Oracle Life Sciences Argus Mart

Extensibility Guide



Release 2026.1.01
G52290-01
March 2026

Oracle Life Sciences Argus Mart Extensibility Guide, Release 2026.1.01

G52290-01

Copyright © 2019, 2026, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

About this content i
Related resources i

1 Extensibility assumptions

2 Introduction to Oracle Argus Mart ETL

3 ETL Hooks

List of Oracle Argus Mart ETL Hooks 2
View Oracle Argus Safety User Interface 2
Execute ETL Hooks 3

4 Configure Flexible Data Re-categorization Code List

5 Customize Tables

Customize RM Tables 1
- Populate a Column Added in the Oracle Argus Safety table 1
- Populate a Calculated Column in RM 2

Customize SM Tables 3
- Define a New Column in SM table 4
- Populate a Custom UVT table 4

6 Customize Table for CLOB columns

Copyright © 2019, 2026, Oracle and/or its affiliates.

Preface

This preface contains the following sections:

- About this content
- Related resources

About this content

Except where noted, information in this guide applies to both Oracle Argus Safety and Oracle Safety One Argus environments.

Related resources

For information about Oracle Argus patches, see My Oracle Support.

All documentation and other supporting materials are available on the Oracle Help Center.

Copyright © 2019, 2026, Oracle and/or its affiliates.

1

Extensibility assumptions

This document contains information that helps a database expert extend the Oracle Argus Mart data structure or logic for use with custom data.

This guide assumes the following:

- Oracle Argus Mart (AM) is dependent on Oracle Argus.
- You should avoid executing any functionality that may be implemented in a simpler and more general way in Oracle Argus.
- It is not necessary to wrap PL/SQL components of the ETL.

Copyright © 2019, 2026, Oracle and/or its affiliates.

2

Introduction to Oracle Argus Mart ETL

Oracle Argus Mart is a data source for analysis and reporting in medical product safety and pharmacovigilance. The primary data is the adverse event cases managed by the Oracle Argus Safety application. Oracle Argus Mart consists of one or more data marts and the software to `Extract` the source data from Oracle Argus Safety; `Transform` the data, and `Load` it into the Oracle Argus Mart data marts. The two sets of tables used in Oracle Argus Mart are:

- Reporting Mart (RM)
- Signal Mart (SM)

The data can be customized before or after fetching data from Oracle Argus Safety database with the help of ETL Hooks. The next section comprises the details about ETL Hooks.

Copyright © 2019, 2026, Oracle and/or its affiliates.

3
ETL Hooks

> **Note**
>
> This section applies only to Argus Safety environments.

ETL Hooks are the custom procedures you can use to extend the existing functionality or behavior of the Oracle Argus Mart ETL.

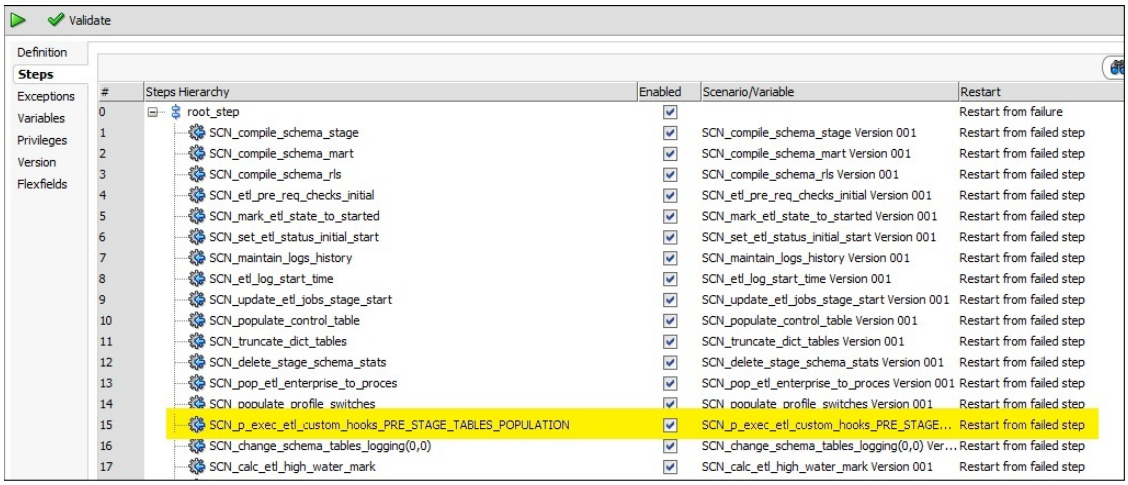
ETL Hooks can be executed to perform actions while fetching data from the source database to the destination database.

The data fetched from the Oracle Argus Safety database can be modified before or after uploading the data into the Oracle Argus Mart database. You can populate customized (non-standard/user-defined) columns to the standard RM and SM tables. You can also populate customized table(s).

For more information on populating customized columns or tables, refer to Customize Tables.

ETL Hooks perform these actions at different stages of both initial and incremental ETL.

The Oracle Data Integrator Load Plan lists all the ETL scenarios, as shown below:



For more information, see:

- List of Oracle Argus Mart ETL Hooks
- View Oracle Argus Safety User Interface
- Execute ETL Hooks

Copyright © 2019, 2026, Oracle and/or its affiliates.

List of Oracle Argus Mart ETL Hooks

> **Note**
>
> This section applies to both Argus Safety and Safety One Argus environments.

Oracle Argus Mart comprises the following ETL Hooks, which can be executed at different stages:

- CUSTOM ROUTINE BEFORE STAGE TABLES POPULATION—Execute before populating the Staging tables.
- CUSTOM ROUTINE BEFORE REPORTING TABLES POPULATION—Execute before populating the Reporting tables (or RM tables).
- CUSTOM ROUTINE AFTER REPORTING TABLES POPULATION—Execute after populating the Reporting tables (or RM tables).
- CUSTOM ROUTINE BEFORE SIGNAL HELPER TABLES POPULATION—Execute before populating the Signal Helper tables.
- CUSTOM ROUTINE AFTER SIGNAL HELPER TABLES POPULATION—Execute after populating the Signal Helper tables.
- CUSTOM ROUTINE AFTER ETL—Execute after running the initial or incremental ETL (post-ETL commit).

> **Note**
>
> ETL hooks CUSTOM ROUTINE BEFORE SIGNAL HELPER TABLES POPULATION and CUSTOM ROUTINE AFTER SIGNAL HELPER TABLES POPULATION are executed only when any case revision is being processed for SM tables in the current ETL run.

View Oracle Argus Safety User Interface

> **Note**
>
> This section applies to both Argus Safety and Safety One Argus environments.

ETL Hooks help you customize the existing ETL to suit your requirements through the Oracle Argus Safety user interface.

To view the Oracle Argus Safety user interface:

1. Log in to the Oracle Argus Safety application.
2. From the menu bar, click **Argus Console**.

   The menu bar updates for Oracle Argus Safety Console.
3. From the new menu bar, click **System Configuration**.

Copyright © 2019, 2026, Oracle and/or its affiliates.

4. Click **System Management (Common Profile Switches)**.

   On the left hand side of the screen, a list of all system-managed common profile switches appears.

5. Expand the Common Profile folder and click **Argus Mart**.

   The **Modify Argus Mart** page appears.

> **Note**
>
> ETL hooks are the global-level switches visible in Oracle Argus Safety Console when you are logged in through the default enterprise. These switches are visible only after installing and creating the Oracle Argus Mart schema.

Execute ETL Hooks

> **Note**
>
> This section applies only to Argus Safety environments.

The ETL Hooks can be executed at different staging levels.

1. Before populating the reporting tables, execute CUSTOM ROUTINE BEFORE STAGE TABLES POPULATION and/or CUSTOM ROUTINE BEFORE REPORTING TABLES POPULATION.
2. Before populating the signal tables, execute ETL Hooks CUSTOM ROUTINE AFTER REPORTING TABLES POPULATION, CUSTOM ROUTINE BEFORE SIGNAL HELPER TABLES POPULATION, and/or CUSTOM ROUTINE AFTER SIGNAL HELPER TABLES POPULATION.
3. After populating reporting and signal tables and before populating the results, execute CUSTOM ROUTINE AFTER ETL.

Copyright © 2019, 2026, Oracle and/or its affiliates.

Figure 3-1 Sequence of ETL Hook Execution



To execute an ETL Hook:

1. On the **Modify Argus Mart** page, enter a procedure name in the text box relevant to the ETL Hook execution stage. The ETL type can be initial or incremental where the customized data will be fetched. The ETL Hook will look for the database object that matches the procedure name.

   A procedure name denotes the ETL Hook that comprises the custom procedure or steps. Based on the stage (initial or incremental) where this procedure is being called, the data is modified before being sent to the destination tables.

2. If the database object (or procedure) is found, the ETL is executed as follows:
   - If the ETL executes without any errors, then move to the next procedure.
   - If the ETL executes with errors, then log the error(s) and exit.

     When executing ETL, if there is any error while populating the staging schema tables, error(s) are logged in the table ETL_STAGE_LOG, whereas while populating the mart schema tables, error(s) are logged in the table ETL_MART_LOG.

3. If the database object (or procedure) is not found, then log the error(s) and fail the ETL. In this case, the ETL may be executed if you have explicitly created an exception-handling for such cases to absorb any exceptions and move on to the next procedure.

   Optionally, to resolve this issue, create a procedure of that name, provide an existing procedure name, or remove the configuration.

⚠ Warning

- You should not modify the existing data or names of the objects. Although additional objects can be created, you should first test all changes in a test environment before implementing them in a production environment.
- No data commit should be done within ETL Hooks.

Copyright © 2019, 2026, Oracle and/or its affiliates.

Copyright © 2019, 2026, Oracle and/or its affiliates.

4

Configure Flexible Data Re-categorization Code List

Flexible Data Re-categorization is an Oracle Argus function you can use to define code list display values (LM tables) in different languages.

Oracle Argus maintains this data in the tables CODE_LIST_MASTER, CODE_LIST_CODE_ATTRIBUTES, and CODE_LIST_DETAIL_DISCRETE for supported languages.

The Oracle Argus Mart ETL populates the code list data in the tables RM_CODE_LIST_MASTER, RM_CODE_LIST_CODE_ATTRIBUTES, and RM_CODE_LIST_DETAIL_DISCRETE, which are used for processing the SM table columns.

Based on the applicable language, the SM table fields are divided into the following categories:

- **LI - Language Independent fields**: These fields are applicable for first-human language or second-human language parameters. The values are the same, irrespective of the language code.
- **FL - First Language fields**: These fields are populated on the basis of the Argus Mart Common Profile Switch FIRST_HUMAN_LANGUAGE, which is present in Argus Console.

  The default value of this switch is **en**. As a result, the default language in which these fields will be populated is English.

  If the you configure this switch with any other available language, and if its corresponding data is also present in the Flexible Data Re-categorization code list tables, these first language fields will contain data for the configured language.
- **SL - Second Language fields**: These fields are populated on the basis of the Oracle Argus Mart Common Profile Switch SECOND_HUMAN_LANGUAGE, which is present in Argus Console.

  The default value of this switch is NULL, which means that the second language fields will be populated in the SM tables only after this switch is configured.

  If you configure this switch with any other configured language, for example German, and:

  – If its corresponding data is also present in the Flexible Data Re-categorization code list tables, these second language fields will contain data for the configured language.

  – If its corresponding data is not available in the Flexible Data Re-categorization code list tables, these second language fields will contain data in English.
- **EN - English fields**: These fields are always populated for English only.
- **J - Japanese fields**: These fields are applicable only if the second-human language parameter is set to Japanese.
- **EN_ABBRV**: These fields use the EN_ABBRV language for the specified code list.
- **SM:** These fields use the SM language for the specified code list.

The following example shows how to configure a code list display value in a new language for a pre-existing code:

Copyright © 2019, 2026, Oracle and/or its affiliates.

Example 4-1 Configure Flexible Data Re-categorization

For a code list GENDER, data in the table CODE_LIST_DETAIL_DISCRETE for code 1 is available in the following four decode contexts (languages):

Figure 4-1 Original Decode Contexts (Languages)

CODE_LIST_ID	DECODE_CONTEXT	CODE	DISPLAY_VALUE	PREFERRED	SORT	LAST_UPDATE_TIME	ENTERPRISE_ID
GENDER	en	1	Male	0	(null)	05-FEB-13	1
GENDER	ja	1	男性	0	(null)	05-FEB-13	1
GENDER	E2B	1	1	0	(null)	05-FEB-13	1
GENDER	SM	1	M	0	(null)	05-FEB-13	1

To configure the same code 1 in the code list GENDER for a new language such as GERMAN (decode context **ge**), populate the table CODE_LIST_DETAIL_DISCRETE with required values in GERMAN. For example:

```
INSERT INTO CODE_LIST_DETAIL_DISCRETE (CODE_LIST_ID, DECODE_CONTEXT, CODE,
DISPLAY_VALUE, PREFERRED, SORT, LAST_UPDATE_TIME, ENTERPRISE_ID) VALUES
( 'GENDER', 'ge', 1, 'männlich', 0, null, sysdate, 1);
```

Figure 4-2 New Decode Contexts (Languages)

CODE_LIST_ID	DECODE_CONTEXT	CODE	DISPLAY_VALUE	PREFERRED	SORT	LAST_UPDATE_TIME	ENTERPRISE_ID
GENDER	en	1	Male	0	(null)	05-FEB-13	1
GENDER	ja	1	男性	0	(null)	05-FEB-13	1
GENDER	E2B	1	1	0	(null)	05-FEB-13	1
GENDER	SM	1	M	0	(null)	05-FEB-13	1
GENDER	ge	1	männlich	0	(null)	20-FEB-13	1

Copyright © 2019, 2026, Oracle and/or its affiliates.

5

Customize Tables

> **Note**
>
> This chapter applies only to Argus Safety environments.

You can customize RM and SM tables.

- Customize RM Tables
- Customize SM Tables

Customize RM Tables

- Populate a Column Added in the Oracle Argus Safety table
  You can populate new column(s) to existing RM tables that are added to the Oracle Argus Safety tables.
- Populate a Calculated Column in RM
  You can populate customized calculated column(s) to the existing RM tables that do not exist in the Oracle Argus Safety database. You can also populate column(s) which are calculated on the basis of the existing columns in a table.

Populate a Column Added in the Oracle Argus Safety table

You can populate new column(s) to existing RM tables that are added to the Oracle Argus Safety tables.

To populate the new column:

1. Check mapping between the source and the target tables, through the table *ETL_SIGNAL_TABLE_MAPPING*.
2. Based on the mapping between the Oracle Argus Safety and RM tables, fetch the desired column in the RM table through ETL Hooks.

There can be different source types to fetch data into tables at initial or incremental stages, such as:

- ATOS - Oracle Argus Safety to Staging tables
- DTOS - DLP to Staging tables
- STORM - Staging to RM tables
- STOSM - PRE_SM to SM Tables

Example 5-1 Populate a new column in the RM table

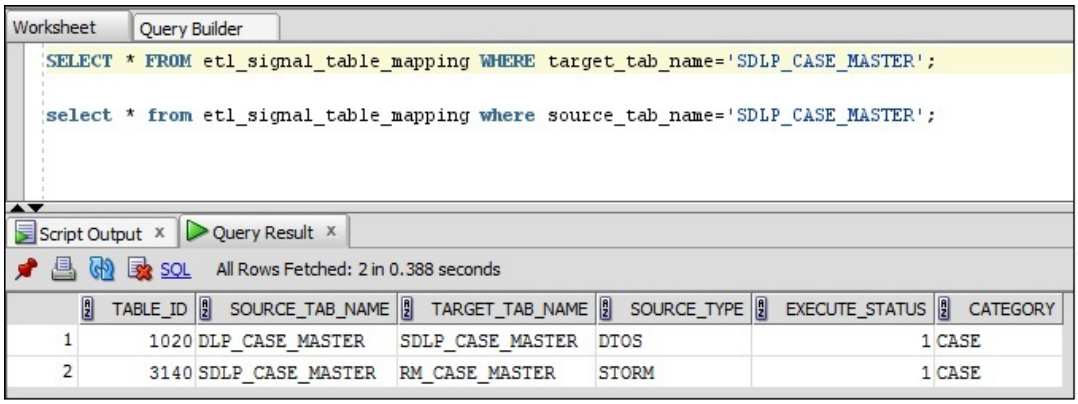
Follow the steps given below to populate a new column in the table RM_CASE_MASTER:

1. Check the mapping between RM_CASE_MASTER, and the source table (SDLP_CASE_MASTER) to fetch columns from the staging table into the RM table. In this case, the source type is STORM.

Copyright © 2019, 2026, Oracle and/or its affiliates.

2. Check the source table for target table (SDLP_CASE_MASTER), where columns are fetched from Oracle Argus Safety into the staging table. In this case, the source type can be ATOS/DTOS.

Figure 5-1 Populating New Columns in the RM Table



3. Once the mapping of the tables has been identified, fetch these column(s) from Argus Safety/DLP into Oracle Argus Mart, as required.

Populate a Calculated Column in RM

You can populate customized calculated column(s) to the existing RM tables that do not exist in the Oracle Argus Safety database. You can also populate column(s) which are calculated on the basis of the existing columns in a table.

To populate such custom columns, create a procedure, and use ETL Hooks to execute them.

Example 5-2 Create a custom procedure

Creating a procedure called P_UPD_RM_CASE_PAT_INFO, to populate a new column called PAT_FULL_NAME in the table RM_CASE_PAT_INFO. The value of this column is calculated, and populated from the existing columns of RM_CASE_PAT_INFO. Here, we concatenate three columns PAT_INITIALS, PAT_FIRSTNAME, and PAT_LASTNAME to populate this value as PAT_FULL_NAME.

1. Create a procedure called P_UPD_RM_CASE_PAT_INFO.
2. In the table RM_CASE_PAT_INFO, populate a new column called PAT_FULL_NAME.

   The value of this column is calculated, and populated from the existing columns of RM_CASE_PAT_INFO.
3. Concatenate three columns PAT_INITIALS, PAT_FIRSTNAME, and PAT_LASTNAME to populate this value as PAT_FULL_NAME.

Copyright © 2019, 2026, Oracle and/or its affiliates.

Figure 5-2 Create Custom Procedure

```
CREATE OR REPLACE PROCEDURE p_upd_rm_case_pat_info AS
BEGIN
  UPDATE rm_case_pat_info
     SET pat_full_name = pat_initials || ' ' || pat_firstname || ' ' || pat_lastname
   WHERE pat_full_name IS NULL;
EXCEPTION
  WHEN OTHERS THEN
    pkg_sm_logging.p_etl_mart_log (SUBSTR (SQLERRM, 1, 300),
                                   'p_upd_rm_case_pat_info',
                                   'Error during updation of RM_CASE_PAT_INFO.');
END p_upd_rm_case_pat_info;
/
```

4. Once this procedure is created, call this procedure using ETL Hook PRE_REPORTING_TABLES_POPULATION.

   The ETL Hook will populate this new column PAT_FULL_NAME in SDLP_CASE_PAT_INFO.

Customize SM Tables

When using ETL Hooks to populate data into SM tables, it is recommended to use RM tables as the source, with join(s) to the chunk table ETL_SM_CASES_TO_PROCESS_CHUNK.

Do not use RM Views such as V_ETL_CASE_MASTER, because each view is joined individually to the chunk table. This can lead to sluggish performance of the application if ETL Hook extracts data from the multiple views.

Example 5-3 Populate data into SM tables using the chunk table

In the example given below, two RM tables RM_CASE_MASTER and RM_CASE_PAT_INFO have been joined with the chunk table ETL_SM_CASES_TO_PROCESS_CHUNK to fetch data into the SM table with the help of the ETL Hooks.

```
SELECT *
  FROM rm_case_master rcm, rm_case_pat_info rcpi, etl_sm_cases_to_process_chunk chnk
 WHERE chnk.enterprise_id = rcm.enterprise_id
   AND chnk.case_id = rcm.case_id
   AND chnk.validstart >= rcm.effective_start_date
   AND chnk.validstart < rcm.effective_end_date
   AND chnk.enterprise_id = rcpi.enterprise_id
   AND chnk.case_id = rcpi.case_id
   AND chnk.validstart >= rcpi.effective_start_date
   AND chnk.validstart < rcpi.effective_end_date;
```

For more information, see:

- Define a New Column in SM table
- Populate a Custom UVT table
  The Oracle Argus Mart ETL maintains Unique Value Tables (UVT) for first-human language and second-human language, as defined in the system. The UVTs contain the list of distinct values available for categorical data items in the case data.

Copyright © 2019, 2026, Oracle and/or its affiliates.

Define a New Column in SM table

New column(s) can be added before or after fetching data into SM tables. The columns to be added into SM can have simple or complex calculations.

- Simple calculation-based columns: SM Views are created based on Argus Safety tables. When a customized column having simple calculations is to be added in an SM table, these calculations are done on SM Views, such as:
  - `select <column-name> from <table-name> table where <column-name> DateOfBirth = '1990';`
  - The definition of these SM Views is based on RM tables and is not updated automatically. When calculations are done, make sure to update the views, to reflect the changes. Once SM Views are updated, use ETL Hooks to fetch this column(s) into SM table, as required.
- Complex calculation-based columns: *PRE_SM* table is a set of tables that comprises the staging data for SM tables. You can perform complex calculations on these tables and then fetch them into SM tables using ETL Hooks.

The basic procedure to define a new column is the same for both simple and complex calculation-based column(s) - SM View and PRE_SM tables.

Simple and complex calculations are segregated to improve performance of execution of the data.

Populate a Custom UVT table

The Oracle Argus Mart ETL maintains Unique Value Tables (UVT) for first-human language and second-human language, as defined in the system. The UVTs contain the list of distinct values available for categorical data items in the case data.

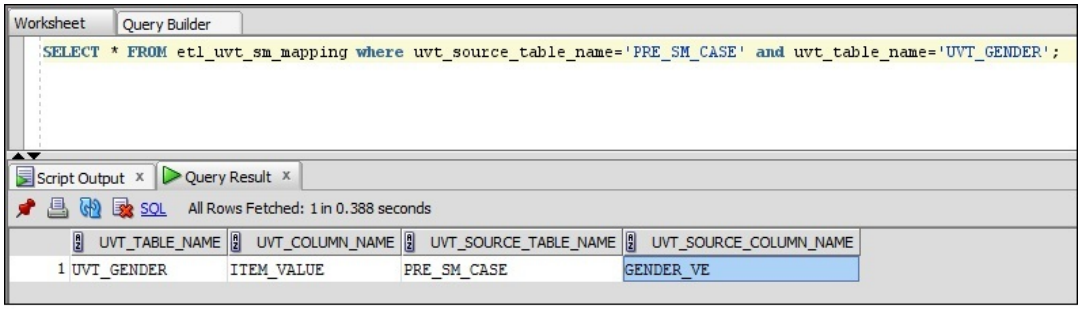
To populate a custom UVT table, you must first check the source and target table mapping from the table *ETL_UVT_SM_MAPPING*.

Fetch the data from the source table into the target table as per the mapping done above.

Example 5-4 Populate a custom UVT table

Follow the steps given below to populate a column in the table UVT_GENDER:

1. Check the mapping between the source table PRE_SM_CASE and the target table UVT_GENDER from the table ETL_UVT_SM_MAPPING.



2. After mapping the tables and their related columns, you can populate the column(s), as required.

Copyright © 2019, 2026, Oracle and/or its affiliates.

Copyright © 2019, 2026, Oracle and/or its affiliates.

6

Customize Table for CLOB columns

> **Note**
>
> This chapter applies only to Argus Safety environments.

To reduce the time taken to load the CLOB data from Oracle Argus source tables to Oracle Argus Mart stage tables, add or modify the data in the table ETL_SPLIT_CLOB_COLUMN_MAP available in Oracle Argus Mart schema.

With this customization, the ETL fetches only the first 32767 bytes of the source CLOB column to the respective column of the stage table. The value in the source CLOB column splits into twenty five VARCHAR2 columns at the source before moving the data to the respective stage column. Hence, reduced loading time of these tables, and reduced ETL time.

> **Note**
>
> - Enabling this feature fetches only the first 32767 bytes of the data from the source CLOB column to the respective stage table column. The rest of the data is not copied.
> - Using this feature is optional; enable it if appropriate for your organization's needs.

The following are the table details for enabling or disabling this customization:

COLUMN NAME	DATA TYPE (LENGTH)	DESCRIPTION
TABLE_NAME	VARCHAR2 (30)	Defines the Oracle Argus source table name from which data is populated in to Oracle Argus Mart stage table. For example, DLP_CASE_NARRATIVE.
COLUMN_NAME	VARCHAR2 (30)	Defines the name of the CLOB column present in the source table. For example, ABBREV_NARRATIVE.
ENABLE_YN	VARCHAR2 (1)	Defines the values for this column. Y—The feature is enabled N—The feature is disabled

Besides, as part of the Factory Data, the following columns of the table ETL_SPLIT_CLOB_COLUMN_MAP are disabled by default (ENABLE_YN = N):

TABLE_NAME	COLUMN_NAME	ENABLE_YN
CMN_REG_REPORTS	NULLIFICATION_REASON	N
DLP_CASE_ASSESS	BFARM_MANUAL_TEXT	N
DLP_CASE_ASSESS	EVALUATION	N

Copyright © 2019, 2026, Oracle and/or its affiliates.

TABLE_NAME	COLUMN_NAME	ENABLE_YN
DLP_CASE_ASSESS	EVALUATION_J	N
DLP_CASE_DOSE_REGIMENS	DOSE_DESCRIPTION	N
DLP_CASE_DOSE_REGIMENS	DOSE_DESCRIPTION_J	N
DLP_CASE_EU_DEVICE	CORRECTIVE_ACTION	N
DLP_CASE_EU_DEVICE	CORRECTIVE_ACTION_FINAL	N
DLP_CASE_EU_DEVICE	COUNTRIES_OF_DISTRIBUTION	N
DLP_CASE_EU_DEVICE	CURRENT_DEV_LOCATIONS	N
DLP_CASE_EU_DEVICE	FURTHER_INVESTIGATION	N
DLP_CASE_EU_DEVICE	INDENTIF	N
DLP_CASE_EU_DEVICE	INVESTIGATION_RESULT	N
DLP_CASE_EU_DEVICE	MANUFACTURER_COMMENTS	N
DLP_CASE_EU_DEVICE	PROJECTED_TIMING	N
DLP_CASE_EU_DEVICE	PROJECTED_TIMING_FINAL	N
DLP_CASE_EVENT	DETAILS	N
DLP_CASE_EVENT	DETAILS_J	N
DLP_CASE_FOLLOWUP	JUSTIFICATION	N
DLP_CASE_FOLLOWUP	JUSTIFICATION_J	N
DLP_CASE_JUSTIFICATIONS	J_TEXT	N
DLP_CASE_JUSTIFICATIONS	J_TEXT_J	N
DLP_CASE_LAB_DATA	COMMENTS	N
DLP_CASE_LAB_DATA	COMMENTS_J	N
DLP_CASE_LAB_DATA	NOTES	N
DLP_CASE_LAB_DATA	NOTES_J	N
DLP_CASE_NARRATIVE	ABBREV_NARRATIVE	N
DLP_CASE_NARRATIVE	ABBREV_NARRATIVE_J	N
DLP_CASE_NOTES_ATTACH	NOTES	N
DLP_CASE_NOTES_ATTACH	NOTES_J	N
DLP_CASE_PAT_HIST	NOTE	N
DLP_CASE_PAT_HIST	NOTE_J	N
DLP_CASE_PAT_INFO	NOTES	N
DLP_CASE_PAT_INFO	NOTES_J	N
DLP_CASE_PROD_DEVICES	NARRATIVE_TEXT	N
DLP_CASE_PROD_DEVICES	PRELIMINARY_COMMENTS	N
DLP_CASE_PROD_DEVICES	PRELIMINARY_COMMENTS_J	N
DLP_CASE_PRODUCT	COMMENTS_TIMEFRAME	N
DLP_CASE_PRODUCT	GENERIC_NAME	N
DLP_CASE_PRODUCT	GENERIC_NAME_J	N
DLP_CASE_PRODUCT	NOTES	N
DLP_CASE_PRODUCT	NOTES_J	N

Copyright © 2019, 2026, Oracle and/or its affiliates.

TABLE_NAME	COLUMN_NAME	ENABLE_YN
DLP_CASE_PRODUCT	QC_ANALYSIS_CAT_TEXT	N
DLP_CASE_PRODUCT	QC_ANALYSIS_CAT_TEXT_J	N
DLP_CASE_PRODUCT	QC_ANAL_SUMMARY_TEXT	N
DLP_CASE_PRODUCT	QC_ANAL_SUMMARY_TEXT_J	N
DLP_CASE_PRODUCT	QC_COMMENT	N
DLP_CASE_PRODUCT	QC_COMMENT_J	N
DLP_CASE_PRODUCT	QC_COMPLAINT_CAT_TEXT	N
DLP_CASE_PRODUCT	QC_COMPLAINT_CAT_TEXT_J	N
DLP_CASE_PRODUCT	QC_RESULT	N
DLP_CASE_PRODUCT	QC_RESULT_J	N
DLP_CASE_PRODUCT	REASON_DOWNGRADE	N
DLP_CASE_PRODUCT	RETRO_INFECTION	N
DLP_CASE_REPORTERS	NOTES	N
DLP_CASE_REPORTERS	NOTES_J	N
DLP_CASE_ROUTING	COMMENT_TXT	N
DLP_CASE_ROUTING	COMMENT_TXT_J	N
DLP_CASE_STUDY	STUDY_DESC	N
DLP_CASE_STUDY	STUDY_DESC_J	N
LM_LABELED_TERMS	NOTES	N
LM_LABELED_TERMS	NOTES_J	N
LM_PRODUCT	PROD_GENERIC_NAME	N
LM_PRODUCT	PROD_GENERIC_NAME_J	N
RPT_ROUTING	COMMENT_TXT	N

To enable this feature:

- For the columns available in the table ETL_SPLIT_CLOB_COLUMN_MAP as part of the factory data, update the column value for ENABLYE_YN as **Y** for the required column. For example:

```
UPDATE ETL_SPLIT_CLOB_COLUMN_MAP
   SET ENABLYE_YN = 'Y'
WHERE TABLE_NAME = '<TABLE_NAME>'
   AND COLUMN_NAME = '<COLUMN_NAME>'
```

- For new columns that are not available in the table ETL_SPLIT_CLOB_COLUMN_MAP as part of the factory data, create new entry in this table with the column value for ENABLYE_YN as **Y** for the required column. For example:

```
INSERT INTO ETL_SPLIT_CLOB_COLUMN_MAP
       (TABLE_NAME , COLUMN_NAME , ENABLYE_YN)
VALUES ('<TABLE_NAME>, '<COLUMN_NAME>', 'Y');
COMMIT;
```


Copyright © 2019, 2026, Oracle and/or its affiliates.

> **Note**
>
> Enabling this feature only affects the data that is fetched from the Oracle Argus Source to Oracle Argus Mart in the Next ETL. It does not affect the existing data in the MART. (It does not trigger the re-load of data for the respective CLOB columns.)

To disable this feature:

Set the column value for ENABLYE_YN as **N** for the required column.

```
For the example:
UPDATE ETL_SPLIT_CLOB_COLUMN_MAP
   SET ENABLYE_YN = 'N'
WHERE TABLE_NAME = '<TABLE_NAME>'
   AND COLUMN_NAME = '<COLUMN_NAME>'
```

> **Note**
>
> Disabling this feature only affects the data that is fetched from the Oracle Argus Source to Oracle Argus Mart in the Next ETL. It does not affects the existing data in the MART. (It does not trigger the re-load of data for the respective CLOB columns.)

Copyright © 2019, 2026, Oracle and/or its affiliates.