENT\_NM** in the above sample, can be defined by listing them after the chain job block identifier (“\*\*JOB=“*Chain Job Catalog id or name*”) but before the “\_PARAM\_LINE” delimiter used for specifying job steps and their parameter values.

Similar to **SUCCESS\_CODE** functionality for Job Block it can also be defined for Chain Job Block in the common parameter area. An unacceptable return code value (actual job return code of last executed job in the chain > SUCCESS\_CODE value specified) leads to a message being logged and the Job Interaction Client to stop processing the driver data file at the current chain job block. Note the decision to continue with the next job step within the chain is controlled by the Pre Condition Return Code setup for the job during configuration of the chain job elements on the Batch Catalog (and not by SUCCESS\_CODE).

**Note:** A special case related to System Maintenance Utility has been addressed in Job Interaction Client where the dat file needs to have DOC\_TYP specified if DOC\_CD is either blank or has a wildcard (\*) for action code validate (161) or submit (162).

## Logging Configuration File

The Job Interaction Client uses the Apache Log4j logging framework to log informational and error messages to the console and log file. A description of Log4J is beyond the scope of this document. Details of the framework are available at <https://logging.apache.org/log4j/2.x/manual/configuration.html>.

## Suggested Usage

Job Interaction Client jobs are executed through the post\_cd script.

## DAT Files

1. Place custom dat files under the following directory outside the container.

```
<InstallLocation>/RTFiles/<Namespace>/<ADV_name>/Custom/JICScripts
```

2. All new dat files must be updated for owner 185:<group for deploy\_account> and have 775 permissions.

```
sudo chmod -R 775 ./*
```

```
sudo chown -R 185:<group for deploy account> ./*
```

## JIC Execution

The following executes the dat file through Job Interaction Client. The command can be added to a .sh script which can then be called by a batch scheduling tool or cron job. If upgrading from a previous Advantage release, this command replaces the ‘runClient’ call in the current script.

```
<InstallLocation>/Automation/Container_Fullbuilds/main/container_postcd.sh -n <Namespace> -s
```

```
<Statefulset> -j jic -f <DAT File Name>
```

Where

- <InstallLocation> = The location of the Container Automation directory. This is the same directory used for deploying containers.
- <Namespace> = The namespace for the environment. For example, prod.
- <Statefulset> = The application name. For example, admprod.
- <DAT File Name> = The file name for the DAT file.

**Example:**

```
/home/cgiadmin/Automation/Container_Fullbuilds/main/container_postcd.sh -n prod -s admjm -j jic -f BeginDay.dat
```

## Running JIC Jobs as a Specific User

The postcd scripts are updated to capture three optional parameters to provide users flexibility to configure/run jobs as different users. These new parameters are:

- Username
- Password
- INI File path

To run JIC jobs as different users, copy the .INI and .DAT files to a JICScripts folder and update the parameter files with the user, password, and jobmanager ID.

**From**

```
$ADV_HOME/RTFiles/$ADVAPPL/Configuration/DefaultFiles/AMSJobInteractionClient/scripts/sample  
s/
```

```
$ADV_HOME/RTFiles/$ADVAPPL/Configuration/DefaultFiles/AMSJobInteractionClient/ADV30JobMa  
nager.ini
```

**To**

```
$ADV_HOME/RTFiles/$ADVAPPL/Configuration/AppConfig/JICScripts
```

When running JIC, pass the custom ADV30JobManager.ini as a parameter in the below command:

```
./container_postcd.sh -n|--namespace <Namespace> -s|--statefulset <Application> -j|--jobs JOBNAME  
[-f|--file DAT_FILE] [-c|--config KUBECONFIGFILE] [-a|--argsforhealthcheck] [-v|--validation] [-h|--help]
```

**Examples:**

```
./container_postcd.sh -n|--namespace <Namespace> -s|--statefulset <Application> -j|--jobs JOBNAME  
[-f|--file DAT_FILE] [-c|--config KUBECONFIGFILE] [-a|--argsforhealthcheck] [-v|--validation] [-h|--help]
```

```
./container_postcd.sh -n test -s admtest -j jic -f AdvIndexer.dat -c /apps/Advantage4/test/kubeconfig -i  
test.ini -u test -p Advantage4
```

**To print help for the script:**



```
./container_postcd.sh -h
```

## Application Stop/Start

To stop, start, and restart the applications for the batch cycle, use the `container_bounce.sh` script with the options below.

```
cd <InstallLocation>/Automation/Container_Fullbuilds/main  
  
./container_bounce.sh -n <Namespace> -s <Statefulset> stop|start|restart [-r  
<NUMBER_OF_REPLICA>]
```

Where

- `<NUMBER_OF_REPLICAS>` = The number of pods you want to start/restart for the application, 1 is typical. Note for job manager applications, this should only be 1. For action start or restart, `NUMBER_OF_REPLICA` is required.

### Examples:

- To stop the application

```
cd /home/cgiadmin/Automation/Container_Fullbuilds/main  
./container_bounce.sh -n prod -s admprod stop
```
- To start the application

```
cd /home/cgiadmin/Automation/Container_Fullbuilds/main  
./container_bounce.sh -n prod -s admprod start -r 1
```
- To restart the application

```
cd /home/cgiadmin/Automation/Container_Fullbuilds/main  
./container_bounce.sh -n prod -s admprod restart -r 1
```