Framework Deployment Guide

Monitoring Health of Configuration Environment

4/25/2025

Contents

Monitoring Health of Configuration Environment

The health of the Configuration Database requires that you verify several key tables, and identify any typical issues that might lead to inconsistent data. The health monitoring scripts discussed in this section enable you to do this. In most cases, you can use the reports created by these scripts to create new scripts that will correct the problems.

All health monitoring scripts support Configuration Server Database Schema 8.1 or newer.

Running the Health Monitoring Scripts

The scripts can be executed using DBMS utilities or from the command line.

> ### Important
>
> Genesys recommends that you run the **search_orphan_flex_prop.sql** script from the command line. When this script is run using DBMS utilities, the output is inconsistent and does not always provide the information requested.

Using DBMS Utilities

Run the script using the corresponding DBMS utility, as follows:

- MSSQL scripts: Microsoft SQL Server Management Studio 2005 or newer
- Oracle scripts: Oracle SQL Developer 1.5 or newer
- PostgreSQL scripts: pgAdmin III 1.12 that supports Postgre 9.0 or newer

For all scripts except **search_orphan_flex_prop.sql**, the script output appears on the console, which you can capture and save as a **.csv** file, suitable for further analysis. If the output contains no results, your database does not contain any records with the specific problem.

From the Command Line

The Configuration Database Maintenance Scripts IP also contains command-line scripts for Windows (MS SQL, Oracle, and Postgre databases) and Linux (Oracle and Postgre databases) operating systems. Each command-line script runs one of the SQL scripts, and writes the output to a **.csv** file in a specified results directory. This directory must exist before running the script.

Each command-line script generates one **.csv** report file, called **csv_<health_script_name>_<DBMS_TYPE>.csv**, where

- `<health_script_name>` is the name of the script being run
- `<DBMS_TYPE>` is the name of the DBMS to which the script is to be applied

The location of the command-line scripts are listed in the following table.

DBMS	Operating System	
	Windows	**Linux**
Windows MS SQL Server	\mssql\ search_duplicates_flex_prop_MSSQL_DB.bat \mssql\search_orphan_flex_MSSQL_DB.bat \mssql\ check_cfg_max_dbid_MSSQL_DB.bat	Not available
Oracle	\oracle\ search_duplicates_flex_prop_ORACLE_DB.bat \oracle\ search_orphan_flex_ORACLE_DB.bat \oracle\ check_cfg_max_dbid_ORACLE_DB.bat	/oracle/ search_duplicates_flex_prop_ORACLE_DB.sh /oracle/ search_orphan_flex_ORACLE_DB.sh /oracle/ check_cfg_max_dbid_ORACLE_DB.sh
Postgre [a]	\postgre\ search_backslash_POSTGRE_DB.bat \postgre\ search_duplicates_flex_prop_POSTGRE_DB.bat \postgre\ search_orphan_flex_POSTGRE_DB.bat \postgre\ check_cfg_max_dbid_POSTGRE_DB.bat	/postgre/ search_backslash_POSTGRE_DB.sh /postgre/ search_duplicates_flex_prop_POSTGRE_DB.sh /postgre/ search_orphan_flex_POSTGRE_DB.sh /postgre/ check_cfg_max_dbid_POSTGRE_DB.sh
[a] For Postgre scripts, configure the PostgreSQL server to allow a client connection with the host on which the scripts are run. For more information, see https://www.postgresql.org/docs/9.3/static/auth-pg-hba-conf.html.		

Command Syntax

Run the script from the folder in which it is located, using the following syntax:

```
<script> <USER> < PASSWORD> [<HOST> <PORT> or <HOST[\INSTANCE]>] [<SERVICE_NAME or DB_NAME>] <CSV _RESULT_DIR>.
```

where:

- `<script>` is the name of the script, as provided in the table above.
- `<USER>` is the name of a user authorized to access the database.
- `<PASSWORD>` is the password required for the authorized user to access the database.
- `<HOST>` is the name of the host on which the database is located.
- `<PORT>` is the number of the port on that host that is used by the DBMS.
- `<HOST\INSTANCE>` is the database host and instance, if not the default.
- `<SERVICE_NAME>` is the Oracle Instance Service Name, that is, the SERVICE_NAME value in **TNSNAMES.ORA**.
- `<DB_NAME>` is the name of the Database.
- `<CSV_RESULT_DIR>` is the path to the directory in which the output **.csv** files will be stored.

> ## Tip
> To see the syntax on-line, enter the script name with no parameters and press ENTER.

Examples

MS SQL Server

```
.\search_duplicates_flex_prop_MSSQL_DB.bat sa password localhost\SQLEXPRESS TestDB D:\MF_mon\results\mssql

.\search_duplicates_flex_prop_MSSQL_DB.bat sa password somehost SomeTestDB D:\MF_mon\results\mssql
```

Oracle

```
./search_orphan_flex_ORACLE_DB.sh sourcedb sourcedb localhost 1521 orcl.genesys.com '/home/genesys/results/oracle'
```

Postgre

```
./search_backslash_POSTGRE_DB.sh postgres password localhost 5432 TestDB '/home/genesys/results/postgre'
```

csv_search_backslash.sql

This script runs only to PostgreSQL databases, and retrieves a list of configuration options, including those set in the Annex, for which the configured value contains two or more sequential backslash characters (\\). This script creates a drill-down report that enables you to determine if any particular object of a given type is configured with an option or Annex value of two or more sequential backslash characters.

When using a PostgreSQL database, if the PostgreSQL database setting **standard_conforming_strings** is not set correctly, any database entries containing backslashes have those backslashes doubled, and data might be lost as a result. Typically, this problem affects Application options and/or flexible properties. This script helps you identify if **standard_conforming_strings** is set incorrectly, and what records it affects.

If the number of records in the report increases significantly immediately after a Configuration Server restart, you can also use this report to determine if the sudden increase is caused by the incorrect setting of **standard_conforming_strings**.

If there are no records in the report, your database does not contain any records with the specific problem.

This script produces CSV-formatted output on the console. There is one header row, and any number of data rows. The number of data rows is variable and can be large. The data rows are grouped by type and then by parent object DBID within each type. The header row identifies each column in the data rows, as follows:

- **type:** The type of object, as visible in Genesys Administrator. The one exception is that a **User** or **Agent** object type is named **Person**.

- `object_name:` The name of the parent object.

- `tenant_name:` The name of the Tenant object to which this object belongs. This information is not provided for Folder object types.

- `object_dbid:` The DBID of the object, enabling a trace of the object in the database and the audit trail.

- `option_type:` The type of configuration option, based on where it is set, **Option** or **Annex**.

- `section_name:` The name of the configuration option section in which the the option is set.

- `option_name:` The name of the option with a value containing two or more sequential backslashes.

- `option_value:` The option value that contains two or more sequential backslashes.

Note that if the value of an option contains one or more comma (,) characters, the `option_value` string in the CSV output might be formatted incorrectly or truncated.

Sample printout, in CSV format:

```
type,object_name,tenant_name,object_dbid,option_type,section_name,option_name,option_value
Application,confserv,Environment,99,Option,backslashOPt,backlashopt,^[\(\)\-\.\+\d\s\*#]*[\d]+[\(\)\-\.\+\d\s\*#/]*$
Application,confserv,Environment,99,Annex,backslashAnnex,backlashopt,^[\(\)\-\.\+\d\s\*#]*[\d]+[\(\)\-\.\+\d\s\*#/]*$
```

Supported DBMS

- Oracle 11g and newer, can be launched using Oracle SQL Developer 1.5 or newer
- MSSQL 2005 and newer
- PostgreSQL 9.0 and newer

csv_search_duplicates_flex_prop.sql

This script retrieves, for all object types, a list of Annex records that have the same name and parent object, but different DBIDs. You can use this drill-down report to determine if any particular objects of a given type have two or more duplicate Annex records. When duplicate Annex records exist, the record with the lowest DBID is masked by duplicate records with higher DBIDs. When the Annex option with the highest DBID is deleted by Configuration Server, the masked option with the next highest DBID becomes active in the Annex after Configuration Server is restarted. If the whole section is deleted, all of the masked duplicate Annex records with lower DBIDs are orphaned.

This report enables you to determine if database integrity was corrupted. If the result of this script is an empty list, your database does not contain any records with the specific problem.

This script produces CSV-formatted output on the console. There is one header row plus data rows. The number of rows is variable per report and can grow large. Rows are grouped by object type and ordered by parent object DBID within each type. The header row defines the columns of a data row, as follows:

- **prop_dbid:** The DBID of the property that has been duplicated.
- **object_dbid:** The DBID of the object, enabling a trace of the object in the database and the audit trail.
- **object_type:** The type of object, as visible in Genesys Administrator. The one exception is that a **User** or **Agent** object type is named **Person**.
- **object_name:** The name of the parent object.
- **object_tenant_name:** The name of the Tenant object to which this object belongs. This information is not provided for Folder object types.
- **section_name:** The name of the configuration option section in which the option is set.
- **prop_name:** The name of the duplicated property.
- **prop_value:** The value of the duplicated property.
- **prop_part:** The part number for the property, if the property has long multi-part values.

Sample printout, in CSV format

```
prop_dbid,object_dbid,object_type,object_name,object_tenant_name,section_name,prop_name,prop_value,prop_part
10000140,101,Switch,A0,Environment,A,a,aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa,1
10000140,101,Switch,A0,Environment,A,a,aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbb,1
10000139,101,Switch,A0,Environment,,,A,,0
140,101,Switch,A0,Environment,A,a,aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa,1
140,101,Switch,A0,Environment,A,a,aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbb.1
139,101,Switch,A0,Environment,,,A,,0
```

Supported DBMS

- Oracle 11g and newer; can be launched using Oracle SQL Developer 1.5 or newer
- MSSQL 2005 and newer
- PostgreSQL 9.0 and newer

search_orphan_flex_prop.sql

This script reports the DBID of any Annex options for which their parent DBID is a section that does not exist. In other words, the option is an orphan. If the result of this script is an empty list, your database does not contain any orphan records.

This script produces CSV-formatted output to the report file. There is one header row and any number of data rows, one for each orphan Annex records for which their parent DBID is of a section that does not exist. The number of rows is variable per report and can grow large. Rows are ordered by the Annex option DBID number. The header row defines the columns of a data row, as follows:

- **dbid:** The DBID of the Annex option.
- **object_dbid:** The DBID of the parent option.
- **object_type:** The numeric identifier of the object type.
- **parent_dbid:** The DBID of the section in which the option is configured.
- **prop_name:** The name of the property.
- **prop_value:** The value of the property.

Sample printout in CSV format

dbid,object_dbid,object_type,parent_dbid,prop_name,prop_value
208096,1123,12,208070,70921417,282254797065223d2253657
208128,1963,12,208118,328025376,282254797065223d22536572
237339,1140,16,236807,English,MH
237340,1140,16,236807,Bonds,HH
237341,1140,16,236807,Stocks,HH
442818,2718,16,442817,*warm,8667764793
477674,2718,16,442817,exit_composite,8667764793
782438,104,43,782437,value,
782440,104,43,782439,value,1
782446,104,43,782445,value,1
782448,104,43,782447,value,1
782450,104,43,782449,value,1

Supported DBMS:

- Oracle 11g and newer
- MSSQL 2005 and newer
- PostgreSQL 9.0 and newer

csv_check_cfg_max_dbid.sql

This script calculates metrics for each object type, all <object-type> Group types, and the history log, as follows:

- The number of objects of this type in the Configuration Database.
- The maximum value of DBID assigned to objects or records of the type or log.
- The value of MAX_DBID for that object type in the CFG_MAX_DBID table, or, for log records, the notify_id field in the CFG_REFRESH table.

The script also indicates if there are any discrepancies between the second and third values.

This script produces CSV-formatted output on the console. There are 41 data rows—one for the history log, one for all <object-type> Group types, and one for each of the 39 object types—and a header row. Each data row consists of information for the respective log, type Groups, or type. The header row defines each column of a data row, as follows:

- **object_type:** The type of object, as visible in Genesys Administrator with three exceptions—a User or Agent object type is labelled Person type, the History Log is labelled cfg_hdb_object type, and the cumulative data for all <object-type> Group types is labelled All Groups type).
- **object_count:** The number of objects of that type in the Configuration Database.
- **object_max:** The maximum value of DBIDs assigned to objects of this type, based on the maximum DBID value in the type's objects table.
- **cfg_max:** The value of max_dbib in the **cfg_max_dbid** table for the object_type.
- **test:** Any discrepancies; PASS if cfg_max >= object_max and FAILED if cfg_max < object_max.

Sample printout in CSV format

```
object_type,object_count,object_max,cfg_max,test
History log,151,194,194,PASS
All Groups,9,109,109,PASS
Switch,101,101,101,PASS
DN,101,101,101,PASS
......
Role,101,101,101,PASS
```

Supported DBMS

- Oracle 11g and newer

- MSSQL 2005 and newer
- PostgreSQL 9.0 and newer