CGI Advantage® 4

Budget Control
Run Sheets Guide

Table of Contents

1 Purpose of the System Administration Guide

This manual is intended to help system administrators initiate, configure, monitor, and control all processing for CGI Advantage. The manual has five parts:

- The CGI Advantage System Administration Guide contains information about the CGI Advantage system architecture, and configuration (including the embedded third party components), post-installation setup, security configuration and considerations, workflow, job framework and its usage/maintenance, and other information pertinent to administering the application.
- The CGI Advantage HRM run sheet guides describe each process of CGI Advantage HRM in detail with its input, output, parameters, sort sequence, and selection criteria.
- The CGI Advantage Financial run sheet guides describe each process of CGI Advantage Financial in detail with its input, output, parameters, sort sequence, and selection criteria.
- The CGI Advantage HRM Payroll Engine System Administration Guide describes the system control tables and utilities for CGI Advantage HRM.
- The CGI Advantage VSS System Administration Guide describes each VSS process in detail with its input, output, parameters, sort sequence, and selection criteria.

System administration tasks include setting up and maintaining application security, querying and viewing the application status through logs and reports, managing workflow, setting up and maintaining system tables, and other critical application maintenance tasks.

1.1 Common terms and glossary used

The terms "Job" and "Batch" have been used interchangeably throughout the document. Please note that the CGI Advantage technical architecture is flexible enough to support the execution of jobs/batch processes while the application is available for online usage. In other words, the jobs/batch processes are technically not required to be "offline" processes.

2 Description of Processes

This chapter describes the processes in CGI Advantage that are considered system administration processes. For each process, you see information on these topics:

- Description
- Steps to Run this Process (if applicable)
- When to Run
- Major Input
- Output
- Parameters – Batch and Custom
- Sort Sequence
- Selection Criteria
- Notes
- Problem Resolution

System Wide Batch Parameters:

System wide batch parameter fields are available with each batch program, which provide the path for the input/output directory. These parameters allow sites to easily and quickly update the path for individual batch processes.

System wide batch parameters can be defined at the System Level, Area Level, Chain Job level, Chain Level or Job level. There has to be a default value set for the system wide batch parameters at any of these levels mentioned above so that the process will generate, read or write the respective files from the given location.

System wide batch parameters are defined at the System Level on the System Level Process Parameters (BATSETUP) reference page, searching for the Catalog Label of *Batch Catalog* and then choosing the record-level action of *Edit.*

- **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 (i.e. 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.

Pivot Date/Year Validation:

Note:

Assumption for date attributes: Set the Earliest Year (EARLIEST_YEAR) and Latest Year (LATEST_YEAR) on the Application Parameter reference page. When defining the year range, attention should be given to setting a range vast enough to accommodate all system impacts (such as imported transactions). The Job input date/year must lie between the above year range; otherwise, the process will fail.

2.1 Budgeting Batch and Chain Processes

Descriptions of the Budgeting processes are organized in this section in alphabetical order:

The following Batch jobs and Chain jobs affect the Budgeting area of CGI Advantage, but are located under a different section in the Batch Catalog. Please refer to the appropriate run sheet guide for more information on these jobs.

Batch or Chain Job Name	Run Sheet Guide Name
System Assurance 10	CGI Advantage Financial – System Assurance Run Sheets
System Assurance 11	CGI Advantage Financial – System Assurance Run Sheets
System Assurance 12	CGI Advantage Financial – System Assurance Run Sheets
Pre-Archive Budget Report	CGI Advantage Financial – Utilities Run Sheets
Budget Archiving	CGI Advantage Financial – Utilities Run Sheets

2.1.1 Budget Formula Recalculation

The Budget Formula Recalculation Job is used to:

1. Clean the system of Needs Initialization flags on the Budget Tracking Amount (BUDTAM) and Budget Formula Administration (BFADM) tables
2. Recalculate all calculated budget amounts on active budget and allotment structures when configuration table changes have occurred.

This job supports the *Restartability* feature, so that if the job fails or is terminated, the job can be restarted and will pick up with the last completed record. Restarting is tracked at the unique ID of a budget line, which is a system-generated number that uniquely identifies a budget line.

When to Run

When a formula for a calculated bucket has been changed or the pending increase/decrease flags have been changed, the system will flag the bucket/formula as Needing Initialization.

This job is just a part of a process to bring application data on the budget structure tables as well as the Budget Constraint Amounts table into sync with configuration tables. Running just the Budget Formula Recalculation job is rarely the only batch job that is required. Only when a calculated amount is changed that is not part of any active budget control, would this be the only job necessary. Otherwise, if a calculated amount is not incorrect, all controls that use the calculated amount directly or indirectly (because the amount is part of another calculated amount) have incorrect information stored on the Budget Constraint Amounts table.

STEPS TO BE FOLLOWED WHEN RUNNING THIS JOB:

1. The user should verify that the budget bucket formulas are correct. This is when the user should catch if a formula has been changed by mistake, correct the mistake, then continue with this process.
2. Bounce all VLS's so that the Needs Initialization flags are properly refreshed.
3. Validate a budget transaction to load the new setup into the Budget Flex Server.
4. Stop all other processes. Take the system offline. This job needs to be run when nothing else is running.
5. Run the Budget Formula Recalculation batch job.
6. After the successful completion of this job – all VLS's should be bounced again.
7. Validate a budget transaction to load the new setup into the Budget Flex Server.
8. Run the Calculate Budget Control Amounts batch job (see that run sheet for more information).
9. After the successful completion of that job – all VLS's should be bounced again.
10. Bring all the systems back online and validate a budget transaction before any other transaction processing. Now the system is ready for normal processing.

Description

The process involves the scheduling of a batch job, which can be accessed by clicking on **Budget: Batch Jobs : Budget Formula Recalculation** on the Batch Catalog.

1. Determines calculated amounts that have had configuration changes (that is, marked as needs initialization)
2. Determines all active budget structures
3. If there are no active budget structures or no calculated buckets are marked as Needs Initialization, a message is written to the job log stating such and the job ends as *Warning.*
4. Populates the temporary table, BFR_WKLD, with records for each budget structure and level found to contain the calculated budget amount(s). A sequence number is assigned to each record for latter processing to track the progress of the batch job.
5. Recalculates budget lines starting with the first budget level of an *expense only* budget structure, continuing through additional levels in that structure (including any allotment level). The next expense structure is then processed, which continues until all expense structures are done.
6. Recalculates budget lines starting with the first budget level of an *expense/revenue* budget structure, continuing through additional levels in that structure (including any allotment level). The next expense/revenue structure is then processed, which continues until all expense/revenue structures are done.
7. Recalculates budget lines starting with the first budget level of a *revenue only* budget Structure, continuing through additional levels in that structure (including any allotment level). The next revenue structure is then processed, which continues until all revenue structures are done.
8. As a structure is completed, messages stating such will be written to the job log for progress tracking.
9. If a save cannot be performed on a budget/allotment line, the unique ID of that line will be written to the job log stating a save was not successful.
10. Update to job log stating progress as each block defined by the LAST_REC_PROCESSED parameter is completed.
11. Sets Needs Initialization flags to unchecked R_GN_BKT and R_GN_FORM.
12. Clears the temporary table, BFR_WKLD, which is used to track progress through budget lines for Restartability.

Parameters:

Job No	Job	Parameter	Description	Default Values	For System Use (non-overridable by user)
1	Budget Formula Recalculation	COMMIT_BLK_SIZE	Number of updates to occur before a save is committed to the database. Must be a positive integer.	250	No
1		SELECT_BLK_SIZE	Number of records to be processed at a time. Must be a positive integer.	350	No
1		LAST_REC_PROCESSED	Number of budget lines to be processed before	5000	No

			writing log entry. Must be a positive integer.		

Please Note - Every time a parameter fails validation the system logs the error and sets the **Job Return Code** to *Failed* and stops the job.

Major Input

Budget Formula Recalculation (BFR_WKLD)

Peripheral Data Objects

- Budget Tracking Amounts (R_GN_BKT)
- Budget Formula Administration (R_GN_FORM)
- Applicable Budget Structure/Level Objects (BUD_STRU_*_LVL_*)
- Applicable Allotment Structure/Level Objects (A LOT_STRU_*_LVL_*)
- Budget Link table (GN_LNK)

Output

- Log of successful completion.
- If no records meet the selection criteria then the job will stop with a return code of Warning.
- Budget Tracking Amounts (R_GN_BKT) and Budget Formula Administration (R_GN_FORM) will be cleaned of their dirty flags.
- All active budget structures using buckets marked as needing initialization will be recalculated and saved.
- Allotment structures using buckets marked as needing initialization will be recalculated and saved.
- Budget Links will be recalculated and save if a linked bucket has been recalculated.

Selection Criteria

Selection for writing to workload table

- Selection criteria for obtaining budget structures that need initialization is the following:
  - Structure must be active (Active structures with inactive levels will still be processed)
  - Structure must contain a bucket that needs initialization

Selection from workload table

1. Order BFR_WKLD by SEQ_NO
2. Select the first SEQ_NO from the last stored checkpoint

Problem Resolution

- If the job fails, view the log for information about the cause of the failure. Resolve the problem and restart the job.

- Error messages and possible causes:

Error Message	Possible Causes	Resolution
Workload Table is not empty. There may have been a previous run not yet completed or another process may be running. Please contact your system administrator.	A previous run of this job failed and has not been resolved.	Solve the issue which caused the failure of the previous job, and restart the previous job.
		If a new run of the process is desired, the Workload table must be emptied by the user, then run the new job.
	There is erroneous data in the workload table.	The user must empty the workload table and then restart the job.
Another instance of the job is running	Another instance of the job is running.	Only one instance of this job may be run at a time. Nothing else should be processed while this job is running.
Error retrieving active budget structures. Please see administrator for more info.	No structures are marked active on GN_BUD_STRU	Mark those structures active that should be and restart all VLS's before running this job.
Could not populate workload table.	A problem occurred while attempting to prepare the work for this job.	Check the error logs for details and take appropriate actions to resolve the problem. Restart all VLS's and then restart the job.
Active Structures and Buckets needing initialization found, but no active structure contains the bucket needing initialization. Please see an administrator for more information.	The buckets marked as needing initialization aren't used on any active structures.	In this case, the needs initialization flags are reset even though there were no structures to recalculate. If any inactive structures that use these buckets are activated later, they will not have been recalculated.
Error initializing process variables.	Errors occurred while setting up necessary values to run this job.	Check the error logs for details and take appropriate actions to resolve the problem. Restart all VLS's and then restart the job.
Error committing set of budget structures to recalculate.	There was a problem saving information to the workload table.	Check the error logs for details and take appropriate actions to resolve the problem. Restart all VLS's and then restart the job.
The job encountered an error. Please see an administrator for	An unexpected error was encountered during the	Check the error logs for details and take appropriate actions to resolve the

more information.	process.	problem. Restart all VLS's and then restart the job.
Could not perform save on allotment line with UNID Could not perform save on budget line with UNID	An error occurred while recalculating a specific budget or allotment line.	See the error log for more details. Once the issue is resolved, restart the job. An option would be to process a budget transaction online to affect a recalculation or receive more information about why an update is not possible.

2.1.2 Budget Roll Process (Modes 1, 2, 3 and 5) – Update Mode

The Budget Roll process, run in update mode, automatically creates budget lines in a target year based on budget lines in a source year. Budget lines in any type of budget structure can use the Budget Roll Process, except those that do not have a Budget Fiscal Year as the key to its budget lines. Additionally, budget lines that have the multiple year value (9999) in the Budget Fiscal Year field will not be selected by the process, as they do not need to be rolled.

The Budget Roll process also contains the ability to roll certain budget features including allotments, links, and line level budget controls.

Most of the parameters of the Budget Roll Process come from the Parameters for Budget Roll Process (PBRP) page. The first job step in the chain contains a batch parameter where the ID of a PBRP record is specified to pass the parameters to the job. Other than performance parameters for record processing, the other batch parameter to control rolling functionality is the specification of a COA Crosswalk ID. The cross-walking functionality allows for Chart of Account codes to be changed in the target year through the Chart of Account Crosswalk (COAX).

When to Run

Modes 1 and 2 are to be run before the start of a new the Budget Fiscal Year after the New Year Table Initialization job has populated reference tables for the new Budget and Accounting Fiscal Years.

Mode 3 and 5 are to be run after a new budget year has started against the prior budget year. They can be run once or multiple times, as subsequent financial activity and bring availability back to budget lines that had been rolled or lapsed.

Other recommendations for running:

- System activity by users should be prohibited when running a budget roll.
- Run the Budget Roll Report job with your selection parameters to get an accurate picture of volume and to ensure any selection criteria used achieve the desired results.
- Ensure that the Automatic Transaction Numbering (ADNT) setup is in place to create the transactions.
- If the volume for selection is significant, use a test environment to tweak the number of processors and block size parameters for optimal performance. For example, using the *All Budget Lines for an Appropriation to 1 Transaction* choice with a budget structure that contains Appropriation as a key field to the 1st budget level and an appropriation can only be used once at that level within a budget fiscal year (sample edit put in place to mimic standard budget functionality of prior applications), would result in budget transactions with a minimal number of level 1 lines. The choice of *Budget Line to 1 Transaction* would not be a good choice in this scenario because the same level 1 line would exist on multiple transactions. Conversely, choosing the *All Budget Lines for a Department to 1 Transaction* may produce budget transactions that contain more lines than acceptable for processing.
- When running budget roll in any mode, be sure that setup on tables such as Transaction Control (DCTRL) and Budget Fiscal Year Staging allow for the budget roll activity to occur without errors
- Approvals will be bypassed and overrides will be automatically applied as part of the load of budget transactions unless you make custom adjustments to the BRLoad.txt file to not perform these actions. They are in place to ensure that the budget transactions do not enter workflow like manually created budget transactions and do not reject for overridable errors. Because the Bypass Approval action is controlled for a transaction code on the Transaction Control (DCTRL) page with the Approval Override Allowed flag, the budget transaction code

used in a roll must have the flag checked before running the roll. If it is not normally checked, it should be unchecked when done.

- The parameters for Transaction Record Date, Source Fiscal Year, Target Fiscal Year, Source Accounting Period, and Target Accounting Period are very useful when the defaults from the Application Date are not desired when a budget roll is performed. However, values from these fields may violate normal production constraints from Transaction Control setup for future and past dating. Be sure you will not be violating any of those rules. If so, then make changes to Transaction Control that will be reversed when the roll is complete.
- When there are chart of account changes between budget years, the Budget Roll chain can use the Chart of Account Crosswalk (COAX) page to make those changes automatic.

Description

Modes 1 and 2: Selected budget lines in a current year (source year) are copied to budget transactions with the budget fiscal year incremented to the new year (target year). If applicable, the COA Crosswalk table will be used to translate COA values in the source year to their target year's values before creating the budget transactions. The difference between the two modes is in what dollar amount is on the budget lines in the target year:

Mode 1 -- Create Zero-Dollar Budget Lines in the new year based on prior year lines. This is common with revenue budget lines where nothing is estimated but the lines provide valid COA combinations.

Mode 2 – Create Non-Zero Budget Lines in the new year, using an amount specified in the prior year. This is also common with revenue budget lines where what was budgeted in the prior year should be budgeted in the following year again. Any differences will be accomplished manually or with an upload.

Mode 3 – Roll Unused Budget takes what is unspent in the prior year, reverts it out of that year, and carries it forward to the current year. This is common for grant budgets and instances where BFY 9999 is not desired.

Mode 4 – This mode does not apply to Budget Roll but is used with the Open Activity & Budget Roll processes. However, it is much like mode 3 when used with the but the amount used is not a single amount on a budget line but the sum of accounting activity being rolled.

Mode 5 – Lapse Unused Budget is used to bring the availability of budget lines in a prior year to $0.00 to reflect nothing remains for spending. This feature is not intended to replace using BFY Staging but used in conjunction with that feature.

The chain process consists of the following batch jobs:

1. Job 1 – Budget Roll
2. Job 2 – Load Transactions
3. Job 3 – Rollup and Submit Transactions
4. Job 4 – Roll Reports

Job No	Job	Description	Output
1	Budget Roll	Generates budget transactions in Target BFY for the source BFY lines	• Transaction XML file (EXP_FILE_NM) containing created budget transactions • "Budget Lines Selected for Roll"

			report • "Budget Lines not Rolled" report. • "Budget links for Revenue Budgets Not Rolled" report due to Revenue Budgets not already existing for the Target Fiscal Year. • Parameter file (LOAD_FILE_NM) to be used by Load Transactions job to load budget transactions. • Parameter file (ROLLUP_SUBMIT_FILE_NM) to be used by Rollup and Submit Transactions job to perform rollup and submit budget transactions. • Common Parameter file (CHAIN_PARM_FILE) to be used by Roll Reports job to access common parameters. All above files are meant for system use.
2	Load Transactions	Loads budget transactions	Budget transactions are created
3	Rollup and Submit Transactions	Rollup and Submits budget transactions	• Rollup is performed on Budget transactions and they are submitted
4	Roll Reports	Generates reports to indicate successful and unsuccessful budget transactions	• "Budget Roll Crosswalk of Successful Transactions" report • "Budget Roll Listing of Unsuccessful Transactions" report.

Parameters:

For the entire chain process a user has to enter parameters for the ‘Budget Roll’ job only. The process propagates required parameters down the chain jobs. Some of them are non-overridable parameters, which are provided with the Job Setup and do not have to be specified by the user. These are meant for system use.

Job No	Job	Parameter	Description	Default Values	For System Use (non-overridable by user)
1	Budget Roll	BUD_ROLL_PARM_ID	If you are running the job in Financial application, you can click on the **Custom Parameter** link and select a parameter record. If you are running the job in the Administration		No

			application, enter a valid Parameter ID from the Custom Parameter table from the Financial application.		
		CHAIN_PARM_FILE	Common Chain Parameters File (.txt)	BRParams.txt	Yes
		CHK_PT_SIZE	Checkpoint Block Size (Number of transactions processed after which checkpoint is to be updated)	100	No
		CLIENT_NM	Client name for Report		No
		COMMIT_SIZE	Commit Block Size for transactions	1	No
		DOC_STA_CD	Transaction Status for loaded Budget Transactions (2-Ready or 1-Held)	2	No
		EXP_FILE_NM	Transaction XML file for Budget Transactions (.xml)	BRDocs.xml	Yes
		LOAD_FILE_NM	Load Parameter File for Budget Transactions (.txt)	BRLoad.txt	Yes
		ROLLUP_SUBMIT_FILE_NM	Report Parameter File for Budget Transactions (.txt)	BRRollupSubmit.txt	Yes
		XWALK_PROC_ID	Process Id to pick COA Crosswalk records		No
		AMSEXPORT (** **Refer to Note**: Assumptions for SWBP on page no. 5)	Export Location at Budget Roll Job	-	Yes
		AMSEXPORT (** **Refer to Note**: Assumptions for SWBP on page no. 5)	Parameter Location at Budget Roll Job	-	Yes
2	Load Transactions	PARM_FILE	Parameter File (.txt). This parameter value should be same as that of the LOAD_FILE_NM parameter of the 'Budget Roll' job	$$AMSPARM$$/BRLoad.txt	Yes

3	Rollup and Submit Transactions	PARM_FILE	Parameter File (.txt). This parameter value should be same as that of the ROLLUP_SUBMIT_FILE_NM parameter of the 'Budget Roll' job	$$AMSPARM$$/BRRollupSubmit.txt	Yes
4	Roll Reports	PARM_FILE	Report Parameter File for Budget Transactions (.txt). This parameter value should be the same as that of the ROLLUP_SUBMIT_FILE_NM parameter of the 'Budget Roll' job	BRRollupSubmit.txt	Yes
		CHAIN_PARM_FILE	Common Chain Parameters File (.txt). This parameter value should be the same as that of the CHAIN_PARM_FILE parameter of 'Budget Roll' job	BRParams.txt	Yes
		AMSPARM (** **Refer to Note**: Assumptions for SWBP on page no. 5)	Parameter Location at Roll Reports Job	-	Yes

Please Note - Every time a parameter fails validation the system logs the error and sets the **Job Return Code** to *Failed*, which in turn deactivates further jobs down in the chain, and prevents them from executing.

In the case where no records were found to be rolled in the Budget Roll job then the job logs a message and sets the **Job Return Code** to *Non fatal error* only to discontinue further jobs down the chain. Roll Reports jobs end with non-fatal **Job Return Code** in the case of transaction submission errors during the Rollup and Submit Transactions job.

Major Input

Budget level tables	BUD_STRU_*_LVL_*
Budget line constraints	GN_LN_CNST
Budget links	GN_LNK

Peripheral Data Objects

Parameter for Budget Roll Process	BUD_ROLL_PARM
Budget Structure	GN_BUD_STRU
Budget Level	GN_BUD_LVL

Budget Allotment	GN_ALOT_OPT
Auto Numbering	AUTO_DOC_NO
COA Crosswalk	R_COA_CROSSWALK

Output

- Mode 1 and 2 generates budget transactions with lines in the Target Budget Fiscal Year.
- Mode 3 generates budget transactions with lines for both the Source and Target Budget Fiscal Years.
- Mode 5 generates budget transactions with lines for the Source Budget Fiscal Year.
- Generates the following reports:
  - Budget lines selected for Roll
  - Budget lines not Rolled
  - Budget links for Revenue budgets not Rolled
  - Budget Roll Crosswalk of Successful Transactions
  - Budget Roll Listing of Unsuccessful Transactions

Sort Criteria

COA element	COA element as specified in Transaction break in BUD_ROLL_PARM
Unique Identifier	UNID

Selection Criteria

- Select criteria for obtaining budget lines from Budget Inquiry Table BUD_STRU_x_LVL_y (x – Structure id from Parameter ID record, y – lowest required level for Structure) is as follows using the Parameter ID from Parameters for Budget Roll Process (PBRP):
  - Budget Fiscal Year = Source BFY
  - Fund = Selected Funds
  - Department = Selected Departments
  - Appropriation = Selected Appropriations
  - Appropriation Type = Selected Appropriations with Appropriation Type
  - Appropriation Category = Selected Appropriations with Appropriation Category. If you do not see this field on PBRP, use Configure Page to make it visible.
  - If Bypass Deactivated Lines is *true* for the parameter, then if the Active indication is *false* for a budget line, then that budget line is not selected.
  - If Bypass Unused Lines is *true* for the parameter, then if all of the expense and revenue amounts updated by accounting transactions are zero, the line will be skipped.

  Order by

  - The Transaction Break for the Parameter Id record determines if budget transactions will be created grouping all lines for a Fund, Appropriation, or Department. The other choice is one budget line per budget transaction. Sorting will be done prior to transaction creation based on this break value.
  - Unid – Unique Identifier on the budget line.

Problem Resolution

- If the chain process was discontinued for any reasons then each of the jobs in the chain has the ability to be restarted from the point it left off. The Restart ability of the jobs can be used

anytime except in the case of program errors or exceptions. Restarting has a condition that no ‘Budget Roll’ chain process should be scheduled between the time frame the job was discontinued and is being restarted and the parameters provided to the chain process should not be changed.

- ‘Roll reports’ job ends up in non-fatal error if the preceding ‘Rollup and Submit transactions’ job had transaction submission errors, thus deactivating other jobs down the chain. If these errors were due to some data setup, fix the data setup errors and reschedule the chain enabling the preceding transaction submit job and Roll Reports job and disabling the prior jobs that ran successfully. This reschedule has a condition that no ‘Budget Roll’ chain process should be scheduled between the time frame the job was discontinued and is being scheduled again and the parameters provided to the chain process should not be changed.
- This batch program produces new Advantage transactions, which are subject to the line limit functionality constraints. Sites should ensure that they run this job with parameters set to ensure that the created transactions are within the line limit controls.

2.1.3 Budget Rollup Update Job

Chain or Job Name	Budget Rollup Update Job
Recommended Frequency	On Demand
Single Instance Required	Yes
Can be restarted?	No
Reports generated	No

Overview

The Budget Rollup Update job facilitates in updating the roll-up values for COA elements on Budget Inquiry pages. The job is useful for budget structures with roll-up as informational fields on budget inquiry pages and eliminates the need to create new budget lines with updated roll-up information. At present, budget structures 30 and 41 are the only structures that have roll-up as informational fields. On these structures, the rollups are present for Object, Revenue and Unit. The update job can hence be run only for these elements.

The process uses FY to identify the latest roll-up values for the selected COA. If a roll-up is blanked out on the COA page, the process will remove the roll-up value from budget inquiry pages if they were populated previously.

Process Steps	Messages
Parameter Validation	• Parameters are valid or invalid depending on the Validation. If the parameter is invalid, the invalid value is displayed in the log with the following message: "Problem with one or more of the Batch Job Parameters" • Batch Parameter validation completed
Identification of roll-up	• After validating parameters, the system uses the FY and COA value (individually) to perform a look-up to the COA page to identify the roll-up values.
Updates to Budget Inquiry Pages	• The process then updates the roll-up values on inquiry records using queries and logs the following messages: "About to Update <Budget Structure and Level> for Column <COA> where COA Table <COA> does have a record" "Updated <Budget Structure and Level> for Column <COA> number of records <n>."

Major Input

- Object (OBJ)
- Revenue (RSRC)
- Unit (UNIT)

Minor Input

- Fiscal Year (FY)

Batch Parameters

Parameter	Description	Default Value
Budget Fiscal Year (BFY)	Required Budget Fiscal Year for selecting Budget Lines for roll-up updates.	<Blank>
COA Element (COA)	Required COA data object for which roll-ups need to be updated. Valid values are R_UNIT, R_OBJ and R_RSRC.	<Blank>
Fiscal Year (FY)	Required Fiscal Year for identifying the Roll-up values.	<Blank>
Budget Structure (STRU_ID)	Required Structure ID for updating the roll-ups. Valid values are 30 or 41.	<Blank>

Major Output

- Budget Structure 30 Inquiry Pages (BQ30LV*)
- Budget Structure 41 Inquiry Pages (BQ41LV*)

Job Return Code

The following table shows the potential job Return Codes for the Budget Rollup Update job:

Return Code	Condition
Successful (1)	Budget Inquiry records updated.
Warning (4)	0 Budget Inquiry records updated.
Non-Fatal Error (8)	N/A
Failed (12)	The job may fail under the following conditions: • Parameters validation failed. • BFY is required • FY is required • Structure ID is required • COA element is required • Structure ID has to be 41 or 30 • FY should be valid on FY • Run time exceptions for unexpected situations.

Terminated (16)	This return code is issued when the job is terminated by the user.
System Failure (20)	This return code is issued when the job is terminated because of database server or network issues.

Sort Criteria

N/A

Selection Criteria

- The records are selected based on BFY and COA element.

Problem Resolution

The process cannot be restarted on failure; however, a new job can be scheduled by resolving the issue.

The following table shows the possible return codes and recommendations for each processing step:

Step 1: Parameter Validation

Possible Return Codes	Condition	Recommendation	Other Instructions
Successful (1)	Successful	N/A	N/A
Warning (4)	When no budget inquiry records are updated	Verify the job parameters.	Reschedule the job with appropriate parameter values.
Non-Fatal Error (8)	N/A	This step does not issue this return code.	This step does not issue this return code.
Failed (12)	Structure ID other than 30 or 41. **Sample Message:** "STRU_ID only valid values are 30 or 41"	Enter a valid Structure ID.	Reschedule job with appropriate parameter values.

Possible Return Codes	Condition	Recommendation	Other Instructions
	Negative integer on FY or BFY. **Sample Message:** "Fiscal Year <n> is invalid, must be a positive integer" "BFY <n> is invalid, must be a positive integer"	Enter a valid FY and BFY.	Reschedule the job with appropriate parameter values.
	Invalid FY. **Sample Message:** "FISCAL_YEAR supplied does not appear to exist inside the FY Table"	Enter a valid FY.	Reschedule the job with appropriate parameter values.
	Invalid COA table. **Sample Message:** "COA Table Name supplied <n> cannot be used for Structure <n>"	Please specify a COA table that has roll-ups and is valid for the structure and	Reschedule the job with appropriate parameter values.
	The job failed because of runtime exceptions for an unexpected situation.	The reason for the failure needs to be investigated before re-scheduling the job.	Reschedule the job with appropriate parameter values.
Terminated (16)	The job is terminated manually by the user.	The reason for the termination needs to be investigated. The job can be rescheduled.	Schedule a new job.
System Failure (20)	The job is terminated because of database server or network issues.	The reason for the System Failure needs to be investigated. The job can be rescheduled.	Schedule a new job.

2.1.4 Budget Structure 90 Activity Load

Chain or Job Name	Budget Structure 90 Activity Load
Recommended Frequency	On Demand
Single Instance Required	Yes
Can be restarted?	No
Reports generated	No

Overview

The Budget Structure 90 Activity Load process loads the budgets and actuals to the Fiscal Year (FY) and Accounting Period (APD) Budget Structure 90 drilldown pages. The process loads all records or records for the BFYs specified in the parameter when run in initial load mode. Subsequently, when run in incremental mode, the process loads records incrementally. There are options to rebuild selected records by using the rebuild mode and specifying the unique IDs (UNID) values from the Budget Structure 90 Level 2 BQ90LV2 page (or rather then BUD_STRU_90_LVL_2 table) that need rebuilding. The process only considers the updates to level 2.

Process Steps	Messages
Parameter Validation	The process begins by validating the parameters. Depending on the parameter and validation, the following messages are logged: • Parameter <name> not found. • Parameter <name> value '<value>' is invalid, must be a positive integer. • Incremental Run must be Yes or No • Initial BFY(s) or Rebuild UNID(s) should not be supplied for incremental runs
Pre-selection processing	After the parameter validation the process checks if the run is for rebuilding existing records. If yes, the process starts deleting the UNID records specified in the parameter and the following messages are logged: • Starting Deletion of UNID Records for Rebuild • Deleted *n* records from Level 2 activity table • Deleted *n* records from Level 4 activity table • Finished Deletion of UNID Records for Rebuild If no, the process resets to $0 the pending amounts on all records that have pending amounts and the following messages are logged: • Starting reset of Pending Amounts • Processed *n* records for Pending Amount Reset for Level 2 activity table • Processed *n* records for Pending Amount Reset for Level 4 activity table • Finished reset of Pending Amounts

Process Steps	Messages
Selection and Processing of Records	The process then starts to select Pending Budget transactions and logs the following message: • Starting Pending Budget Transactions • Executing Query for Pending Budget Transactions Once completed the process logs the following messages: • Processed *n* records • Finished Pending Budget Transactions The process then starts processing the pending accounting transactions and logs the following messages: • Starting Pending Accounting Transactions • Executing Query for Pending Accounting Transactions Once completed the process logs the following messages: • Processed *n* records • Finished Pending Accounting Transactions The process then starts processing the final budget transactions and logs the following messages: • Starting Final Budget Transactions • Executing Query for Final Budget Transactions Once completed the process logs the following messages: • Processed *n* records • Finished Final Budget Transactions The process then starts processing final accounting transactions and logs the following messages: • Starting Final Accounting Transactions • Executing Query for Final Accounting Transactions Once completed the process logs the following messages • Processed *n* records • Finished Final Accounting Transactions When no record is found, then the process logs the following message: • No records found to process for current query
Selection and Processing of Records (Rebuild UNID)	The process, when run with rebuild mode set to true, starts processing selected UNID records. The process follows the same flow and logs the same messages as in the previous section for pending budget and accounting transactions and final budget and accounting transactions.

Major Input

- Pending Budget Activity: BG_DOC_LN (Budget Transaction Lines)
- Pending Accounting Activity: PSTNG_LN_CAT (Posting Line Catalog)
- Accepted Accounting Activity: JRNL_ACTG (Accounting Journal)
- Accepted Budget Activity: JRNL_BUD (Budget Journal)

Minor Input

Application Control (APPCTRL) parameters:

- Budget Structure 90 Inquiry Activity Initialized BFYs (BUDINQ_ACTVNLYS_INITIAL_BFYS)

  The comma separate BFY list provides a list of BFYs that have been run in initialize mode. These BFYs are used in incremental runs.

- Budget Structure 90 Inquiry Activity Last Run Date (BUDINQ_ACTVNLYS_LASTRUN)

  The date and time when the process was last run in incremental mode. This date is used for both initial and incremental runs. Any manual updates will cause issues with job processing, and may require manual intervention before running the job to fix the resulting issues.

Batch Parameters

Parameter	Description	Default Value
Commit Block Size (COMMIT_SIZE)	A required performance parameter that specifies the number of records after which the job should perform a commit.	1000
Select Block Size (SELECT_BLOCK)	A required performance parameter used in the selection of data for updates.	1000
Progression Message Block Size (PROG_CTR_SZ)	A required performance parameter that specifies the number of records after which the job should log progression messages. It should be a positive integer.	5000
Initial BFYs (INITIAL_BFYS)	An optional selection parameter used only when initially loading past data. Multiple values allowed if separated by commas. If a regular run then leave blank.	No Default
Rebuild Unique IDs (REBUILD_UNIDS)	An optional selection parameter used when rebuilding information for one or more budget lines. Multiple values allowed if separated by commas.	No Default
Is Incremental Run? (INCREMENTAL_RUN)	A required indication of how the job loads data. Yes, indicates the process is running from the point it left off last. No indicates it is running to rebuild data or the initial load. If left blank, Yes defaults.	Yes

Major Output

- ACT_BUD_STRU_90_LVL_2
- ACT_BUD_STRU_90_LVL_4

Minor Output

Application Parameter (APPCTRL) parameters:

- Budget Structure 90 Inquiry Activity Initialized BFYs (BUDINQ_ACTVNLYS_INITIAL_BFYS)

  This BFYs are appended every time the process is run in initialize BFY mode with BFYs not currently in the parameter. There is no need to manually update this parameter.

- Budget Structure 90 Inquiry Activity Last Run Date (BUDINQ_ACTVNLYS_LASTRUN)

  This date and time gets updated when the process is run in incremental mode. It is not updated when a new BFY is initialized as the initialization takes records only to the current date present in this parameter. Any manual updates will cause issues with job processing, and may require manual intervention before running the job to fix the resulting issues.

Note: If there is a gap between when the process was last run in incremental mode and when the process is run to initialize a BFY and there are transactions that were created with that BFY after the last incremental run there is a possibility that the process may not consider these transactions. In such cases it is recommended to also run the process in incremental mode immediately to consider these transactions.

Job Return Code

The following table shows the potential job Return Codes for the Budget Structure 90 Activity Load process:

Return Code	Condition
Successful (1)	All the selected records are processed successfully.
Warning (4)	No eligible records found. This could be because of the following reasons: • No transaction processing since the last incremental run • No transactions processed for a particular BFY • No records found to process for current query
Non-Fatal Error (8)	N/A
Failed (12)	The job may fail under the following conditions: • Parameters validation failed. • Failed to initialize additional member variables • Run time exceptions for unexpected situations.
Terminated (16)	This return code is issued when the job is terminated by the user.
System Failure (20)	This return code is issued when the job is terminated because of database server or network issues.

Sort Criteria

The records are grouped by BFY, Department, Fund, Appropriation Group, Appropriation Unit, Fiscal Year, Period, Transaction Code, Transaction Department and Transaction ID.

Selection Criteria

When selecting pending budget transactions from budget lines (BG_DOC_LN) or final budget transactions from budget journal (JBUD) the records are selected as follows

- Fiscal Year, Period, BFY, Department, Fund, Appropriation Group, Appropriation Unit, Transaction Code, Transaction Department, Transaction ID, Posting Code, Increase/Decrease indicator, Expense Budget Bucket ID, Revenue Budget Bucket ID

When selecting pending accounting transactions from posting line catalog (PSTNG_ LN_CAT) or final transactions from accounting journal (JACTG) the records are selected as follows

- Fiscal Year, Period, BFY, Department, Fund, Appropriation Group, Appropriation Unit, Transaction Code, Transaction Department, Transaction ID, Transaction Function, Posting Code, Offset Posting Code, Increase/Decrease indicator, Expense Budget Bucket ID, Revenue Budget Bucket ID

Problem Resolution

The process cannot be restarted on failure; however new job can be scheduled by fixing the issue.

The following table shows the possible return codes and recommendations for each processing steps:

Step 1: Parameter Validation

Possible Return Codes	Condition	Recommendation	Other Instructions
Successful (1)	Successful		
Warning (4)	N/A	This step does not issue this return code.	
Non-Fatal Error (8)	N/A	This step does not issue this return code.	
Failed (12)	Required Parameters are not valid. **Sample Messages:** Cannot convert BFY to a number. BFY is equal to the value in BUDINQ_ACTVNLYS_INITIAL_BFYS on APPCTRL	Enter correct Budget Fiscal Year parameter value.	Reschedule job with appropriate parameter values.

Possible Return Codes	Condition	Recommendation	Other Instructions
	Invalid Parameters: **Sample Messages:** Initial BFY(s) or Rebuild UNID(s) should not be supplied for incremental run	In case of Rebuild please do not specify any value in initial BFY parameter. In case of initial run please do not specify any value in Rebuild UNID.	Reschedule job with appropriate parameter values.
	The job failed because of runtime exceptions for an unexpected situation.	The reason for the failure needs to be investigated before re-scheduling the job.	
Terminated (16)	The job is terminated manually by the user.	The reason for the termination needs to be investigated. The job can be rescheduled.	
System Failure (20)	The job is terminated because of database server or network issues.	The reason for the System Failure needs to be investigated. The job can be rescheduled.	

Step 2: Record Selection

Possible Return Codes	Condition	Recommendation	Other Instructions
Successful (1)	Successful		
Warning (4)	If no transactions are found for the parameters entered: **Sample Message:** “Records processed: 0”	Verify the parameters entered and reschedule the job if records had processed since the last run.	
Non-Fatal Error (8)	N/A	This step does not issue this return code.	
Failed (12)	The job failed because of runtime exceptions for an unexpected situation.	The reason for the failure needs to be investigated before rescheduling the job.	
Terminated (16)	The job is terminated manually by the user.	The reason for the termination needs to be investigated. The job can be rescheduled.	

System Failure (20)	The job is terminated because of database server or network issues.	The reason for the system failure needs to be investigated. The job can be rescheduled.	

Step 3: Record Processing

Possible Return Codes	Condition	Recommendation	Other Instructions
Successful (1)	All records are processed successfully.		
Warning (4)	N/A	This step does not issue this return code.	
Non-Fatal Error (8)	N/A	This step does not issue this return code.	
Failed (12)	If any exception occurs while processing records. **Sample Message:** "Problem setting other critical information"	Ensure parameter data is accurate.	Reschedule job after corrections.
	The job failed because of runtime exceptions for an unexpected situation.	The reason for the failure needs to be investigated before rescheduling the job.	
Terminated (16)	The job is terminated manually by the user.	The reason for the termination needs to be investigated. The job can be rescheduled.	
System Failure (20)	The job is terminated because of database server or network issues.	Reason for the system failure needs to be investigated. The job can either be restarted or a new job can be scheduled.	

2.1.5 Budget Structure 90 Activity Systems Assurance Job

Chain or Job Name	Budget Structure 90 Activity Systems Assurance Job
Recommended Frequency	On Demand
Single Instance Required	No
Can be restarted?	No
Reports generated	Yes

Overview

The Budget Structure 90 Activity System Assurance report lists the level 2 budget lines where the stand alone (not calculated) amounts do not match between the Budget Structure 90 Level 2 (BQ90LV2) and the Detail Activity Inquiry. It is recommended to run the Budget Structure 90 Activity Load job in the incremental or rebuild mode prior to running the Budget Structure 90 Activity System Assurance job. Any out of sync condition reported by this job should then be verified with the System Assurance 1 or System Assurance 11 jobs to determine if the issue was with the Budget Structure 90 Activity Load job or with the BQ90LV2 page.

Implementation Consideration: If a budget line has been deleted, such records would continue to exist in the activity tables. The Budget Structure 90 Activity System Assurance report lists all such records irrespective of the selection BFY. In such cases, it is recommended to delete such records from the activity database (ACT_BUD_STRU_90_LVL_2 & ACT_BUD_STRU_90_LVL_4) if reporting them is not desired.

Process Steps	Messages
Parameter Validation	The process begins by validating the parameters. Depending on the parameter and validation, the following messages are logged: • Selection BFY is required • Selection BFY is invalid
Selection & writing to Report	After parameter validation, the process checks if the BFYs specified have been initialized (Application Parameter record for BUDINQ_ACTVNLYS_INITIAL_BFYS lists those years initialized – written to the detail inquiry). If the BFY has not been initialized, the process fails after writing the following message to the job log: • BFY: <*BFY*> has not been initialized If all BFYs have been initialized, the process starts comparing the budget amounts between Budget Structure 90 inquiry and Activity pages based on the BFY parameter. For budget lines where the amounts are out of sync the process writes the details to the report and write the following message to the job log: • *n* OOS record(s) processed.

Major Input

- BUD_STRU_90_LVL_2 (BQ90LV2)

- ACT_BUD_STRU_90_LVL_2 (Budget Activity page structure 90)

Minor Input

- IN_APP_CTRL (Application Control Parameter – Budget Structure 90 Inquiry Activity Initialized BFYs (BUDINQ_ACTVNLYS_INITIAL_BFYS))

Batch Parameters

Parameter	Description	Default Value
Commit Block Size (COMMIT_SIZE)	A required performance parameter to specify the number of records after which the job performs a commit.	1000
Select Block Size (SELECT_BLOCK)	A required performance parameter used in the selection of data for updates.	1000
Progression Message Block Size (PROG_CTR_SZ)	A required performance parameter used to specify the number of records after which the job should log progression messages.	5000
Selection BFYs (SELECTION_BFYS)	A required selection parameter of one or more Budget FYs for selection. Separate multiple values with a comma. Please see earlier note about the need to be initialized first.	No Default
Client Name (CLIENT_NM)	An optional name to appear in the header of the report generated.	No Default

Major Output

- Budget Structure 90 Activity System Assurance Report

Job Return Code

The following table shows the potential job Return Codes for the Budget Structure 90 Activity System Assurance Report job:

Return Code	Condition
Successful (1)	No record found with out of sync condition or only

	orphaned records found.
Warning (4)	N/A
Non-Fatal Error (8)	Record with at least one OOS condition.
Failed (12)	The job may fail under the following conditions: • Parameters validation failed. • Selection BFY is required • Selection BFY is invalid • BFY: <BFY> has not been initialized • Run time exceptions for unexpected situations.
Terminated (16)	This return code is issued when the job is terminated by the user.
System Failure (20)	This return code is issued when the job is terminated because of database server or network issues.

Sort Criteria

The records are sorted by UNID (from Budget Structure 90 Inquiry Level 2).

Selection Criteria

- The records are selected based on selection BFYs.
- The job selects orphaned records irrespective of the BFY specified.
- The process does not select the Charges amount because that amount is not part of the special query and is on the budget structure as a hidden field.

Problem Resolution

The process cannot be restarted on failure; however new job can be scheduled by fixing the issue.

The following table shows the possible return codes and recommendations for each processing steps:

Step 1: Parameter Validation

Possible Return Codes	Condition	Recommendation	Other Instructions
Successful (1)	Successful	N/A	N/A
Warning (4)	N/A	This step does not issue this return code.	This step does not issue this return code.

Possible Return Codes	Condition	Recommendation	Other Instructions
Non-Fatal Error (8)	N/A	This step does not issue this return code.	This step does not issue this return code.
Failed (12)	Required Parameters are not valid. **Sample Message:** "Selection BFY is invalid."	Enter a valid Budget Fiscal Year.	Reschedule job with appropriate parameter values.
	BFY not initialized. **Sample Message:** "BFY <BFY> has not been initialized"	Please initialize the Budget Fiscal Year using the Budget Structure 90 Activity Load Job.	Reschedule job with appropriate parameter values after initializing the Budget Fiscal Year.
	The job failed because of runtime exceptions for an unexpected situation.	The reason for the failure needs to be investigated before re-scheduling the job.	Reschedule job with appropriate parameter values.
Terminated (16)	The job is terminated manually by the user.	The reason for the termination needs to be investigated. The job can be rescheduled.	Schedule a new job.
System Failure (20)	The job is terminated because of database server or network issues.	The reason for the System Failure needs to be investigated. The job can be rescheduled.	Schedule a new job.

Step 2: Selection and Processing of Records

Possible Return Codes	Condition	Recommendation	Other Instructions
Successful (1)	Successful	N/A	N/A
Warning (4)	N/A	This step does not issue this return code.	This step does not issue this return code.
Non-Fatal Error (8)	OOS record/s processed. **Sample Message:** "*n* OOS record(s)	Research the reasons for OOS condition and address the issue.	Reschedule job with appropriate parameter values.

	processed”		
Failed (12)	The job failed because of runtime exceptions for an unexpected situation.	The reason for the failure needs to be investigated before rescheduling the job.	Reschedule job with appropriate parameter values.
Terminated (16)	The job is terminated manually by the user.	The reason for the termination needs to be investigated. The job can be rescheduled.	Schedule a new job.
System Failure (20)	The job is terminated because of database server or network issues.	The reason for the system failure needs to be investigated. The job can be rescheduled.	Schedule a new job.

2.1.6 Budget Sync Process - Sync Budget Actuals to Ledgers, Journals and Posting Line Catalog

Job Name	Budget Sync Process
Recommended Frequency	On demand. Note: SA01 **must** be run prior to running the Budget Sync Process. All transaction processing should be halted while running the Budget Sync Process. Verification that amounts reported by SA01 are accurate and budget line information is not must be performed.
Single Instance Required	Yes
Can be restarted?	Yes, the second and fourth jobs.
Reports generated	A report is generated from the preprocessor job

Overview

The overall purpose of the Budget Synchronization process is to update the accounting budget amounts (buckets) found on a budget and allotment inquiries (BQ##LV## page) and the audit level records associated with those amounts in the event of a Systems Assurance 1 (SA01) out of sync condition. That out of sync condition needs initial verification to ensure the amounts reported by SA01 from the Posting Line Catalog, Accounting Journal, and input ledger (normally LDGR SA BUD) are correct and the amount found on BQ##LV## and allotment inquiry pages are incorrect. In the condition the BQ##LV## and allotment inquiry pages are correct then another means is needed to correct the out of sync condition besides the Budget Sync process.

The Budget Sync chain has the following jobs, each of which is discussed in detail later:

1. Bud Sync Preprocessor
2. Bud Sync Budget Lines
3. Multi Process Build Audits
4. Bud Sync Post Processor

In brief, the primary task of the process is to update the accepted and pending accounting amounts on one or more budget lines. A secondary and optional task is to clear and rebuild the audit records (those transaction listings one sees by drilling down into a budget amount with the magnifying glass icon).

When creating audit records the program inserts them for all budget levels upwards from the line/level being corrected. This behavior is different that the adjustment of budget amounts where updates only occur at a single budget level. Since audit records are created in this manner, synchronizing them only requires 1 run of the process whereas the process runs multiple times to fix budget and allotment lines on a multiple-level budget structure.

As audit information can be massive, this process uses the Sync Facilitator (BUD_SYNC_FCLTR) table so the audit level posting work can be distributed across multiple jobs. The individual jobs within the chain exist in the Budget/Batch Jobs folder to make it easier for sites to run multiple instances of Multi-process Build Audit job. If the process is run with the individual jobs they must be run in order and have the same Run ID on each job.

The program only deals with accounting amounts and not the budget ones recorded in the Budget Journal.

Some reasons for an out of sync SA01 condition can include the following, each of which can be remedied by this process:

Catch Up a Budget Line:

1. A budget structure was once optional but is now required so that budget with or without allotment lines have to be created with COA combinations that have been used before.
2. A budget level was once optional but is now required so that budget with or without allotment lines have to be created with COA combinations that have been used before.

Correct a Budget Line:

1. Settings to update or not update a budget amount were incorrect on the Posting Code (PSCD) page and have now been corrected.
2. Transaction processing error where a budget update was made but not rolled back.
3. Data corruption on a BUD_STRU_##_LVL_## table

Note: When a budget line is out of sync in the SA01 report, the Budget Sync process can be used to sync that line with any child allotment lines using a single budget line UNID. However, custom queries need to be created to identify budget line UNID for budget lines that are in sync with allotment lines out of sync since such lines will not be listed in the SA01 report as out of sync.

Some key points for performance

It is not recommended to run the process once to sync all budget levels within a multi-level structure. The highest level recommended is a single structure and level. When lines at all levels within a structure need to be synchronized, the process should be run multiple times (one time per each structure/level), starting with the lowest level.

The journalization of all eligible posting line records should be done by running the Journal Engine. Even if using Real Time Journal Posting and no transaction codes defined as Asynchronous Posting on Transaction Control, the Journal Engine is recommended to catch those unintended situations where posting lines are not journalized. This should be standard procedure for Systems Assurance 01 runs.

The ledgerization of all eligible journal records should be done by running the Ledger Engine in all three modes: normal, gap processing, and failed work processing. This should be standard procedure for Systems Assurance 01 runs.

Sites should chose to run the process to only update the bucket amounts first, and then later, when they have more time, run the process to build the audit levels. Use of multiple instances of the audit building job are highly recommended, even if synchronizing a single budget line as a single line could have been updated by hundreds of transactions.

When a very large rebuild is necessary or very frequent rebuilds are being done, it is recommended that indices be created on the System Assurance Ledger (LDGR_SA_BUD or other ledger being used as input), JRNL_ACTG and PSTNG_LN_CAT to make the queries used in the process run more efficiently. In each of these indices, columns will vary based on the budget structure and level to be synchronized.

For the structure to be synched get the primary key of parent budget level, excluding UNID from it. E.g. for structure 80 primary key of BUD_STRU_80_LVL_1 is BFY, FUND_CD, DEPT_CD, and APPR_CD.

1. To make pre-processor logic more efficient create index on LDGR_SA_BUD and JRNL_ACTG on these columns.
2. To make budget sync lines step more efficient create indices on PSTNG_LN_CAT on these columns and create one more index on PSTNG_LN_CAT on DOC_PHASE_CD, JRNL_PSTNG_IND, BUD_PSTNG_IND, PSTNG_CD_ID.
3. To make post processor step more efficient create index on audit level of structure to be synched on parent levels primary key columns.

Syntax to create index is as follows:

1. Create index <table_name>_BudSync on <table name> (< comma separated list of parent budget level primary key columns>)

Major Input of the Chain

- Input Text File (BudSyncParms.txt) - This parameter file will be the means by which the site specifies which Budget Level and/or Budget Record Unique ID should be corrected.
- Systems Assurance 01 Temporary Table (SA_BUD)
- Input Ledger (likely LDGR_SA_BUD)
- Budget Structure & Level (BUD_STRU_##_LVL_##)
- Allotment Level (ALOT_STRU_##_LVL_##)
- Budget Audit Level (BUD_STRU_##_LVL_##)
- Accounting Journal (JACTG / JRNL_ACTG)
- Posting Line Catalog (PSTNG_LN_CAT)
- Journal/Ledger Control Detail (JLCTRL / JRNL_LDGR_CTRL)
- Journal/Ledger Cross Reference (JLXREF / JRNL_LDGR_XREF)
- Posting Code (PSCD / R_PSCD) table

Major Output of the Chain

- Budget Sync Workload (BUD_SYNC_WRKLD)
- Budget Sync Facilitator (BUD_SYNC_FCLTR)
- Synchronize Budgets (SYNC_BUD)
- Budget Structure & Level (BUD_STRU_##_LVL_##)
- Allotment Level (ALOT_STRU_##_LVL_##)
- Audit Level (BUD_STRU_##_LVL_## & ALOT_STRU_##_LVL_##)
- Budget Sync Report

Chain Job Return Code

The following table shows the potential return codes for the Budget Sync chain. Note that the chain will end with the highest return code across all of the individual jobs.

Return Code	Condition
Successful (1)	All of the jobs end successfully.
Warning (4)	Chain does not use this return code.
Non Fatal (8)	Chain does not use this return code.

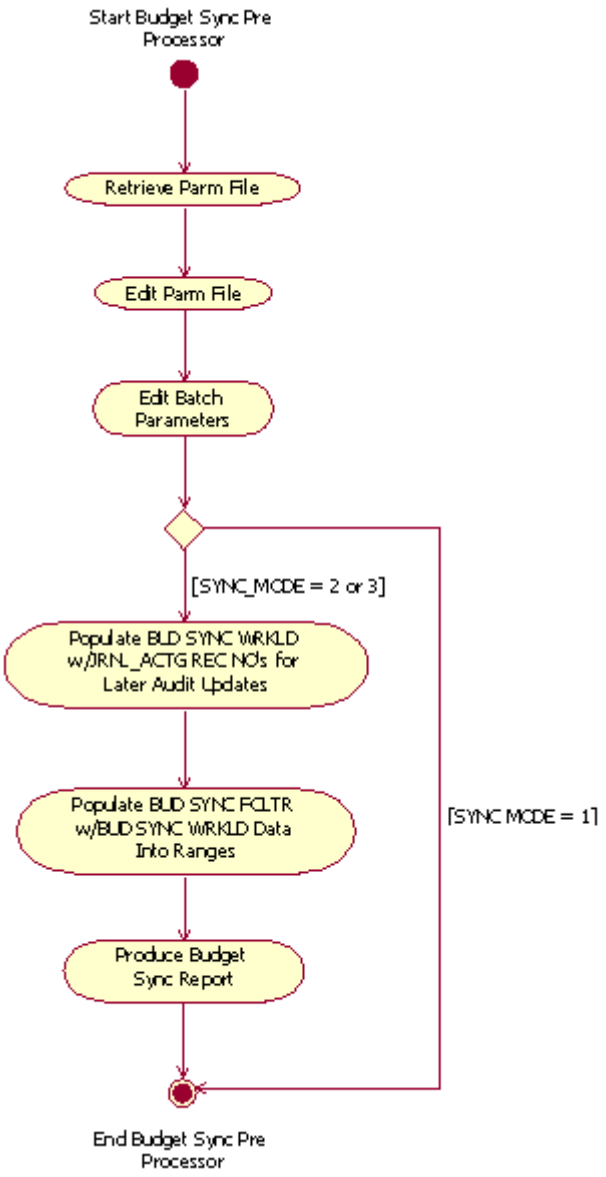
Failed (12)	One of the jobs in the chain ends with a return code of “Failed”.
Terminated (16)	One of the jobs in the chain ends with a return code of “Terminated”.
System Failure (20)	One of the jobs in the chain ends with a return code of “System Failure”.

Problem Resolution

Please refer to the individual job “Problem Resolution” section for more detail.

Budget Sync Process Chain: Bud Sync Pre Processor Job

Job Name	Budget Sync Preprocessor
Recommended Frequency	On Demand. This job can be run as part of the Budget Sync Process chain or it can be run independently through a Job Interaction Client (JIC) script or manually scheduled.
Single Instance Required	Yes
Can be restarted?	No
Reports generated	Yes – Budget Sync Report



The Budget Sync Pre Processor job step, the first in the chain or the first in a series of independent jobs, performs the processing depicted in the activity diagram to the left.

After various validations, the first significant processing task of the job step is one only performed if rebuilding audit level records. The job step locates all the Accounting Journal records, containing posting codes defined to update the type of budget structure being corrected, which match the budget line(s) being corrected. Retrieving the Accounting Journal record numbers for SA_BUD records for unledgerized journal activity (PSTNG SRC = 2) is straight forward.

Retrieving the journal records for SA_BUD record of ledgerized activity requires the Journal/Ledger Cross-Reference (JRNL_LDGR_XREF) table. The job step locates records from the input ledger (likely LDGR_SA_BUD as it is used for SA1 at most sites) that match the budget line(s) being corrected with posting codes defined to update the type of budget structure being corrected. That ledger record ID is then located on the JRNL_LDGR_XREF table, from which all journal record numbers are identified.

All identified journal record numbers are written to the BUD SYNC WORKLD table and assigned sequential ID numbers.

The records in the SYNC WORKLD table are then defined into ranges on the BUD SYNC FCLTR table for the Multi Process Build Audits job step to use.

Lastly, the Budget Sync Report is produced by the job step.

The report displays the Budget Structure(s) and Level(s) that were successfully processed. UNID is additionally included when Sync Level is Sync Budget Line. The report lists details of Budget lines for which corresponding SA_BUD records were not found. There is a common reason an SA_BUD record would not be found for a budget line – budget line has not been updated by any accounting

transaction yet. Here there is no sync to be performed. The other is that the SA1 run used to populate SA_BUD did not use the proper parameters to select enough information to create an SA_BUD record for the budget line.

The job log messages produced for each processing step are detailed in the following table.

Process Steps	Messages
1. Parameter Validation	• Any invalid parameter will be listed. • All valid parameters are listed.
2. Report Creation	• Reports output folder mapped • Rendering report started • Rendering report completed • Report generated
3. Populate Budget Sync Workload Table	• Populate Budget Sync Workload table step is done
4. Populate Budget Sync Facilitator Table	• Populate Budget Sync Facilitator table step is done

Major Input

Many application parts are involved in this job step. Each is listed out below and then interconnected in a class diagram to show the relationships and key attributes on each.

This process reads in the following:

1. Input Text File (BudSyncParms.txt) - This parameter file will be the means by which the site specifies which Budget Level and/or Budget Record Unique ID should be corrected. The file will have the following format:

```
STRUCTURE = x
LEVEL = y
UNID = z
_PARAM_LINE_
```

x is replaced by the number of the Budget Structure (e.g. STRUCTURE = 1);
y is replaced by the level of the Budget Structure (e.g. LEVEL = 2); and
z is replaced by the Unique ID of the Budget Record to be corrected (e.g. UNID = 12345). If all records for a Budget Level need to be corrected, z should be blank.

Multiple _PARAM_LINE_ records may be included in the Input Text File. However, each must follow the UNID precedent of the first _PARAM_LINE_. If the UNID is blank for the first _PARAM_LINE_, all of the UNIDs in the file must be blank. Conversely, if the first UNID is specified, all of the UNIDs must be specified. The process uses Budget Record Unique ID for syncing allotment lines.

Below are several examples of the text file:

1) Correct an entire budget level

```
STRUCTURE = 30
LEVEL = 1
UNID =
_PARAM_LINE_
```

2) Correct a single budget line on a single budget level

```
STRUCTURE = 30
LEVEL = 1
UNID = 56487
_PARAM_LINE_
```

3) Correct multiple budget lines on a single budget level

```
STRUCTURE = 30
LEVEL = 1
UNID = 54478
_PARAM_LINE_
STRUCTURE = 30
LEVEL = 1
UNID = 54479
_PARAM_LINE_
```

2. Systems Assurance 01 Temporary Table (SA_BUD)
3. Input Ledger (likely LDGR_SA_BUD)
4. Budget Structure & Level (BUD_STRU_##_LVL_##)
5. Allotment Level (ALOT_STRU_##_LVL_##)
6. Accounting Journal (JACTG / JRNL_ACTG)
7. Journal/Ledger Control Detail (JLCTRL / JRNL_LDGR_CTRL)*
8. Journal/Ledger Cross Reference (JLXREF / JRNL_LDGR_XREF)*
9. Posting Code (PSCD / R_PSCD) used to determine what posting codes update the type of budget

Major Output

Several application parts are updated by this job step. Each is listed out below.

The process updates the following tables:

1. Budget Sync Workload (BUD_SYNC_WRKLD)
2. Budget Sync Facilitator (BUD_SYNC_FCLTR)
3. Budget Sync Report

Batch Parameters

Parameter	Description	Default Value
Parameter Location at Bud Sync Preprocessor job AMSPARM	Parameter Directory Location	$$AMSROOT$$/ Parms

Block Size – Number of records that can be sent to budget posting routine BLOCK_SIZE	Required parameter to provide a number of records that can be sent to the budget posting routine at one time. It is used by the preprocessor to determine how many records from the Workload table should be identified for a single facilitator table entry. This parameter must be a positive, non-zero integer.	100
Build Zero Dollar SA BUD Record BLD_ZERO_SABUD_RCRD	Parameter set to True when user needs to synch a Budget Line to zero dollars but Budget Line is not zero dollars. Use only when rebuilding an individual budget line (UNID in parameter file and SYNC_LVL parameter is 2) that has been confirmed as out of sync and has no SA_BUD record created from the System Assurance 1. Use extreme caution as this parameter will cause a zero-dollar line to be inserted into SA_BUD as part of the Budget Sync Process to ensure all non-zero accounting amounts on the budget line are set to zero. Please ensure SA1 run had a broad enough selection of BFY values to ensure SA_BUD is not missing a record by mistake.	false
Client Name CLIENT_NM	Optional parameter to appear on the generated report header.	<blank>
Commit Block COMMIT_BLOCK	This required parameter specifies the number of record updates to occur before a commit is performed. In order to free up memory, the Budget Synchronization Process will need to perform periodic commits. This parameter must be a positive, non-zero integer.	100
Allotment Level ALOT_LVL	This optional parameter specifies the budget level where allotment is defined for the Budget Structure. Values need to be specified with run to sync allotment lines.	
Input Text File INPUT_TEXT_FILE	Required parameter to supply file name of the input file that specifies the budget level or budget lines to be rebuilt. If the default value is changed, then it must be changed on all subsequent jobs as well. All jobs must have the same Input Text File.	BudSync_Parm.txt
Ledger Name LDGR_NM	Required name of the ledger where records are retrieved from. Should be the same ledger that is input into the Systems Assurance 01 process.	LDGR_SA_BUD
Report ID	An optional Report ID to appear on the	NDBUDSYNREP

REPORT_ID	generated report header.	
Restart Flag (Y-Yes, N-No) RESTART_FL	Required parameter with valid values of Y for Yes and N for No. It specifies whether the process is being restarted to handle a prior run's failed processing. The process will issue errors if a valid value is not supplied or if the RESTART_FL is set to Y and there are no failed records to process. Although this first job step itself cannot be restarted, it would be part of a restarted chain. In that event the parameter is Y, and the job step wouldn't perform any action other than parameter validation.	N
Run ID RUN_ID	Required parameter that will normally not be specified by the operator when the job is run. It will be automatically set to the Chain ID by the process. However, in the event of a manual restart, the operator will need to specify the RUN_ID of the last prior run that ended in a failed status. *If the four jobs are run separately (not in a chain), the Default Run ID must be entered. It can be set to '1', but then '1' must be used as the Run ID on all subsequent jobs.* Although this first job step itself cannot be restarted, it would be part of a restarted chain. In that event, the job step wouldn't perform any action other than parameter validation.	$$@CHAINJOBID@$$
Sync Allotment (Y-Yes, N-NO) SYNC_ALOT	This required parameter indicates whether allotment lines need to be synched for the budget line. Valid values are Y for Yes and N for No, with a default value of N.	N
Sync Level (1-Structure/Level, 2-Sync Budget Line) SYNC_LVL	Required parameter to specify whether the INPUT_TEXT_FILE will contain budget line unique IDs or whether the file specifies a budget level. The valid values are: 1 - indicating Structure/Level only and 2 - indicating individual Budget Lines. Errors will be issued if the SYNC_LVL does not equal 1 or 2, or if the SYNC_LVL does not match with the data provided in the INPUT_TEXT_FILE. If the default value is changed then it must be changed on jobs 2 and 4 so all have the same value.	2
Sync Mode (1-Sync Buckets, 2- Sync	Required parameter to define whether the process is being run to update budget buckets, rebuild the audit level records, or	3

Audit, 3-Sync Both) SYNC_MODE	both. Valid values are: 1 - indicating that the process will update the budget buckets only; 2 - indicating that the process will update the audit level records only; and 3 - indicating that the process will update both budget buckets and audit levels. An error will be issued if this parameter does not equal 1, 2 or 3. If the default value is changed then it must be changed on all subsequent jobs. All jobs must have the same value.	

NOTE: The Job Setup information delivered contains descriptions, override controls, and default values delivered for the jobs in the chain and individual jobs. If any of this information needs to be adjusted permanently for all future runs, instead of being changed on individual runs, then please use the Job Setup page. Keep in mind those parameters that need to be entered on all jobs with identical values. This chain does not pass these values down from the preprocessor job step to later job steps.

Job Return Code

The following table shows the potential job return codes for the Budget Sync Preprocessor job.

Return Code	Condition
Successful (1)	• All of the selected records are processed successfully.
Warning (4)	• Job does not use this return code.
Non Fatal (8)	• Job does not use this return code.
Failed (12)	• The job will fail under the following conditions: • Parameters are invalid • Input text file not found • When this job ends with a return code of Failed, subsequent jobs in the chain will be set to inactive.
Terminated (16)	• This return code will be issued when the job is terminated by the user. When this job ends with a return code of Terminated subsequent jobs in the chain will be set to inactive.
System Failure (20)	• This return code will be issued when the job is terminated because of database server or network issues. When this job ends with a return code of System Failure, subsequent jobs in the chain will be set to inactive.

Sort Criteria

The report is ordered by Structure ID, Level ID, and UNID.

Selection Criteria

Based on the Input Text file, one or more budget lines are selected.

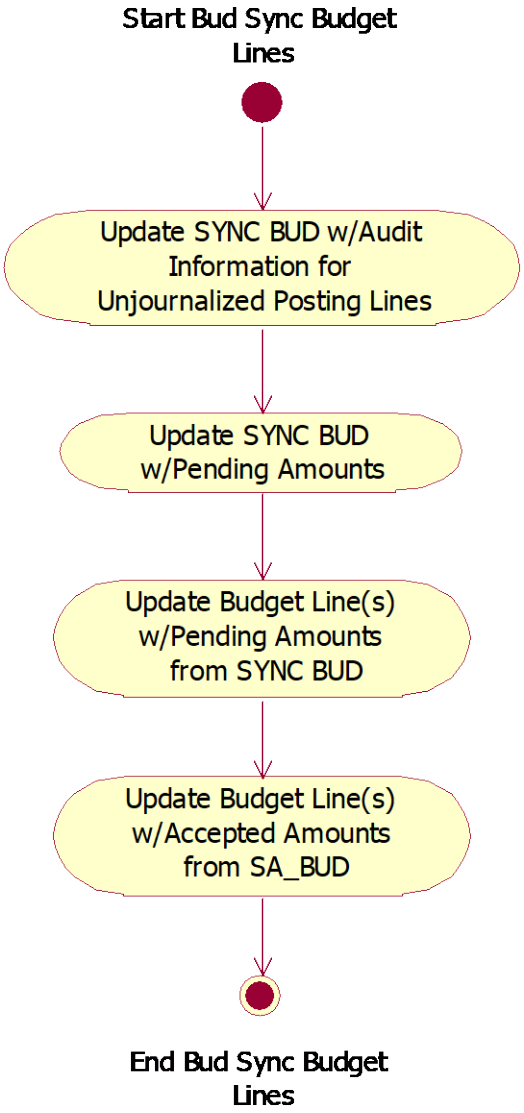
All Accounting Journal records are selected that have posting codes defined to updated the type of budget structure (expense or revenue) specified in the parameter file.

Problem Resolution

The pre processor step does not have any checkpoint logic. If Budget Sync Process Chain failed while executing this job, no database restore is required. One could submit a new chain after addressing the failure issue or submit a new individual job can be submitted, if not running as a chain.

Budget Sync Process Chain: Budget Sync Lines Job

Job Name	Budget Sync Lines
Recommended Frequency	On Demand. This job can be run as part of the Budget Sync Process chain or it can be run independently through a Job Interaction Client (JIC) script or manually scheduled. If run as a single thread, a Bud Sync Preprocessor must have finished successfully before.
Single Instance Required	Yes
Can be restarted?	Yes
Reports generated	No



The second job of the Budget Sync Process starts with updates to the SYNC_BUD table with audit information for unjournalized posting lines. Only this job step uses the Posting Line Catalog. Unjournalized posting lines should be eliminated from the process by running the Journal Engine before Systems Assurance 01.

The Sync Budget Lines job then goes on to update SYNC_BUD with pending amounts from the Posting Line Catalog. These SYNC_BUD records are unique because the main type of record on the table is a transaction ID for audit level rebuilding. Although Systems Assurance 01 does not deal with pending amounts, the budget synchronization process does update them. There is no practical way to eliminate pending transactions. They will slow the total time of the process, but they cannot be avoided.

Updates to the one or more budget/Allotment lines being synchronized now starts with the job step taking the pending amount records from SYNC_BUD and updating the BUD_STRU_##_LVL_## and ALOT_STRU_##_LVL_## tables. Updated to the accepted budget amounts then done with information from SA_BUD, which is why no transaction processing should occur between the population of SA_BUD and this job step. These two updates are why it is critical that no transaction processing occur after the SA_BUD table has been populated.

The job log messages produced for each processing step are detailed in the following table.

Process Steps	Messages
1. Parameter Validation	• Any invalid parameter will be listed. • All valid parameters are listed.
2. Update SYNC_BUD	• Reposting of Audit Records for Posting Lines is done
3. Repost Pending Amounts	• Reposting of Pending Buckets for Posting Lines is done
4. Update Pending	• No messages are issued for this step

Amounts	
5. Update Accepted Amounts	• Event Driven buckets on Budget Lines were synced

Major Input

Many application parts are involved in this job step. Each is listed out below and then interconnected in a class diagram to show the relationships and key attributes on each.

This process reads in the following:

1. Input Text File (BudSyncParms.txt) - This parameter file will be the means by which the site specifies which Budget Level and/or Budget Record Unique ID should be corrected.
2. Systems Assurance 01 Temporary Table (SA_BUD)
3. Posting Line Catalog (PSTNG_LN_CAT)

Major Output

Several application parts are updated by this job step. Each is listed out below.

This process updates the following:

1. Synchronize Budgets (SYNC_BUD)
2. Budget Structure & Level (BUD_STRU_##_LVL_##)
3. Allotment Level (ALOT_STRU_##_LVL_##)

Batch Parameters

Parameter	Description	Default Value
Parameter Location at Bud Sync Budget Lines job AMSPARM	Parameter Directory Location	$$AMSROOT$$/Parms
Block Size – Number of records that can be sent to budget posting routine BLOCK_SIZE	Required. This field provides the number of records that can be sent to the budget posting routine at one time. It will be used by the preprocessor to determine how many records from the Workload table should be identified for a single facilitator table entry. This parameter must be a positive, non-zero integer.	100
Commit Block – Used for interval commits COMMIT_BLOCK	Required. In order to free up memory, the Budget Synchronization Process will need to perform periodic commits. This required parameter will specify the number of record updates to occur before a commit is performed. This parameter must be a positive, non-zero integer.	100
Allotment Level ALOT_LVL	This optional parameter specifies the budget level where allotment is defined	

	for the Budget Structure. Values need to be specified with run to sync allotment lines.	
Input Text File INPUT_TEXT_FILE	Required. This defines the full path and file name for the input file that specifies the Budget Records that need to be rebuilt.	BudSync_Parm.txt
Restart Flag (Y-Yes, N-No) RESTART_FL	Required. This required parameter, with valid values of Y for Yes and N for No, specifies whether the process is being restarted to handle a prior runs failed processing. The process will issue errors if a valid value is not supplied, or if the RESTART_FL is set to Y(Yes) and there are no failed records to process.	N
Run ID RUN_ID	Required. This parameter will normally not be specified by the operator when the job is run. It will be automatically set to the Chain ID by the process. However, in the event of a manual restart, the operator will need to specify the RUN_ID of the last prior run that ended in a failed status. If these jobs are run separately then the Default Run ID must be changed. It can be changed to '1', but then '1' must be used as the Run ID on all subsequent jobs.	$$@CHAINJOBID@$$
Sync Allotment (Y-Yes, N-NO) SYNC_ALOT	This required parameter indicates whether allotment lines need to be synched for the budget line. Valid values are Y for Yes and N for No, with a default value of N.	N
Sync Level (1-Structure/Level, 2- Sync Budget Line) SYNC_LVL	Required. This required parameter specifies whether the INPUT_TEXT_FILE will contain Budget Record Unique IDs or whether the parameter file will only contain Budget Levels. The valid values are: 1, indicating Structure/Level only; and 2, indicating individual Budget Lines. Errors will be issued if the SYNC_LVL does not equal 1 or 2, or if the SYNC_LVL does not match with the data provided in the INPUT_TEXT_FILE.	2
Sync Mode (1-Sync Buckets, 2- Sync Audit, 3-Sync Both) SYNC_MODE	Required. This required parameter defines whether the process is being run to update the Budget Buckets, rebuild the Audit Level records, or both. Valid values are: 1, indicating that the process will update the Budget Buckets only; 2, indicating that the process will update the Audit Level records only; and 3, indicating	3

	that the process will update both Budget Buckets and Audit Levels. An error will be issued if this parameter does not equal 1, 2 or 3.	

NOTE: The Job Setup information delivered contains descriptions, override controls, and default values delivered for the jobs in the chain and individual jobs. If any of this information needs to be adjusted permanently for all future runs, instead of being changed on individual runs, then please use the Job Setup page. Keep in mind those parameters that need to be entered on all jobs with identical values. This chain does not pass these values down from the preprocessor job step to later job steps.

Job Return Code

The following table shows the potential job return codes for the Budget Sync – Sync Budget Lines job.

Return Code	Condition
Successful (1)	All of the selected records are processed successfully.
Warning (4)	Job does not use this return code.
Non Fatal (8)	Job does not use this return code.
Failed (12)	The job will fail under the following conditions: • Parameters ware invalid • When this job ends with a return code of Failed, subsequent jobs in the chain will be set to inactive.
Terminated (16)	This return code will be issued when the job is terminated by the user. When this job ends with a return code of Terminated subsequent jobs in the chain will be set to inactive.
System Failure (20)	This return code will be issued when the job is terminated because of database server or network issues. When this job ends with a return code of System Failure, subsequent jobs in the chain will be set to inactive.

Sort Criteria

Records selected from Posting Line Catalog are ordered by Transaction Code, Transaction Department Code, Transaction ID, Transaction Version Number, Internal Transaction Posting Number.

Records selected from SA_BUD are ordered by Structure ID, Level ID, UNID and Posting Source.

Selection Criteria

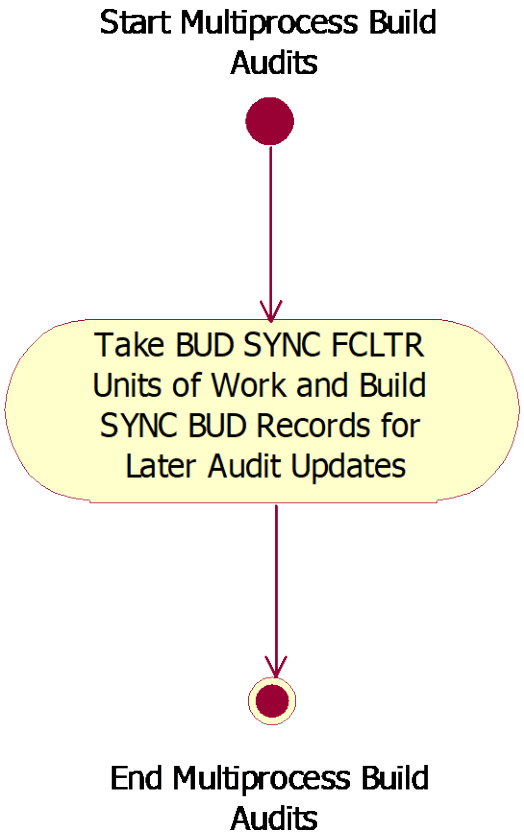
1. All unjournalized and pending bucket amounts on the Posting Line Catalog are selected. Please note that you may see other budget structures, levels and lines posted to SYNC_BUD but these records will not be used.
2. SA_BUD bucket amounts for the Budget Line(s) or Budget/Level(s) indicated on the Input Text File.

Problem Resolution

The Sync Budget Line job has checkpoint logic. If it fails, no database restore is required. One could submit a new chain after addressing the failure issue or submit a new individual job, if not running a chain. The job will start from the last checkpoint stored. Using the same RUN_ID and setting the RESTART_FL to Y are required.

Budget Sync Process Chain: Multi Process Build Audits Job

Job Name	Multi Process Build Audits
Recommended Frequency	On Demand. This job can be run as part of the Budget Sync Process chain or it can be run independently through a Job Interaction Client (JIC) script or manually scheduled. If run as a single thread, a Bud Sync Preprocessor and Bud Sync Budget Lines must have finished successfully before.
Single Instance Required	No
Can be restarted?	No
Reports generated	No



This is the third job of the Budget Sync Process. This job takes the BUD_SYNC_FCLTR units of work and builds SYNC_BUD records for later Audit updates. The number of steps may seem simple, but this job step may have to perform a very large amount of work. As the job selects a record from the facilitator table a range of Accounting Journal records is specified in that record. The job gets each record from the journal and updates the SYNC_BUD table with transaction information and all other fields needed to later move the SYNC_BUD record directly to the audit level table.

The ratio of Accounting Journal records to SYNC_BUD records is a 1:1 relationship. The next job in the process will ensure that audit level records are created.

Process Steps	Messages
1. Parameter Validation	• Any invalid parameter will be listed. • All valid parameters are listed.
2. Processes Budget Sync Facilitator records	• Processed all Budget Sync Facilitator records

Major Input

Many application parts are involved in this job step. Each is listed out below and then interconnected in a diagram to show the relationships and key attributes on each.

This process reads in the following:

1. Budget Sync Facilitator (BUD_SYNC_FCLTR)
2. Budget Structure & Level (BUD_STRU_##_LVL_##)
3. Allotment Level (ALOT_STRU_##_LVL_##)
4. Accounting Journal (JACTG / JRNL_ACTG)

Major Output

Several application parts are updated by this job step. Each is listed out below.

This process updates the following:

- Synchronize Budgets (SYNC_BUD)

Batch Parameters

Parameter	Description	Default Value
Run ID RUN_ID	Required. This parameter will normally not be specified by the operator when the job is run. It will be automatically set to the Chain ID by the process. However, in the event of a manual restart, the operator will need to specify the RUN_ID of the last prior run that ended in a failed status. If these jobs are run separately then the Default Run ID must be changed. It can be changed to '1', but then '1' must be used as the Run ID on all subsequent jobs.	$$@CHAINJOBID@$$
Sync Allotment (Y-Yes, N-NO) SYNC_ALOT	This required parameter indicates whether allotment lines need to be synched for the budget line. Valid values are Y for Yes and N for No, with a default value of N.	N
Sync Level (1-Structure/Level, 2-Sync Budget Line) SYNC_LVL	Required. This required parameter specifies whether the INPUT_TEXT_FILE will contain Budget Record Unique IDs or whether the parameter file will only contain Budget Levels. The valid values are: 1, indicating Structure/Level only; and 2, indicating individual Budget Lines. Errors will be issued if the SYNC_LVL does not equal 1 or 2, or if the SYNC_LVL does not match with the data provided in the INPUT_TEXT_FILE.	2
Sync Mode (1-Sync Buckets, 2- Sync Audit, 3-Sync Both) SYNC_MODE	Required. This required parameter defines whether the process is being run to update the Budget Buckets, rebuild the Audit Level records, or both. Valid values are: 1, indicating that the process will update the Budget Buckets only; 2, indicating that the process will update the Audit Level records only; and 3, indicating that the process will update both Budget Buckets and Audit Levels. An error will be issued if this parameter does not equal 1, 2 or 3.	3

NOTE: The Job Setup information delivered contains descriptions, override controls, and default values delivered for the jobs in the chain and individual jobs. If any of this information needs to be

adjusted permanently for all future runs, instead of being changed on individual runs, then please use the Job Setup page. Keep in mind those parameters that need to be entered on all jobs with identical values. This chain does not pass these values down from the preprocessor job step to later job steps.

Job Return Code

Return Code	Condition
Successful (1)	Processed all Budget Sync Facilitator records.
Warning (4)	Job does not use this return code.
Non Fatal (8)	Job does not use this return code.
Failed (12)	The job will fail under the following conditions: • Parameters are invalid When this job ends with a return code of Failed, subsequent jobs in the chain will be set to inactive.
Terminated (16)	This return code will be issued when the job is terminated by the user. When this job ends with a return code of Terminated subsequent jobs in the chain will be set to inactive.
System Failure (20)	This return code will be issued when the job is terminated because of database server or network issues. When this job ends with a return code of System Failure, subsequent jobs in the chain will be set to inactive.

Sort Criteria

Records selected from Accounting Journal are ordered by Record Number.

Selection Criteria:

All unjournalized and pending bucket amounts on the Posting Line Catalog are selected that have posting codes defined to update the type of budget structure (expense or revenue).

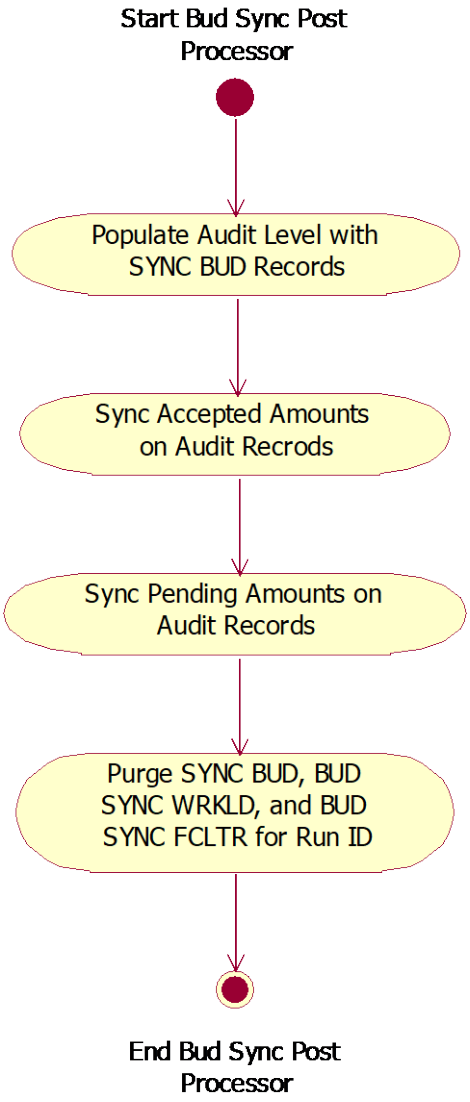
Problem Resolution

The Build Audit Record step does not have any checkpoint logic. If chain or a single instance of the job failed, no database restore is required. Instead, submit one or more stand alone instances of the job with a RUN_ID that matches the value used in the Preprocessor step.

Budget Sync Process Chain: Bud Sync Post Processor Job

Job Name	Bud Sync Post Processor
Recommended Frequency	On Demand. This job can be run as part of the Budget Sync Process chain or it can be run independently through a Job

	Interaction Client (JIC) script or manually scheduled. If run as a single thread, a Bud Sync Preprocessor, Bud Sync Budget Lines, and Multi Process Build Audits jobs must have finished successfully before.
Single Instance Required	Yes
Can be restarted?	Yes
Reports generated	No



The fourth and final job of the Budget Sync Process starts with the transfer of records from the SYNC_BUD table to the audit level table for the budget structure being updated. That transfer is a 1:1 relationship.

After the creation of audit records, they are synchronized with the accepted budget amounts and then the pending budget amounts.

The last step in this final job is to purge the SYNC BUD, BUD_SYNC_WRKLD and the BUD_SYNC_FCLTR tables for the given Run ID.

Process Steps	Messages
1. Parameter Validation	• Any invalid parameter will be listed. • All valid parameters are listed.
2. Audit Record Creation	• Audit Lines basic buckets are synched.
3. Sync Basic Buckets	• Audit Lines basic buckets are synched.
4. Sync Pending Buckets	• Audit Lines pending buckets are synched.
5. Purge Tables	• No messages are issued for this step.

Major Input

Many application parts are involved in this job step. Each is listed out below and then interconnected in a class diagram to show the relationships and key attributes on each.

This process reads in the following:

- Synchronize Budgets (SYNC_BUD)

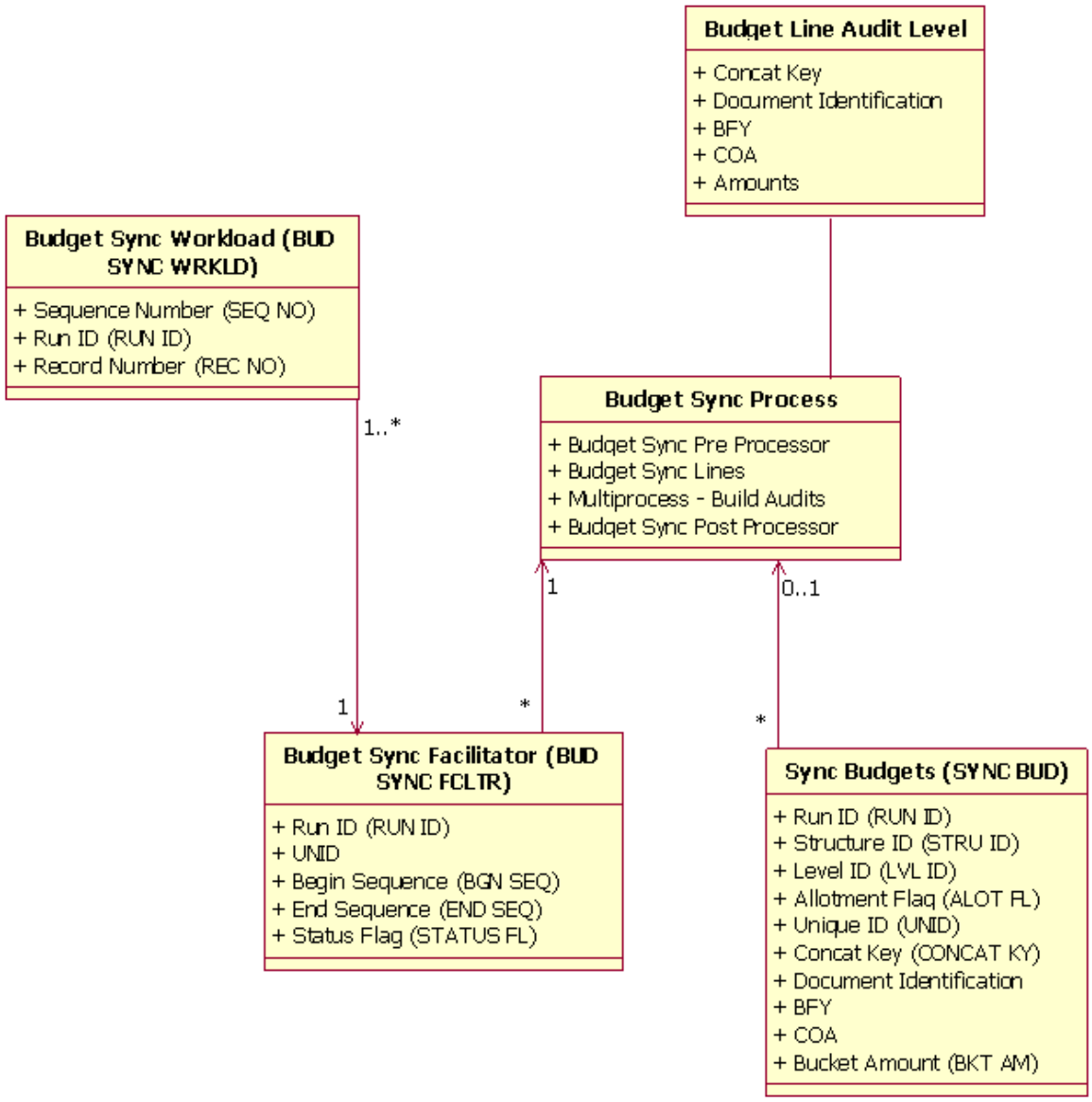
Major Output

Several application parts are updated by this job step. Each is listed out below and then interconnected in a class diagram to show the relationships and key attributes on each. The diagram is not intended to be one that shows data flow.

This process updates the following:

1. Audit Level (BUD_STRU_##_LVL_##, ALOT_STRU_##_LVL_##)
2. Budget Sync Workload (BUD_SYNC_WRKLD)
3. Budget Sync Facilitator (BUD_SYNC_FCLTR)
4. Synchronize Budgets (SYNC_BUD)

Class Diagram – Budget Sync Post Processor



Batch Parameters

Parameter	Description	Default Value
Parameter location at Bud Sync Post Processor AMSPARM	Parameter Directory Location	$$AMSROOT$$/Parms
Commit Block -Used for interval commits COMMIT_BLOCK	Required. In order to free up memory, the Budget Synchronization Process will need to perform periodic commits. This required parameter will specify the number of record updates to occur	100

Parameter	Description	Default Value
	before a commit is performed. This parameter must be a positive, non-zero integer.	
Allotment Level ALOT_LVL	This optional parameter specifies the budget level where allotment is defined for the Budget Structure. Values need to be specified with run to sync allotment lines.	
Input Text File INPUT_TEXT_FILE	Required. This defines the full path and file name for the input file that specifies the Budget Records that need to be rebuilt.	BudSync_Parm.txt
Run ID RUN_ID	Required. This parameter will normally not be specified by the operator when the job is run. It will be automatically set to the Chain ID by the process. However, in the event of a manual restart, the operator will need to specify the RUN_ID of the last prior run that ended in a failed status. If these jobs are run separately then the Default Run ID must be changed. It can be changed to '1', but then '1' must be used as the Run ID on all subsequent jobs.	$$@CHAINJOBID@$$
Sync Allotment (Y-Yes, N-NO) SYNC_ALOT	This required parameter indicates whether allotment lines need to be synched for the budget line. Valid values are Y for Yes and N for No, with a default value of N.	N
Sync Level (1-Structure/Level, 2-Sync Budget Line) SYNC_LVL	Required. This required parameter specifies whether the INPUT_TEXT_FILE will contain Budget Record Unique IDs or whether the parameter file will only contain Budget Levels. The valid values are: 1, indicating Structure/Level only; and 2, indicating individual Budget Lines. Errors will be issued if the SYNC_LVL does not equal 1 or 2, or if the SYNC_LVL does not match with the data provided in the INPUT_TEXT_FILE.	2
Sync Mode (1-Sync Buckets, 2- Sync Audit, 3-Sync Both) SYNC_MODE	Required. This required parameter defines whether the process is being run to update the Budget Buckets, rebuild the Audit Level records, or both. Valid values are: 1, indicating that the process will update the Budget Buckets only; 2, indicating that the process will update the Audit Level records only; and 3, indicating that the process will update	3

Parameter	Description	Default Value
	both Budget Buckets and Audit Levels. An error will be issued if this parameter does not equal 1, 2 or 3.	

NOTE: The Job Setup information delivered contains descriptions, override controls, and default values delivered for the jobs in the chain and individual jobs. If any of this information needs to be adjusted permanently for all future runs, instead of being changed on individual runs, then please use the Job Setup page. Keep in mind those parameters that need to be entered on all jobs with identical values. This chain does not pass these values down from the preprocessor job step to later job steps.

Job Return Code

Return Code	Condition
Successful (1)	Audit Lines basic buckets are synched. Audit Lines pending buckets are synched.
Failed (12)	The job will fail under the following conditions: • Parameters are invalid When this job ends with a return code of Failed, subsequent jobs in the chain will be set to inactive.
Terminated (16)	This return code will be issued when the job is terminated by the user. When this job ends with a return code of Terminated subsequent jobs in the chain will be set to inactive.
System Failure (20)	This return code will be issued when the job is terminated because of database server or network issues. When this job ends with a return code of System Failure, subsequent jobs in the chain will be set to inactive.

Problem Resolution

The Sync Budget Post Processor job has checkpoint logic. If it fails, no database restore is required. One could submit a new chain after addressing the failure issue or submit a new individual job, if not running a chain. The job will start from the last checkpoint stored. Using the same RUN_ID is required.

2.1.7 Calculate Budget Controls Amounts

When to Run

The Process can be run in Initialization mode during initial data setup for the client to initialize the Budget Controls Amounts Table with computed Controls Amounts for the existing Budget lines.

The Process can be run in Re-Calculation mode on-demand when Budget Controls go dirty or when a new Budget Control is added.

Description

The process computes Left hand side and Right hand side formulae for Budget Controls for Budget Lines and stores them on the Budget Controls Amounts table. The process can be run in two different modes:

1. Initialization mode

   This mode will be typically run during the initial stage of setting day zero entries for a client and after the client has established budgets. The Budget Controls Amounts table should be empty for this mode to execute. In this mode, the process gets all active Budget Controls defined in the System, computes the LHS and RHS formulae for each active applicable Budget constraint against each budget line and stores them on the Budget Controls Amounts table.

   Please Note:

   Active Budget Control means a Budget Control having a Default Violation set to any value other than Not Selected.

   Budget Control can be active and applicable for a given Budget Line if it is active at any one of the following levels: System-wide, Budget Structure Level wide, Fund level or Line Level.

2. Recalculation mode

   This mode can be run when any of the active Budget Controls have gone dirty (any activity that caused a change of value needing Budget Controls to be re-computed for all budget lines) and there is a requirement to re-compute all the Budget Controls LHS and RHS formulae on the Budget Constraint Amounts table. This mode can also be run when new active Budget Controls are introduced into the system and there is a requirement to compute LHS and RHS formulae for the newly introduced Budget Control against each budget line and store them on the Budget Controls Amounts table. The newly introduced Budget Control should be marked as dirty for it to be picked up by this process. In this mode, the process gets all dirty and active Budget Controls, re-computes the LHS and RHS formulae for each dirty and active Budget Control for all entries in the Budget Constraint Amounts table. For newly introduced active and dirty Budget controls it adds new entries into the Budget Constraint Amounts table computing LHS and RHS entries for each newly introduced active and dirty Budget Control against each budget line. After all the computations on the Budget Constraint Amounts table are done all the dirty flags on selected Budget Controls are reset.

Please Note:

Budget Control can be active, dirty and applicable for a given Budget Line if it is active and dirty at any one of the following levels: System-wide, Budget Structure Level wide, Fund level or Line Level.

Process Implementation

Since this process needs to deal with huge volumes of production data in form of budget lines, this process is split into two sub-processes.

Scheduler Run

When the end-user schedules the “Calculate Budget Controls Amounts” batch job, then internally a Scheduler is started. The Scheduler gets all active Budget Structures in the system and spawns ‘N’ number of batch jobs where

N = active Budget Structures in the System.

Each spawned batch job is passed with the Budget Structure id that it is responsible to process. These spawned jobs are independent and execute concurrently, thus giving parallel processing capabilities.

This way the Scheduler distributes the processing workload to different spawned batch jobs that are capable of executing by themselves.

The Scheduler passes on the end-user provided batch parameter values to the spawned batch jobs. Each spawned job gets Budget lines for the assigned Budget Structure id and does the necessary processing depending on the Run modes. The Scheduler writes out an entry for each spawned job into the Budget Cleanup Status table. When each spawned job finishes processing successfully, it removes its associated entry from the Budget Cleanup Status table. If the run mode was ReCalculation then the spawned batch job last to finish will reset the dirty flags on Budget Controls.

Usability Notes

End-user schedules a single batch job of “Calculate Budget Controls Amounts” and this behaves as the Scheduler and spawns off multiple other jobs. The Scheduler and the spawned batch jobs can be viewed under the same Job Summary of the “Calculate Budget Controls Amounts” batch process. It is highly recommended that when an end-user schedules this batch process then the end-user should assign a distinct Job Name for each batch run. The Scheduler run is assigned this user-defined Job Name and the Job Name of all the spawned jobs generated by this Scheduler run is formulated as:

Spawned-Scheduler <Job Id of Scheduler> -Bud Stru <Assigned Budget Structure Id> -<Job Name of Scheduler Run>.

Note that maximum data capacity of the Job Name is 50 characters and the last piece of the generated Job Name for the spawned job can be truncated if the Job Name exceeds 50 characters.

For Example an end-user scheduled a “Calculate Budget Controls Amounts” batch process with Job Name as “Init1” then the Scheduler Run gets Job Name as “Init1” and all spawned jobs get Job name in the format as:

Spawned-Scheduler 11 -Bud Stru 1 –Init1.

Where 11 is the Job Id of Scheduler Run, 1 is the Budget Structure id this particular spawn job is dealing with and Init1 is the Job Name copied from the Job Name of the Scheduler.

This way if the end-user gives a Search with Job Name using wildcard characters as “*Init1”, the end-user can view the Scheduler job as well as all spawned jobs by this Scheduler.

Note that the Scheduler Run is the batch job run the end-user schedules online.

Requisites and Important Instructions for the Process

- Before this batch job can be run the system must be brought down and then brought up again. There should be no other Batch VLSs polling other than the ones meant to run this processes batch jobs. There should be no online or offline activities going on in the system. System here implies all VLSs hosting the application or Job Managers. The System would only be running this job and there should be no other activities going on.
- For Re-Calculation mode dirty flags for the Budget Controls should have been set before the system is brought down. Newly added Budget Controls should have the dirty flag checked before the system is brought down.
- When this batch job (Scheduler and all spawned jobs) is being executed, there should be no online or offline activity other than this batch job going on in the system.
- After the batch job (Scheduler and all spawned jobs) is completed successfully the system will have to be bounced again. This holds true even for scheduling another instance of this same job.
- Only a single instance of this job (Scheduler) can run at a time. There are edits that prevent multiple scheduling of this job.
- Spawned jobs do not have the value of App Server Id provided. These spawned jobs cannot be picked up by Batch VLSs that have the Process Assigned Jobs Only value as true. For achieving higher throughput an adequate number of Batch VLSs can be configured. These Batch VLSs should have the Process Assigned Jobs Only value as false. Value of Maximum Jobs Allowed for each Batch VLS must be configured optimally. For example, if there are 5 active Structures, each having an equal amount of Budget lines, and there were 5 Batch VLSs designed to run this job, then the optimal value that can be provided as Maximum Jobs Allowed for each Batch VLS will be 1. Suppose 2 was provided then it may happen that a single Batch VLS picked up 2 jobs simultaneously when there were other Batch VLSs that were available to pick and execute them.

Major Input

- All non-activity Budget Level/Allotment level inquiries
- Budget Controls Amounts table
- Budget Control (BUDCON) table
- Budget Fund Control (BUDFCON) table
- Budget Level Control (BUDLCON) table
- Budget Line Control table

Output

- Budget Controls Amounts table
- Budget Cleanup Status table

Parameters

Batch Parameters

Description (Caption)	Parameter Name	Default Value
Scheduler Run (Case insensitive values of Y - Yes, N - No)	Scheduler Run	Y
Run Mode (1- Initialization Mode, 2 - Recalculation Mode)	Run Mode	
Commit Block Size. If not entered then the default is 100. After processing the number of budget lines indicated for the Commit Block Size, a commit is issued. This parameter can be used for performance tuning.	Commit block Size	100
Select block Size. If not entered then defaulted to 1000. It is the number of Budget lines that can be fetched as a single block and stored to be processed. Can be used for Performance tuning.	Select block Size	1000
Budget Structure Id. For System Use Only. Value should not be supplied.	Budget Structure Id	

- Scheduler Run. This parameter value is required and is non-overrideable. The value should be always Y. The end-user basically can only schedule the Budget Controls Amounts batch job in Scheduler Run. It is then the Scheduler Run that spawns rest of the jobs.
- Run Mode. This parameter value is required. Valid run modes are 1 – Initialization Mode and 2- Recalculation mode.
- Commit block Size. This is a performance tuning parameter. The value can be tuned as per system configuration and its processing capabilities. If a value is not entered then value of 1000 is defaulted. This gives the number of Budget lines after processing which a commit is issued. The commit issued saves records in Budget Controls Amounts table. For each Budget line the number of records saved in Budget Controls Amounts table depends upon number of applicable Budget Controls against it. Too low value can increase the number of database commits and too high value can cause out of memory issues.
- Select block Size. This is a performance tuning parameter. The value can be tuned as per system configuration and its processing capabilities. It is the number of Budget lines fetched in a single database fetch query that is fired to obtain the budget lines to be processed. Too low value can increase the number of database fetches and too high value can cause out of memory issues.
- Budget Structure Id. This parameter is non-overrideable and is only meant of System Use. The Scheduler provides this value appropriately for the spawned jobs.

Sort Criteria

Budget lines fetched for each Budget Structure Level/ Allotment Inquiry table is sorted by the Unique Identifier.

Selection Criteria

The Scheduler job gets all active Budget Structures in the System.

For each Budget Structure, data from all non-audit Budget Levels Inquiry tables are selected for processing.

If the Budget level has an allotment, then data from all non-audit Allotment tables are selected for processing.

Problem Resolution

Each spawned batch job has checkpoint implemented. If the spawned jobs fail for certain errors, then the errors can be fixed and the spawned job can be restarted to start processing from the point where it left off. It depends on the types of errors encountered to see if restart of jobs makes sense.

2.2 Budgeting Report Processes

The Budgeting report run sheet included in this section is:

- Budget Roll Process (Modes 1, 2, and 3) – Report Mode

2.2.1 Budget Roll Process (Modes 1, 2, 3, and 5) – Report Mode

Description

The Budget Roll Process in report mode generates the "Budget Lines Selected for Roll" and the "Budget Lines Not Rolled" reports. The process can be used as a dry run to generate reports to view the budget lines selected and not selected for roll before actually running the Budget Roll Process (modes 1, 2, 3, & 5) in update mode. The "Budget Lines Selected for Roll" report displays concatenated keys of selected budget lines to roll and their budget name. The "Budget Lines Not Rolled" report displays concatenated keys of budget lines not selected for roll and their budget name. Budget lines are not selected for rolling in modes 1 or 2 even if they have met selection criteria if the budget line already exists in the target budget year. Budget lines are not selected for rolling or lapsing in modes 3 or 5 even if they have met selection criteria if the budget line had a negative amount or $0.00 in the source bucket.

When to Run

To be run before running the Budget Roll Process in modes 1, 2, 3, or 5 in update mode.

Major Input

- Budget level tables BUD_STRU_*_LVL_*

Other Input

- Parameter for Budget Roll Process BUD_ROLL_PARM
- Budget Structure GN_BUD_STRU
- COA Crosswalk R_COA_CROSSWALK

Output

- "Budget Lines Selected for Roll" report
- "Budget Lines Not Rolled" report

Parameters

Batch Parameters

Job	Parameter	Description	Default Values
Budget Roll	Budget Roll Parameter Id (BUD_ROLL_PARM_ID)	Budget Roll Parameter Id – Required Parameter	
	Client Name (CLIENT_NM)	Client name for Report – Optional Parameter	
	Process Id to pick COA Crosswalk records	Process Id to pick COA Crosswalk records – Optional Parameter	

	(XWALK_PROC_ID)		

Sort Sequence

- COA element COA element as specified in Transaction break in BUD_ROLL_PARM
- Unique Identifier UNID

Selection Criteria

- Select criteria for obtaining budget lines from the Budget Inquiry Table BUD_STRU_x_LVL_y (x – Structure id from Parameter Id record, y – lowest required level for Structure) is:
  - Budget Fiscal Year = Source BFY on Parameter Id record
  - Fund = Selected Funds if provided on Parameter Id record
  - Department = Selected Departments if provided on Parameter Id record
  - Appropriation = Selected Appropriations if provided on Parameter Id record
  - If Bypass Deactivated Lines is checked for the Parameter Id record, then if the Active Flag for a budget line is not = true, that budget line is not copied
  - If Bypass Unused Lines is checked on Parameter Id record then if all of the expense and/or revenue basic bucket amounts are zero, then that budget line is not copied.
- Order by
  - The Transaction Break for the Parameter Id record determines if budget lines will be grouped by all lines for a Fund, Appropriation, or Department. The other choice is one budget line per budget transaction. If this is selected, budget lines are listed in the order found in the source year.
  - Unid – Unique Identifier

Troubleshooting

No database restore is required. Correct the problem and rerun the job executing the program. No restoration of datasets or files from backups is required for this program.