CA Release Automation 5.x / 6.x

Microsoft SQL Server

Database Care and Feeding Guide

Author : Keith Puzey

Version: 2.00

Filename: CA Release Automation Database Care and Feeding Guide- SQL.docx

Date: Monday, 25 April 2016

Table of Contents

Overview

This document has been written to be used by a Database DBA to purge data from the CA Release Automation database. The tables being purged are the tables containing event information and temporary auditing information. The document includes a high level overview of the Database schema for information.

Database Maintenance Guidelines

This document has been created as a Best Practice guide for the recommended maintenance tasks of the CA Release Automation (Nolio) Database. It is recommended for the DBA to read the References listed below:

- Develop a backup and recovery plan. Regular backups are critical to the safety of the database*. Recovery exercise should be performed at least once a year.
- Set auto growth sizes of data file and transaction log file to appropriate values. Ensure that ample disk space is available for the files to grow.
- The query optimizer relies on statistical information about the data so it is crucial that table and index statistics are collected and up-to-date. Ensure database Auto Create Statistics and Auto Update Statistics options are set true (the default).
- Evaluate and install security patches regularly.
- Monitor SQL Server log the event log. Check for problematic errors or warnings.
- Use monitor tools to check database performance, availability, etc.
- Check Database Integrity regularly. This can be done using SQL Server Maintenance Plan Wizard or DBCC CHECKDB.
- Reorganize or rebuild fragmented indexes*. See http://technet.microsoft.com/en-us/library/ms189858.aspx.and http://technet.microsoft.com/en-us/library/ms179349(v=sql.105).aspx

*Can be done using SQL Server Maintenance Plan Wizard in Management Studio.

Data Purging and Maintenance Guidelines

Overview

The scripts included in supplied zip file provide a mechanism to purge data from the Release Automation database. The two stored procedures are provided for purging data in the Release Automation Database:

The first script is used for purging offline execution Jobs

The second script can be used to purge audit records.

It is recommended that the stored procedures be executed during off hours though a scheduled job. The two stored procedures can be executed concurrently in different database sessions. However, do not execute the same stored procedure more than once simultaneously.

Important

Due of the large amount of data the purge procedures may be deleting, the database transaction log may grow exponentially. Please ensure there is adequate free disk space for the transaction log to grow and the transaction log is being backed up properly. For more information, please reference http://technet.microsoft.com/en-us/library/ms189085(v=SQL.105).aspx

Due to the large amount of data that could be purged the database indexes could require re indexing or reorganizing to rebuild fragmented indexes - see http://technet.microsoft.com/en-us/library/ms189858.aspx and http://technet.microsoft.com/en-us/library/ms179349(v=sql.105).aspx also the database should be regularly checked to ensure that the database integrity is consistent. This can be done using SQL Server Maintenance Plan Wizard or DBCC CHECKDB.

Stored Procedure Action	Script Name	Description
Purging Offline Execution Jobs	sp_purge_execution_jobs - sqlserver.sql	Create stored procedure sp_purge_execution_jobs for purging offline execution jobs. Create an index on offline_execution_jobs if it does not exist.
Purging Audit Records	sp_purge_audit - sqlserver.sql	Create stored procedure sp_purge_audit for purging audit records. Create indexes on audit tables if they do not already exist.
Creating Job for Purging Offline Execution Jobs	create_job_for_purging_execution_jobs - sqlserver.sql	Sample script to create a job for purging offline execution jobs every night for 1 hour.

Purging Offline Execution Jobs

The script “sp_purge_execution_jobs” can be used to create a stored procedure for purging offline execution jobs and its associated records one day at a time starting from the oldest records.

The following tables are purged:

offline_distribution_events
offline_exec_param_links
offline_exec_servers
offline_execution_jobs
offline_flow_events
offline_manual_operations
offline_parameter_requests
offline_parameters
offline_propagation_events
offline_step_events
offline_steps
offline_user_events

Parameters:

retention_days: Number of days to retain execution job records. Default is 90 days

time_limit: Number of seconds after which no command is executed. The procedure will stop when the time limit is reached and the last command (started before the time limit) is complete. The default is no time limit (null).

time_delay: Number of seconds to delay between each delete command. The default is null, no delay.

delete_chunk_size: Maximum number of rows to delete per command. The default is 10000 rows.Example:

The following example purges jobs older than 120 days, with a time limit so that no command is executed after 3600 seconds (1 hour)

```
exec sp_purge_execution_jobs @retention_days=120, @time_limit=3600
```

Note: When using this purge procedure on large databases the number of days purged within the time period will vary depending on the amount of events, as an example this stored procedure tested on a large customer database purged 5 days in one hour.

Purging Audit Records

The script “sp_purge_audit” can be used to create a stored procedure for purging audit records when designrevisionentity .processStatus is not null.

The following tables are affected by the purging script:

auditingentry
revision_auditingentry
All table with _aud suffix

The following tables are excluded:

health_record_aud
health_round_samples_aud
health_sample_aud
health_threshold_aud

Parameters:

time_limit:	Number of seconds after which no command is executed. The procedure will stop when the time limit is reached and the last command (started before the time limit) is complete. The default is no time limit (null).
time_delay:	Number of seconds to delay between each delete command. The default is null, no delay.
delete_chunk_size:	Maximum number of rows to delete per command. The default is 10000 rows.

Example:

The following example purges audit records with a time limit so that no command is executed after 3600 seconds (1 hour)

```
exec sp_purge_audit @time_limit=3600
```

Creating Job for Purging Offline Execution Jobs

The Sql Script "*create_job_for_purging_execution_jobs - sqlserver.sql*" contains a sample script to create a SQL Server job for purging offline execution jobs. The stored procedure sp_purge_execution_jobs (in sp_purge_execution_jobs - sqlserver.sql) must be created before the job starts. For information on creating job and its limitations and security, please reference

http://technet.microsoft.com/en-us/library/ms190268.aspx#Security

This sample script a job to purge execution jobs older than 120 days for one hour every midnight. Some options can be changed if needed.

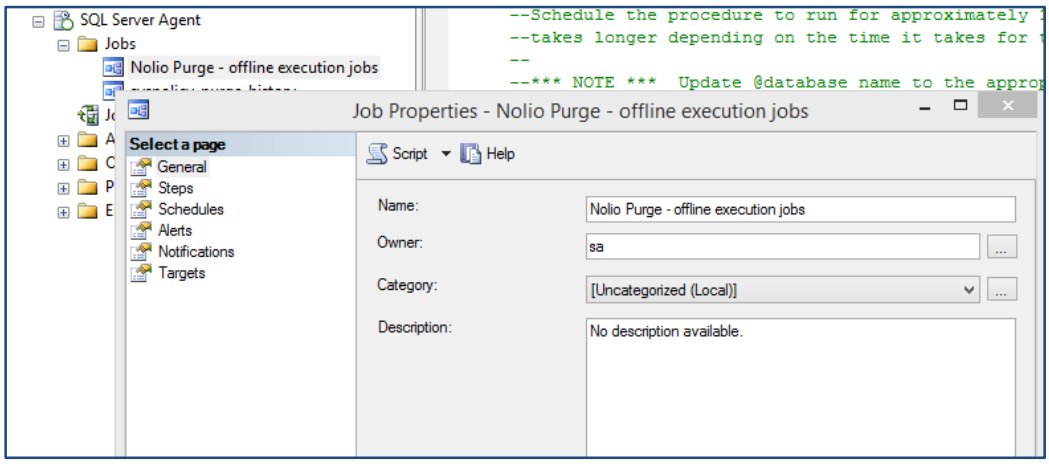
On line 29, update retention_days and time_limit in seconds

```
@command=N'sp_purge_execution_jobs @retention_days=120,
@time_limit=3600'
```

On line 30, update the nolio database name

```
@database_name=N'nolio_db'
```

The job should be edited in SQL Server Management Studio to select the target database and RunAS account. Please refer to the link above for more information.



CA Release Automation Database Schema

CA Release Automation table overview

- All tables with “rc_” prefix belong to ROC entities
- All tables with “_aud” postfix are auditing tables that are used to track changes done in the corresponding entities (server -> servers_aud)
- All other tables are ASAP studio, management console and general system tables.

Schema Overview:

The following table has the high level details for the Release Automation Database and the following section contains more details for some of the tables.

Table Type	Table Name	Description
ACL tables	Tables with Prefix ACL_	Acl table are used for security (spring security framework)
Auditing Tables	Tables with Suffix _aud	Auditing tables are used for reporting all changes in design process
Parameter Tree Tables	static_ parameter _folder	Represents all the built-in folders
	basic_parameter	Represents the parameters
	child_param_folders	Represents the folders nested under this folder
	basic_parameter	An abstract table and all relations is a concrete parameters
Action Tables	action_class_info	The action’s java class name
Values tables	Composite_value	A parameter value can contain values and parameters
Actions and flows under component	executable	Action and flow details
	pre_ executable & post_executable	Used by action loops
	order_link	Links between actions
Monitor tables	All tables with prefix health_	Used for health monitor in the ROC
High availability tables	nac_nodes	Name of NAC nodes
	master_nac	Name of Master NAC and last heartbeat timestamp
Notification tables	All tables with prefix - “notification_”.	Used for notification in ASAP
Offline execution tables	All tables with prefix- “offline_”.	Used for reporting about execution processes
Event Tables	flow_events	All events related to execution process
	distribution_events	Events related to distribution
Artifacts Tables	All tables with prefix rc_artifact	Entry of artifacts in POC

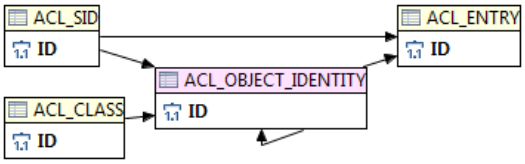
Table Type	Table Name	Description
Release Structure Tables	rc_releases	Contains all releases
	rc_stages	Release have two stages: Init stage Run Stage
	rc_modules	Contains steps in release
Release Parameter Tables	All tables with a prefix rc_param	Release parameters
Template Tables	All tables with a prefix rc_template	Templates in ROC
Servers Tables	servers server_ips server_type_instance access_by_jxta	Represent server and NES machines
Execution Tables	All tables with Prefix "execution"	All data for current execution jobs (After job is done the data is removed to the offline tables.)
Schedule Tables	All tables with prefix "schedule_"	ASAP Scheduling processes
Calendar Tables	All tables with prefix "rc_scheduled"	Calendar in ROC

Note : Some of the DB entities have been renamed in the UI since its development. For example:

- Step -> Action
- Module -> Release Step

ACL table details

Acl table are used for security (spring security framework)



Acl classes are system entities that require permission to access:

EnvironmentDetails, ProcessToEnvironmentAssignment, Application , TemplateOnEnvironmentProjection , EnvModuleTemplate, Module, Release , Template , TemplateModule.

Each acl entry represents a user or a group with permission to a specific entity.

Auditing table details

Auditing tables are used for reporting all changes in design process which Include:

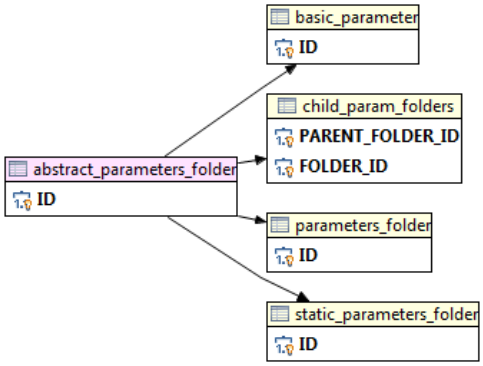
- all tables with suffix "_aud" .
- auditingentry table.
- Auditreportentry table.

The data in the "_aud" tables is generated immediately after changes done to the corresponding entity. The internal auditing service runs a scheduled task to process and transfer the data from these "aud" tables to the main audit report table – "auditreportentry"

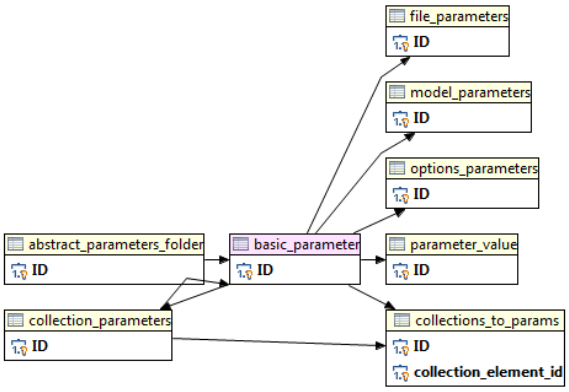
Parameter tree table details

Parameter tree – represent the parameter tree in ASAP.

- "static_ parameter _folder" - represents all the built-in folders.
- "basic_parameter" - represents the parameter itself.
- "child_param_folders" - represents the folders nested under this folder.



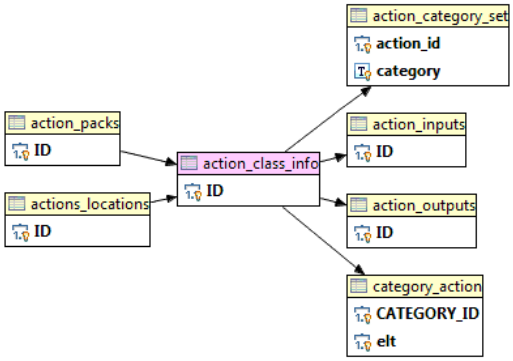
"basic_parameter" - is an abstract table and all relations is a concrete parameters.



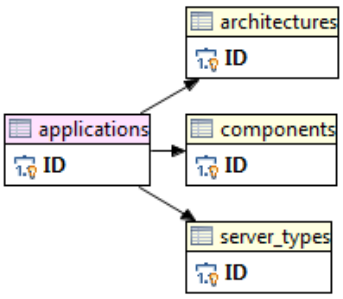
Action Table details

These tables hold the metadata of the all the actions loaded to the system. The actions themselves are defined in java classes, with special annotations (Action name and description, inputs and outputs etc...) These tables are populated when the action packs are loaded to the system.

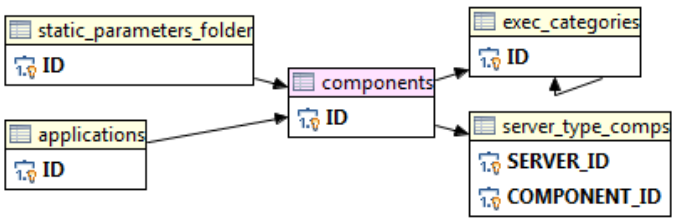
"action_class_info" –The action's java class name.



Process design structure details



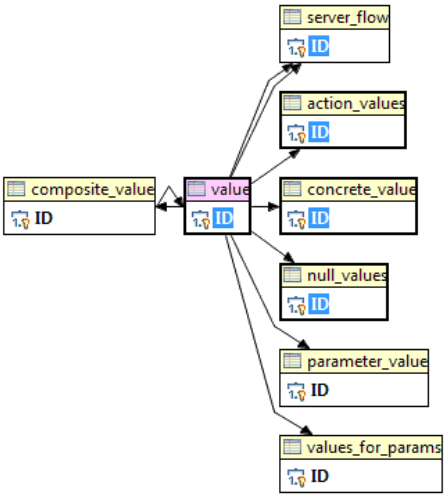
The tree is exactly like an ASAP



Components can be assigned to "server_types"

Values table details

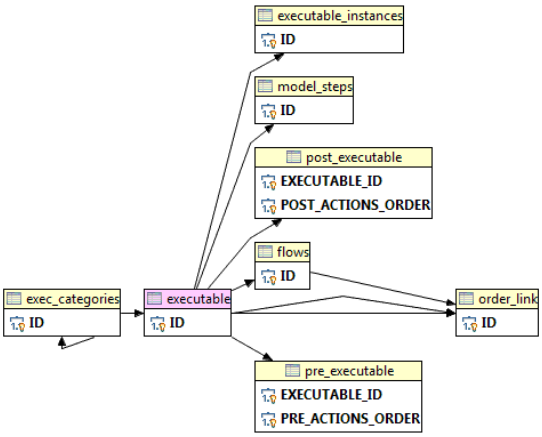
Represent type and concrete value in the parameters.



"composite_value" - is a parameter value can contain values and parameters.

Actions and flows under component details

Table Name	Description
Executable	Action and flow
pre_ executable & post_executable	Used by action loops.
order_link	Links between actions.



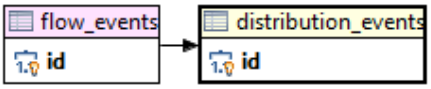
High availability table details

Indicates which server is active/passive.

Event table details

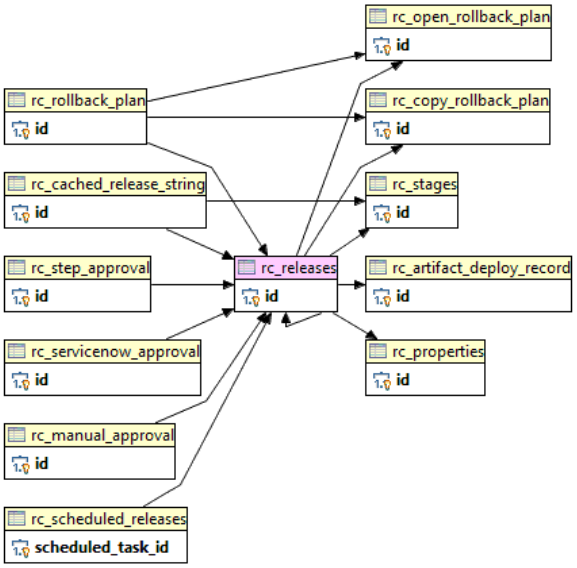
Tables used to manage the events in the application.

Table Name	Description
Flow_events	All events related to execution process.
distribution_event	Events related to distribution.



Release structure table details

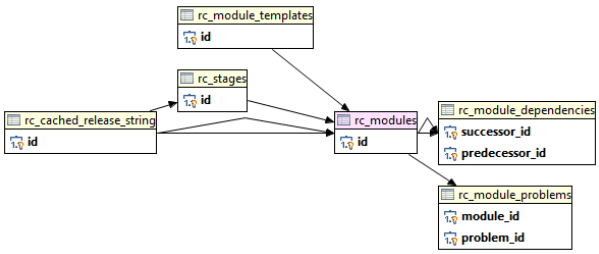
Represent structure of release in ROC



“rc_releases” - contain all releases

“rc_stages” – all releases have some stages. Currently there are two:

- Init stage - for init step.
- Run stage – for running steps.

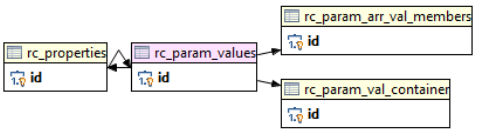


“rc_modules” – contains steps in release.

“rc_modules_template” – is an abstract for “rc_modules”.

Release parameter table details

Represent release parameters.



Servers table details

Represent agent and NES machines