Genesys Intelligent Automation Reference Guide

Hive Off Recovery Process for SQL Server

4/30/2025

Contents

Hive Off Recovery Process for SQL Server

Process Summary

The purpose of this memo is to give background information on the Genesys Intelligent Automation ‘hive off’ process and to provide instruction as to the most efficient way to recover from a failure in the hive off process in an SQL Server environment.

Due to the large amount of reporting data captured and processed by the Genesys Intelligent Automation application it is imperative that, should a failure in the hive off process occur, the system is quickly brought back into a state whereby the next night’s hive off can run successfully.

> **Important**
>
> If corrective action is not taken within several hours of the original failure, then data loss is likely to result.

Definitions and Background

Term	Definition
Hive off	The nightly batch job that runs on the primary Genesys Intelligent Automation GUI server in order to manage the large amounts of reporting data that is captured during the day. The hive off job typically runs in the early hours of each morning and can take several hours to complete.
Daily tables	The DB tables that receive details about incoming calls and Control Centre auditing for the current day. These tables are constantly receiving new reporting data from the Genesys Intelligent Automation servers: • calls • call_sites • call_steps • business_tasks • gui_actions
Historical tables	Read-only data for previous days, that has been

Term	Definition
	moved from the ‘daily’ tables to their ‘historical’ equivalents as part of the hive off: • historical_calls • historical_call_sites • historical_call_steps • historical_business_tasks • historical_gui_actions
Aggregate tables	Read-only data for previous days, that has been moved from the ‘daily’ tables as part of the hive off process and report generation: • daily_dashboard_aggregates • daily_health_aggregates • daily_health_aggregates_start • last31days_slots • reports_blockresults_blockResults_aggregate • reports_businessTasksSummary_businessTasksSummary_aggre • reports_callJourneys_aggregate • reports_callsbyday_callsByDate_aggregate • reports_callsbyHour_callsByHour_aggregate • reports_recognitionsummary_inputBlockSummary_aggregate • reports_summary_businesstasks1_aggregate • reports_summary_businesstasks2_aggregate • reports_summary_callsummary_aggregate • reports_summary_lastmenu_aggregate • reports_summary_recognitionsummary1_aggregate

The Genesys Intelligent Automation hive off process consists of four distinct steps:

1. Calculating aggregates and copying records from the ‘daily’ tables into the ‘historical’ tables;
2. Disabling indexes on the ‘daily’ tables and deleting those records from the ‘daily’ tables that were copied to the ‘historical’ tables in step 1;
3. Deleting any expired data from the ‘historical’ tables as per the **NumDaysHistoricalDataToKeep** setting;
4. Re-enabling indexes on the ‘daily’ tables.

Should the hive off process fail for any reason, the corrective actions to ensure that the next night’s hive off runs successfully will be different depending on which step the hive off failed.

Determining at Which Step the Hive Off Failed

This can be best determined by looking at the fish.log file (and, perhaps, fish.log.1, fish.log.2, etc.) in the primary GUI server's *SpeechStorm\Platform\TomcatGUI\logs* directory.

The names of the steps in the log file appear as:

1. populateAndHiveOff
2. deleteBatchAfterHiveOff
3. deleteOldHistoricalData
4. rebuildIndexes

The following entries should be written to the logs during each hive off run, and the presence or absence of certain of the entries below will indicate which step of the hive off failed, thus allowing you to follow the correct path in the Recovery Process in the next section.

Start of step #1:

```
[INFO ] 2016-06-15 00:15:00,833 (Timer-1::)
com.speechstorm.fish.reporting.PopulateAggregatesAndHiveOffHistoricalDataTask.populateAggregates
```
starting at ...

The following entry indicates that step #1 has finished and step #2 is about to begin:

```
[INFO ] 2016-06-15 00:37:48,557 (Timer-1::)
com.speechstorm.fish.reporting.PopulateAggregatesAndHiveOffHistoricalDataTask.populateAggregates
```
finished populateAndHiveOff at ...

This entry indicates that step #2 has finished and step #3 is about to begin:

```
[INFO ] 2016-06-15 01:26:43,590 (Timer-1::)
com.speechstorm.fish.reporting.PopulateAggregatesAndHiveOffHistoricalDataTask.populateAggregates
```
finished all deleteBatchAfterHiveOff steps at ...

This entry indicates that step #3 has finished and step #4 is about to begin:

```
[INFO ] 2016-06-15 02:06:24,271 (Timer-1::)
com.speechstorm.fish.reporting.PopulateAggregatesAndHiveOffHistoricalDataTask.deleteOldHistorica
```
finished at ...

This entry indicates that all steps have completed:

```
[INFO ] 2016-06-15 02:06:45,571 (Timer-1::)
```

com.speechstorm.fish.reporting.PopulateAggregatesAndHiveOffHistoricalDataTask.rebuildIndexes()
finished rebuildIndexes at ......

Recovery Process

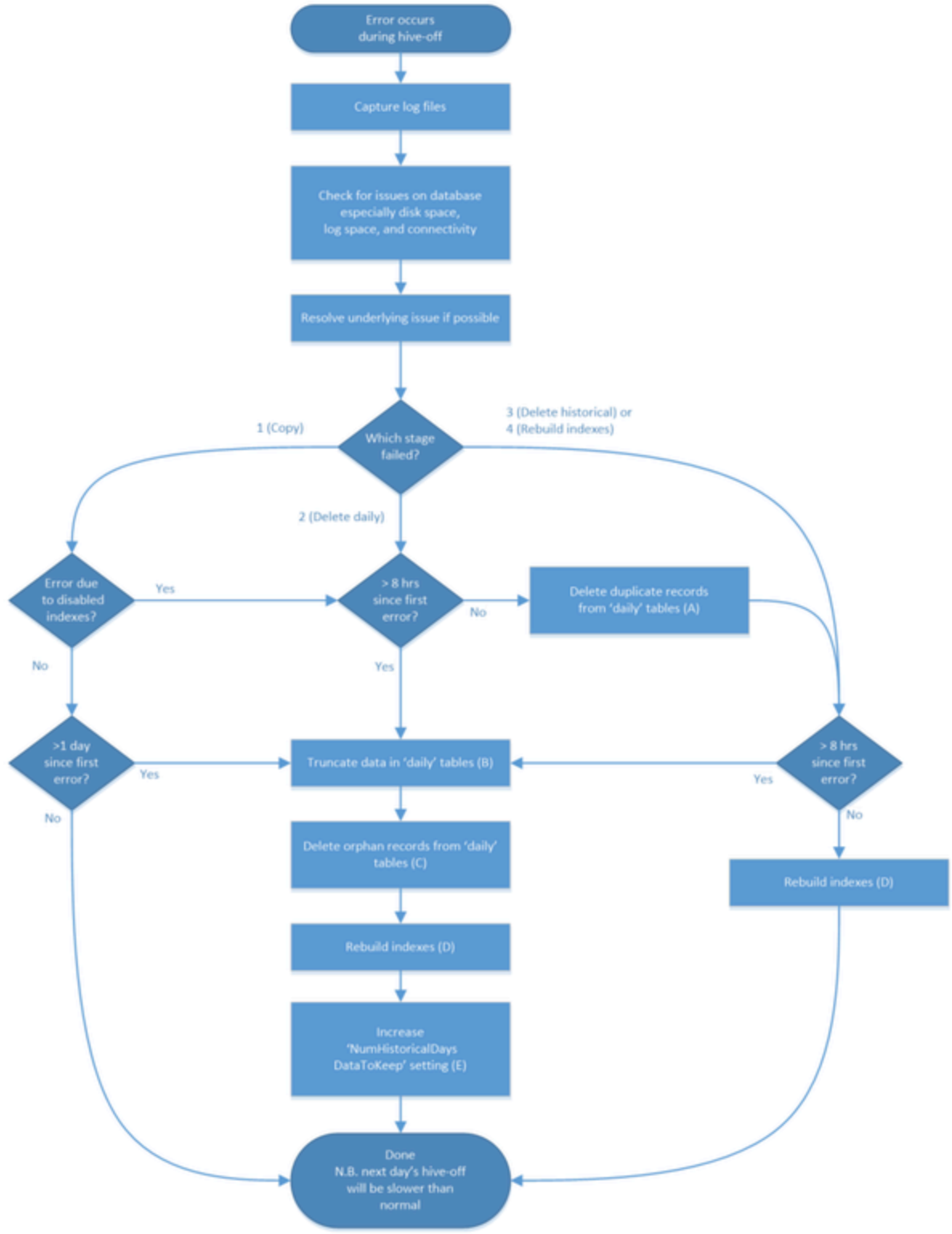
While it is important to try to understand and fix the problem that caused the hive off to fail, if this cannot be done immediately (for example, it requires more investigation) then it is better to take action immediately to ensure that the next night's hive off still runs.

There is a short window of time each night in which to run the hive off (for example, between midnight and 7am) and if the hive off does not run successfully for several days in a row then the amount of data in the 'daily' tables will become prohibitive and will have an impact on both call handling and reporting; the only practical solution should this be allowed to happen is to do a bulk delete of the 'daily' tables, **thus causing reporting data to be lost**.

The main factors to take into account are:

- There must be no duplicate records in the 'daily' and 'historical' tables when the hive off runs;
- The indexes on the 'daily' tables need to be enabled before the hive off runs;
- Re-enabling indexes on the 'daily' tables when there is a lot of data in those tables will lock the tables for a significant period of time (for example, minutes or even hours) which could adversely affect call handling or cause a large backlog of reporting data waiting to be written;

If the hive off has not run for several days and the value of the **DBOvernightJobs.NumDaysHistoricalDataToKeep** setting is not increased accordingly then the hive off will attempt to delete a much larger chunk of records from the 'historical' tables, which will take proportionally longer to execute and could lead to running out of transaction log space on the DB server. The overall hive off recovery process is described in the following diagram:


Error occurs during hive-off
Capture log files
Check for issues on database especially disk space, log space, and connectivity
Resolve underlying issue if possible
Which stage failed?
1 (Copy)
2 (Delete daily)
3 (Delete historical) or 4 (Rebuild indexes)
Error due to disabled indexes?
Yes
No
> 8 hrs since first error?
No
Yes
Delete duplicate records from 'daily' tables (A)
>1 day since first error?
Yes
No
Truncate data in 'daily' tables (B)
> 8 hrs since first error?
Yes
No
Delete orphan records from 'daily' tables (C)
Rebuild indexes (D)
Rebuild indexes (D)
Increase 'NumHistoricalDays DataToKeep' setting (E)
Done
N.B. next day's hive-off will be slower than normal

Corrective Actions in Detail

One or more of the following actions may be required according to the diagram above.

(A) Delete Duplicate Records From ‘Daily’ Tables

This action is required if the hive off stopped during step 2. There will be no loss of reporting data when you execute this step, as the data has already been copied to the ‘historical’ tables. The SQL to be run against the ‘fishreports’ database is as follows:

```
set ROWCOUNT 20000;

delete from call_steps where id in (select id from historical_call_steps);

set ROWCOUNT 20000;

delete from call_sites where id in (select id from historical_call_sites);

set ROWCOUNT 20000;

delete from business_tasks where id in (select id from historical_business_tasks);

set ROWCOUNT 20000;

delete from calls where call_id in (select call_id from historical_calls);

set ROWCOUNT 20000;

delete from gui_actions where id in (select id from historical_gui_actions);
```

> **Important**
>
> This SQL should be run multiple times until there are no more duplicate records reported.

The max **ROWCOUNT** limit can be increased with caution, but care needs to be taken that this does not have an adverse effect on table locking; monitoring is key here.

(B) Truncate Data in ‘Daily’ Tables

This action is required if the amount of data in the ‘daily’ tables is such that re-enabling the indexes would lock the tables for a prohibitive amount of time.

Be aware that this step will result in a loss of all reporting data in the ‘daily’ tables, although reporting data in the ‘historical’ tables will not be affected. We recommend taking a backup of the database just before running it in order to preserve the data in the ‘daily’ tables for further analysis of the root cause.

The SQL to run against the ‘fishreports’ database is as follows:

```
truncate table calls;

truncate table call_sites;

truncate table call_steps;

truncate table business_tasks;

truncate table gui_actions;

declare @maxid bigint;

select @maxid = max(call_id) + 2000 from historical_calls;

dbcc checkident (calls, RESEED, @maxid);

select @maxid = max(id) + 2000 from historical_call_sites;

dbcc checkident (call_sites, reseed, @maxid);

select @maxid = max(id) + 2000 from historical_call_steps;

dbcc checkident (call_steps, reseed, @maxid);

select @maxid = max(id) + 2000 from historical_business_tasks;

dbcc checkident (business_tasks, reseed, @maxid);

select @maxid = max(id) + 2000 from historical_gui_actions;

dbcc checkident (gui_actions, reseed, @maxid);

delete from calls where call_id < 2000;

delete from call_sites where id < 2000;

delete from call_steps where id < 2000;

delete from business_tasks where id < 2000

delete from gui_actions where id < 2000;
```

(C) Delete Orphan Records from ‘Daily’ Tables

This action is required once the ‘daily’ tables have been truncated, to ensure that there are no child records (for example, call_steps) whose parent records (i.e. calls) have been truncated. This situation can occur due to the timing of the truncate statements. It will result in the loss of a minute amount of reporting data, and can be viewed as the completion of the ‘truncate’ operation above.

There should be a one-minute pause between executing action (B) and this action. The SQL to run against the ‘fishreports’ database is as follows:

```
delete from call_sites where call_id not in (select call_id from calls);
```

```
delete from call_steps where call_id not in (select call_id from calls);

delete from business_tasks where call_id not in (select call_id from calls);
```

(D) Rebuild Indexes

This action is required any time the hive off fails after step 1, in order to ensure that the indexes on the 'daily' tables are enabled and working for the next night's hive off. The SQL to run against the 'fishreports' database is as follows:

```
exec USP_RebuildIndexes
```

This could be time-intensive, depending on the amount of data that is in the 'daily' tables at the time that the command is issued.

(E) Increase NumDaysHistoricalDataToKeep Setting

Run the following SQL against the 'fishreports' database to determine what value should be used for the **NumDaysHistoricalDataToKeep** setting in order to only hive off a maximum of 2 days' old historical data at once:

```
declare @mindate DATE

select @mindate = min(call_start_date) from historical_calls

select @mindate, datediff(d, @mindate, GETDATE()) - 1 as NewDaysToKeepSettingValue
```

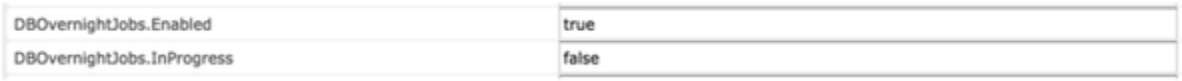
You need only change the value of the setting if the result of the above SQL is greater than the original/current value of the setting. The setting can be changed via the Genesys Intelligent Automation Control Centre, in the **Administration** -> **Default Server Settings** page:

The value should be decreased each day from this point forward until it reaches its original value. In this way each night's hive off will involve deleting a maximum of 2 days old historical data.

Next Procedure

1.Ensuring That the Next Hive Off Will Run

Ensure that the hive off job is enabled and that the **in progress** flag is set to *false*. These settings can be changed via the Genesys Intelligent Automation Control Centre, in the **Administration** -> **Default Server Settings** page:



2.(Optional) Manually Re-Running the Hive-Off

In high-call traffic scenarios, it is best not to run the hive-off during the day but instead to wait for it to run automatically the following night.

If the recovery steps have been undertaken shortly after a failure, or in a low-call traffic scenario, however, it is possible to kick off the hive-off process manually. To do this, change the following setting (in the same location as above) from 1 to 0:

Now restart the primary GUI server and the hive-off will begin immediately, even though the current time is after the schedule hour and minute that are defined in the server settings. When the hive-off has finished, be sure to change this setting back to 1 so that any future restarts of the GUI server will not trigger an immediate hive-off. (It is not required to restart the GUI again at this point.)

3.Identifying the Root Cause

If the underlying cause for the failure has not yet been identified and resolved, then you should take steps to record more detailed log events in order to better troubleshoot if the issue recurs. On the database server, your DBAs will be able to advise on the best additional logging setup. On the Genesys Intelligent Automation primary GUI server, you should enable full SQL debug logs by editing the `SpeechStorm\Platform\TomcatGUI\webapps\fish-gui\WEB-INF\classes\log4j.properties` file and ensure that the following line is not commented out (i.e. ensure that it does not begin with a '#' character):

```
log4j.logger.java.sql=DEBUG
```

The setting should be picked up automatically without requiring a restart.

Enabling full SQL debug logs will create a lot of log files on the primary GUI server, so care must be taken to perform a daily archive of these logs stretching back to at least 3 months.

4.Managing the Ongoing Deletion Of Old Historical Records

Remember to decrease 'NumDaysHistoricalDataToKeep' by 1 each day if it was increased as per action (E) above. This should be done daily until the setting is back to its original value.