Genesys Info Mart Deployment Guide

Info Mart Database Scripts

5/1/2025

Info Mart Database Scripts

This page describes how to modify and run the SQL scripts needed to create the Info Mart database and predefined views. This page also describes how to configure the Info Mart database to optimize performance of the merge operation for voice interactions.

Contents

Before You Begin

The following information is important for you to know:

- The Genesys-supplied SQL scripts are provided in the **sql_scripts** folder in your Genesys Info Mart installation package. They are also available as a separate SQL Scripts installation package. Use your database-specific tool (for example, SQL *Plus) to run the supplied SQL scripts.
- The Genesys Info Mart–provided SQL scripts do not qualify database objects by their schema or owner. When you run the SQL scripts, make sure that you use the ID of the schema or owner when you log in to the database. (You noted the schema or owner ID and password of each database in the appropriate section of the Database Worksheets.)
- The Genesys Info Mart–provided SQL scripts create objects without specifying tablespaces or storage parameters. Work with your database administrator or data-warehousing specialist to develop a database implementation that is optimal for your environment, and make the necessary changes to the SQL scripts. See Database Considerations for more information.

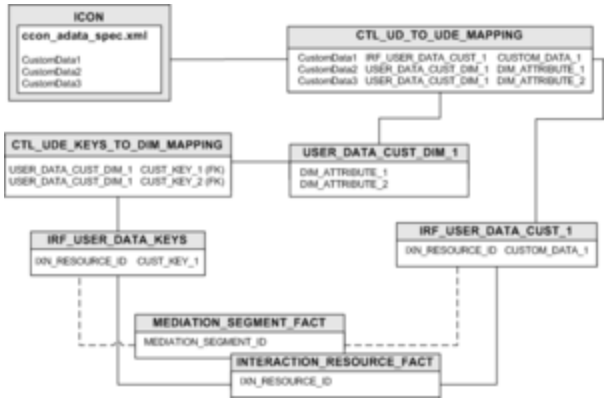
Preparing Custom User-Data Storage

Genesys Info Mart provides SQL scripts to use as a template for modifications you can make to the Info Mart database schema to customize user-data reporting:

- **make_gim_UDE_template.sql** — For use with nonpartitioned databases. For Microsoft SQL Server, use this script for single-language databases.
- **make_gim_UDE_template_partitioned.sql** — For use with partitioned databases. For Microsoft SQL Server, use this script for single-language databases.
- **make_gim_UDE_template_multilang.sql** — Starting with release 8.5.007, for use with nonpartitioned, multi-language databases in Microsoft SQL Server deployments.
- **make_gim_UDE_template_multilang_partitioned.sql** — Starting with release 8.5.007, for use with partitioned, multi-language databases in Microsoft SQL Server deployments.

You modify the applicable user-data template script, as required, to create custom user-data extension tables and columns and to specify storage of custom KVPs. You can use the User Data Assistant, **User_Data_Assistant.xlsm,** to help prepare the customized script.

The following Figure shows the relationships that the user-data template script creates, to illustrate how custom user data is stored in the Info Mart database and populated in interaction records. Inclusion of the `MEDIATION_SEGMENT_FACT` (MSF) table in the Figure illustrates optional storage of user data for interactions that are in mediation.



Custom User Data Storage

You can define the names of the custom tables and columns as you choose to see them in the Info Mart database. In custom fact tables, you can also specify the data types — character, numeric, or date/time — for the columns that store KVP values.

To make the best use of the flexible user-data storage that Genesys Info Mart provides, Genesys recommends that you use table and column names that reflect the meaning of the user-data KVPs in your deployment. Meaningful names of columns in recognizable user-data extension tables makes it easier to write unambiguous reporting queries. However, in multi-language databases, do not use non-Latin Unicode characters in table or column names. If you use the Data Export feature with export views (supported in on-premises deployments starting with release 8.5.011.22), table names must not be longer than 26 characters.

For more information about planning user-data storage in your deployment, see Storing User Data.

Preparing the User Data Script

Use the following procedure to prepare the SQL script to customize user-data storage in the Info Mart database. You can modify the script and use it to customize the user-data tables in the Info Mart database schema at any time.

Even if you use the User Data Assistant to prepare the customized script for your deployment, Genesys recommends that you review the information in the following manual procedure, so that you can verify the validity of the generated script before you execute it.

Procedure: Customizing the user-data template script

Purpose: To customize the Genesys-provided user-data script in order to specify user-defined KVP names and define custom user-data extension tables.

Prerequisites

- The worksheet for mapping user-data keys that are used for reporting in your environment is complete. For the mapping worksheet, see Mapping User Data Worksheet. Alternatively, if you are using the User Data Assistant, the Business Analyst and Report Developer tabs have been completed.
- The ICON application has been configured to store the required user-data KVPs.

Steps

1. Locate a copy of the applicable template script (**make_gim_UDE_template.sql**, **make_gim_UDE_template_partitioned.sql**, **make_gim_UDE_template_multilang.sql**, or **make_gim_UDE_template_multilang_partitioned.sql**) in the RDBMS-specific **sql_scripts** folder on the Genesys Info Mart product CD.
2. Save a copy of the script to a local machine.
3. Modify your copy of the script to provide columns in a user-data fact table to store high-cardinality KVPs that you will use in your reports. By default, the script creates a table named `IRF_USER_DATA_CUST_1`.
   Note the following about modifications you can make:
   - **The table name** — You can change the name of the `IRF_USER_DATA_CUST_1` table to any name that you want to see in the Info Mart database. However, if you change the name, ensure that you change all instances in the script, including the parts of the script that are described in Steps 11 and 4.
   - **Column names** — If you are modifying the script to prepare for the initial deployment, simply replace default names for the columns that store KVP values, such as `CUSTOM_DATA_1`, with names that are more meaningful in your deployment. Genesys recommends that you use the actual names of the high-cardinality KVPs.
   - **Character data** — Starting with release 8.5.007, you can increase the size of the data type for columns that store character data from 255 to 1024 characters. For information about enabling support for Unicode characters, see Multi-Language Support.
   - **Numeric data** — If you want to store particular KVP values as numeric data, change the data type for the columns that store those KVP values to any numeric data type that is supported by your RDBMS.
   - **Date/Time data** — If you want to store particular KVP values as date/time data, change the data type for the columns that store those KVP values to one of the following data types:
     - For Microsoft SQL Server, `DATETIME`
     - For Oracle, `DATE` or `TIMESTAMP`
     - For PostgreSQL, `TIMESTAMP`

     If the date/time that you want to store is in the Genesys Info Mart default format for date/time (`yyyy-mm-ddThh24:mi:ss.ff`), you do not need to perform any further mapping. If the date/time is in another date format, you must specify the conversion expression when you map the KVP to the fact table column (see Step 12).

- **Updating the database schema** — If you are modifying the script to update the database schema after Genesys Info Mart has already been deployed, you must:
  1. Delete the `DROP TABLE` SQL statement, which appears in the template script before the `CREATE TABLE` statement.
  2. Change the `CREATE TABLE` SQL statement to an `ALTER TABLE` one.

4. Modify the script, as required, to create an index for the user-data fact table that you created in Step 3.
5. Continue modifying your copy of the script to provide columns in user-data dimension tables to store low-cardinality KVPs that you will use in your reports. The script provides placeholders for tables named `USER_DATA_CUST_DIM_1` and `USER_DATA_CUST_DIM_2`.
   Note the following about modifications you can make:
   - **The table name** — You can change the name of the `USER_DATA_CUST_DIM_1` table to any name that you want to see in the Info Mart database. However, if you change the name, ensure that you change all instances in the script, including the parts of the script described in Step 6 and Steps 8 through 11.
   - **Column names** — If you are modifying the script to prepare for the initial deployment, simply replace default column names, such as `DIM_ATTRIBUTE_1`, with names that are more meaningful in your deployment. Genesys recommends that you use the actual names of the low-cardinality KVPs.
   - **Updating the database schema** — If you are modifying the script to update the database schema after Genesys Info Mart has already been deployed, you must:
     1. Delete the `DROP TABLE` SQL statement, which appears in the template script before the `CREATE TABLE` statement.
     2. Change the `CREATE TABLE` SQL statement to an `ALTER TABLE` one.

   > **Important**
   > Do not modify the data types or the mandatory status of the columns. Genesys Info Mart does not support numerical data types or nullable columns for user-data dimensions.

6. Modify the script, as required, to create an index for the user-data dimension table that you created in Step 5.
7. If necessary, repeat Steps 3 through 6 to add SQL commands to create additional custom user-data fact and dimension tables.
8. Modify the script, as required, to create foreign key reference(s) for the user-data dimension table(s) in the `IRF_USER_DATA_KEYS` table.
   The script includes the following placeholders:
   - `CUSTOM_KEY_1` and `CUSTOM_KEY_2` — The name of the foreign key that Genesys Info Mart will use to reference the user-data dimension table that you created in Step 5. Genesys recommends that you use a key name that provides an obvious association with the table name. You map this key to the referenced table later (Step 10).

> **Warning**
>
> Do not change the data type of the fields that you add to the IRF_USER_DATA_KEYS table. In releases earlier than 8.5.011.14, also do not change the mandatory status or the default value of your custom fields. (The default value -2 indicates NO_VALUE.)

> **Tip**
>
> Adding columns to a big IRF_USER_DATA_KEYS table can consume significant DBMS resources and time. If you are modifying the script to prepare for the initial deployment, consider adding redundant columns in advance. Later, you can map new user-data dimensions to existing IRF_USER_DATA_KEYS columns, as required.

9. For the user-data dimension table(s) that you created in Step 5, modify the script, as required, to populate the table(s) with mandatory values for predefined keys (for example, UNKNOWN). By default, the script inserts the required values into a table named USER_DATA_CUST_DIM_1.
10. Map the user data dimension table(s) to the foreign key(s). To do so, modify the script to add to the CTL_UDE_KEYS_TO_DIM_MAPPING table the mapping between the user-data dimension table(s) and the foreign key(s) that you added to the IRF_USER_DATA_KEYS table (Step 8). The script includes the following placeholders:
    - USER_DATA_CUST_DIM_1 — The user-data dimension table name (which you defined in Step 5)
    - ID — The primary key for the user-data dimension table
    - CUSTOM_KEY_1 — The foreign key for the user-data dimension table (which you specified in the IRF_USER_DATA_KEYS table in Step 8)
11. Map user-data keys to user-data fact and dimension table columns. For each column that you defined for user-data fact and dimension tables (see Steps 3 through 7), modify the script to:
    - Add to the CTL_UD_TO_UDE_MAPPING table the mapping between user-data keys and the user-data table columns.
    - Specify default values.
    - Specify the custom conversion expression (for custom date conversion in user-data fact tables).

      Use the worksheet that you prepared for user-data mapping to identify the required script changes. To customize the date conversion expression, see Step 12.

      The script includes the following placeholders for fact tables:

    - CustomData*N* — The key name (as stored by ICON)
    - IRF_USER_DATA_CUST_1 — The user-data fact table name (which you defined in Step 3)
    - CUSTOM_DATA_*N* — The column name (which you defined in Step 3)

The script includes the following placeholders for dimension tables:

- `CustomAttribute`*`N`* — The key name (as stored by ICON)
- `USER_DATA_CUST_DIM_1` — The user-data dimension table name (which you defined in Step 5)
- `DIM_ATTRIBUTE_`*`N`* — The column name (which you defined in Step 5)

The script also requires you to specify the propagation rule, default value, and activity status for each KVP. For more information about values for these fields, see the column descriptions for the `CTL_UD_TO_UDE_MAPPING` table in the *Genesys Info Mart Reference Manual* for your RDBMS.

Ensure that the default values that you specify are consistent with the data type for the column.

12. If you want Genesys Info Mart to store a date/time value expressed in a format other than the Genesys Info Mart default format for date/time (for example, `DD Mon YY` instead of `yyyy-mm-ddThh24:mi:ss.ff`), specify the conversion expression in the `CONVERT_EXPRESSION` field in the `CTL_UD_TO_UDE_MAPPING` table entry for the KVP. Genesys Info Mart includes the conversion expression in SQL statements to convert the data.

    - For Microsoft SQL Server, the conversion expression is:

      ```
      ${schema}.GIM_TO_TIMESTAMP_ISO8601(${})
      ```

      where:

      - `${schema}` is a placeholder for the Info Mart database schema name; Genesys Info Mart gets the value of the `${schema}` parameter from the default-schema option in the Info Mart DAP.
      - `${}` is a placeholder for the KVP value to be converted.
      - **GIM_TO_TIMESTAMP_ISO8601** is an out-of-box function, which first executes another out-of-box function, **GIM_IS_ISO8601_DATE**, to check format before calling the Microsoft SQL Server system function **convert(datetime, ${}, 126)** to convert the datetime expression in format `yyyy-mm-ddThh24:mi:ss.ff`.

        It is not possible to use **convert()** without first checking the format of the expression because of a Microsoft SQL Server limitation that makes transactions unusable after most conversion errors. To customize the conversion, you must define your own conversion function in the database, and then call that function in the `CONVERT_EXPRESSION` field using the syntax shown above. Use the **GIM_TO_TIMESTAMP_ISO8601** and **GIM_IS_ISO8601_DATE** functions, which are defined in the **make_gim.sql** and **make_gim_partitioned.sql** scripts, as examples for your custom functions, in conjunction with RDBMS documentation about syntax requirements and date formats. Do not modify the out-of-box functions.

    - For Oracle, the conversion expression is:

      ```
      TO_DATE(${},'yyyy-mm-dd'T'hh24:mi:ss')
      ```

      or

      ```
      TO_TIMESTAMP(${},'yyyy-mm-dd'T'hh24:mi:ss.ff')
      ```

      where `${}` is a placeholder for the KVP value to be converted.

To customize the conversion, use one of the standard RDBMS-provided functions (**TO_DATE** or **TO_TIMESTAMP**), replacing the date-format expression with a valid format, as defined in the RDBMS documentation.

- For PostgreSQL, the conversion expression is:
  ```
  TO_TIMESTAMP(${},'yyyy-mm-dd'T'hh24:mi:ss.ms')
  ```
  where `${}` is a placeholder for the KVP value to be converted.

13. Save the modified copy of the script.

Next Steps

- Execute the modified script when you create the Info Mart database schema (see **Creating the Info Mart database schema,** Step 4) or as required to update an existing database schema. To verify correct mappings, execute the following SQL command against the Info Mart database:

  ```
  SELECT * FROM CTL_UD_TO_UDE_MAPPING
  ```

  Compare the results of the query against the mapping that you prepared before customizing the script, as described in the Prerequisites.

  > **Important**
  > Before you execute the script to update an existing database schema, Genesys recommends that you back up the Info Mart database.

- If you update an existing database schema that uses read-only tenant views to access Info Mart data for reports, you must re-create the read-only tenant views. For more information, see Creating Read-Only Tenant Views.

Preparing the Info Mart Database

The RDBMS-specific SQL scripts that are provided with Genesys Info Mart create the Info Mart database schema. This includes merge tables, which are a required part of the Info Mart database schema in any deployment. Genesys Info Mart provides separate scripts for partitioned and nonpartitioned database schemas. For Microsoft SQL Server, starting with release 8.5.007 Genesys Info Mart also provides separate scripts for single-language and multi-language databases.

The Genesys Info Mart database scripts do not create the additional database objects that are required to support aggregation. For more information about database preparation for deployments that use the Genesys historical reporting presentation layer (GCXI or the separately installed RAA package, see the *Reporting and Analytics Aggregates 8.x Deployment Guide* and the *Genesys CX Insights Deployment Guide*.

Procedure: Creating the Info Mart database schema

Prerequisites

- For multi-language databases, you have reviewed requirements for the Info Mart database and other system components (see the links provided in Multi-Language Support) and prepared your environment accordingly.
- You have created a database instance for your Info Mart database.
- The required SQL scripts are available from the RDBMS-specific **sql_scripts** folder on the Genesys Info Mart product CD.

> **Important**
>
> If you are creating a partitioned database schema on Microsoft SQL Server, do not alter **make_gim_partitioned.sql** or **make_gim_multilang_partitioned.sql** so as to create the partitions in multiple filegroups.

- (Optional) If you plan to store user-defined attached data, you have customized the applicable user-data SQL script template, as instructed in Preparing the User Data Script.
- (Optional) If you plan to create multiple calendar dimensions to support your reporting, you have customized the applicable database-creation script to create additional calendar tables. Alternatively, you can create the custom calendars after you have installed Genesys Info Mart. For more information, see Creating Custom Calendars.

Steps

1. Ensure that the database access account that you use to create the Info Mart database schema is available and has the required privileges (see Required Database Privileges).
   Refer to your completed Database Worksheets to determine the ID to use.
2. Log in to the Info Mart database using the Info Mart user account.
3. Run the applicable SQL script to create the Info Mart database schema:
   - For a nonpartitioned database, use **make_gim.sql.** This script creates the Genesys Info Mart dimension and fact tables and related indexes. On Microsoft SQL Server, starting with release 8.5.007 use **make_gim.sql** for a single-language database and **make_gim_multilang.sql** for a multi-language database.
   - For a partitioned database, use **make_gim_partitioned.sql.** This script creates the Genesys Info Mart dimension and fact tables and related indexes. On Microsoft SQL Server, starting with release 8.5.007 use **make_gim_partitioned.sql** for a single-language database and **make_gim_multilang_partitioned.sql** for a multi-language database. For the tables and indexes that are partitioned, this script creates a single, outdated partition that is expected to be purged during the first run of the maintenance job.
4.

(Optional) Run the modified UDE template script (**make_gim_UDE_template.sql**, **make_gim_UDE_template_partitioned.sql**, **make_gim_UDE_template_multilang.sql**, or **make_gim_UDE_template_multilang_partitioned.sql**) that you have updated with required KVP names. This script creates extension tables in the Info Mart database schema to store custom user-data, configures user-data mappings, and adds the specified dimension key fields to the IRF_USER_DATA_KEYS table.

5. Ensure that the database access account that the ETL jobs will use to access the Info Mart database is available and has the required user account privileges (see Required Database Privileges).
   Refer to your completed Database Worksheets to determine the ID to use.

   The user account does not have to be the same as the owner account. For more information about the rules and recommendations that pertain to database access accounts for Genesys Info Mart, see Database Object Owners and User IDs.

Next Steps

- (Required for Voice details only) Update the GSYS_DNPREMOTELOCATION table, as required, to optimize performance of the merge operation. For more information, see Configuring the Info Mart database for merge.
- (Recommended, but optional) Configure database links. For more information, see Optimizing Database Performance: Database Links.
- Tune up your Info Mart database, as appropriate for your RDBMS environment. For more information, see Optimizing Database Performance: Database Tuning.

Procedure: Configuring the Info Mart database for merge

Purpose: To optimize performance of the merge operation.

Prerequisites

- The Info Mart database schema has been created, as described in Creating the Info Mart database schema.

Steps

1. If any switches are not monitored by ICON, store those Switch object names in the GSYS_DNPREMOTELOCATION table of the Info Mart schema. Otherwise, merging of some interswitch voice interactions will be delayed until the configured IS-Link timeout occurs, and this delays transformation of those voice interactions.
   Example

   Suppose that you have four switches and two ICON instances:

   - ICON1 monitors switch SITE1_sw1.

- ICON2 monitors switch SITE2_sw2.
- SITE3_sw3 and SITE4_sw4 are not monitored by either ICON instance.

  To avoid delays in merging, add the following two records to the `GSYS_DNPREMOTELOCATION` table in the Info Mart schema:

- `GSYS_DNPREMOTELOCATION.REMOTELOCATION=SITE3_sw3`
- `GSYS_DNPREMOTELOCATION.REMOTELOCATION=SITE4_sw4`

  For each row in the `GSYS_DNPREMOTELOCATION` table, populate the `ID` field with a unique value.

2. Review the settings of the max-call-duration, merge-chunk-size, and merge-failed-is-link-timeout configuration options, and modify them as required for your deployment.

Next Steps

- (Recommended, but optional) Configure database links. For more information, see Optimizing Database Performance: Database Links.
- Tune up your Info Mart database, as appropriate for your RDBMS environment. For more information, see Optimizing Database Performance: Database Tuning.