Software Design Specification

GCIE – ci360-udm-db-loader

Revision 2.0

Table of Contents

Revision History

Version	Name	Reason For Changes	Date
0.5	*Prasad Paranjpe, Pranav Gade, Nikita Pawar*	*Initial Revision*	
1.0	*Pranav Gade*	*Updated*	*02/05/2025*

Reviewed By

Name	Department	Date
Roel Van Assche	*Global CI Delivery*	*02/05/2025*

1. Introduction

1.1 Purpose

This document contains the design specifications for the ci360-udm-db-loader Code Generator utility developed by the Global CI PSD Delivery Team.

1.2 Overview

ci360-udm-db-loader Code Generator is a utility which helps the implementation teams and customers to load data sets from Unified Data Model (UDM) for SAS Customer Intelligence 360 into third party databases. It is an enhancement to a similar utility "SAS Customer Intelligence 360 CDM Loader – SAS" developed by SAS R&D and provides support for multiple target third party databases and has ability to generate updated static code to incorporate the changes made to the UDM schema in SAS Customer Intelligence 360.

The static code generated by this utility can be used to upload the CDM/UDM data downloaded from 360 using "SAS Customer Intelligence 360 Download Client – SAS" utility provided by SAS R&D.

1.3 Definitions and Acronyms

360 – SAS Customer Intelligence 360
CDM – SAS Customer Intelligence 360 Common Data Model
UDM – SAS Customer Intelligence 360 Unified Data Model
Target DB – Third-party database where downloaded UDM data from 360 is to be uploaded (updated).

2. Design Considerations

All design considerations were handled in Binder Release Phase 1.

2.1 Assumptions and Constraints

This document assumes knowledge about UDM Data Download process and basic database and SAS programming knowledge. It is intended to be used by SAS developer who need to modify the utility for specific customer project needs.

This is not a user guide for the utility. A separate document is available in case you want to use the utility in your 360 implementations.

2.2 System Environment

This utility is developed using SAS Programing language and hence it assumes that you have access to a SAS Programming environment (Base SAS or Enterprise Guide or SAS Studio) where you can view and edit the SAS Code and Datasets. The utility also assumes that you have access to any one of the supported third-party databases which holds the target UDM data mart.

This utility also assumes that you have access to tenant secret and access key for your 360 Tenant and you have a mechanism in place to periodically download the UDM data from this tenant into a specific location using the "SAS Customer Intelligence 360 Download Client – SAS" utility provided by SAS R&D.

2.3 Design Methodology

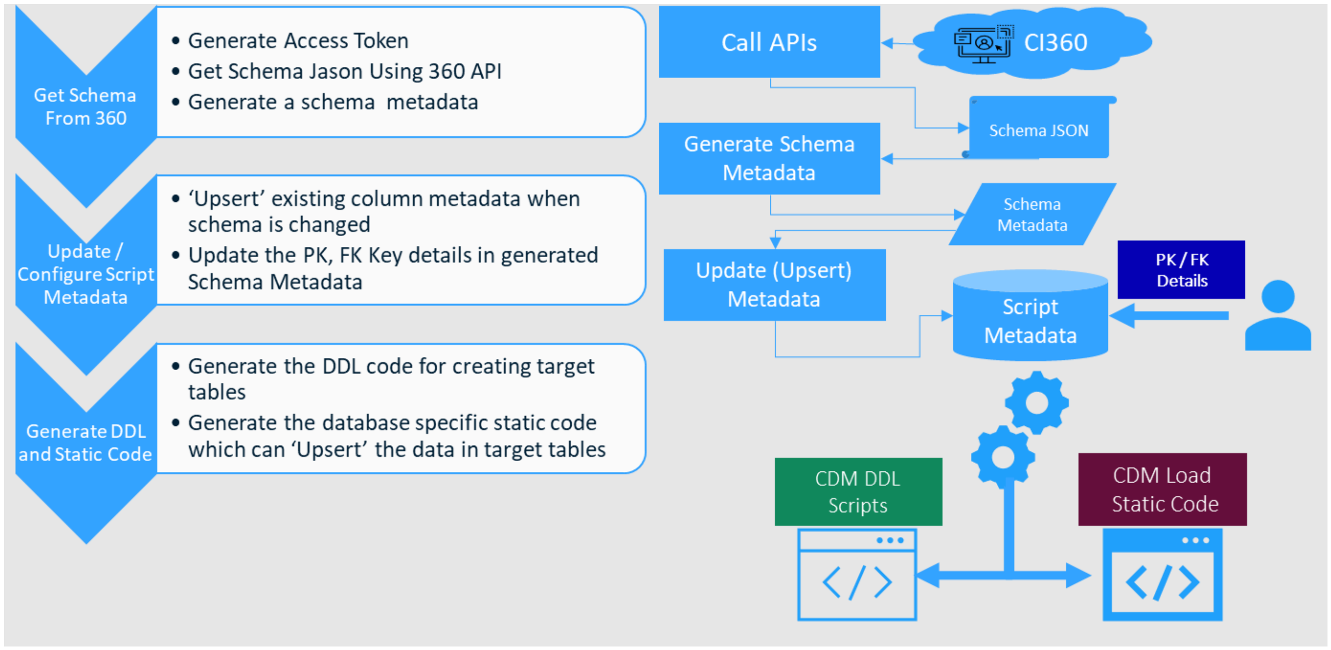
The utility design is created to adopt to the changes to the CI360 UDM schema. It relies on the schema information provided by 360 via available public APIs. This information is then used to create / update the utility metadata. The static code is generated based on this metadata which can be used to schedule periodic updates to the target UDM data mart.

2.4 Dependencies and Risks

1. APIs to download the UDM schema details in Json format
2. Access to the 360 tenant
3. Access to Base SAS
4. Preconfigured SAS Customer Intelligence 360 Download Client – SAS for downloading the UDM data from 360
5. Any changes in the format of the UDM schema metadata provided by 360 APIs may require the utility code modifications
6. The PK relationships in the Target schema need to be manually updated as per the target database requirements.

3. Architecture

3.1 Overview – Process and Data Flow


Get Schema From 360
• Generate Access Token
• Get Schema Jason Using 360 API
• Generate a schema metadata
Update / Configure Script Metadata
• 'Upsert' existing column metadata when schema is changed
• Update the PK, FK Key details in generated Schema Metadata
Generate DDL and Static Code
• Generate the DDL code for creating target tables
• Generate the database specific static code which can 'Upsert' the data in target tables
Call APIs
CI360
Schema JSON
Generate Schema Metadata
Schema Metadata
PK / FK Details
Update (Upsert) Metadata
Script Metadata
CDM DDL Scripts
CDM Load Static Code

3.2 Modules

3.2.1 Utility Startup Script

SAS Program - 'udmloader.sas'

SAS EG should be used for code generation process with below parameters.

- GENERATEMETADATA: Create base metadata using CI360 API to fetch the metadata from CI360
- CREATEDDL: Create DDL for specific database as defined in the utility configuration
- CREATEETLCODE: Generate Database Specific ETL code which can be used to schedule UDM Data Load into target DB

This macro needs to be called via command line to use this utility. A sample command line is as seen below:

```
sas -sysin udmloader.sas -sysparam EXECUTEDDL
```

Command line parameter which can have following values:

- EXECUTEDDL : Run the generated database specific DDL to create target tables
- LOADDATA : Run the generated database specific static code generated by this utility

Based on the command line parameter value provided by the user this macro will execute other macros (modules) to provide the relevant functionality.

3.2.2 Fetch and update UDM Schema Metadata 360 using API

SAS Macro - generate_base_metadata_tbl

This SAS macro is invoked when the command line parameter value is "GENERATEMETADATA". This macro uses get_udm_structure_from_api macro to invoke 360 API to download the schema details for UDM tables in JSON format and then generates SAS Metadata tables.

The get_udm_structure_from_api macro uses get_authentication_token macro to generate the authentication token required for invoking the 360 REST API to fetch the Schema Metadata JSON. The get_authentication_token macro uses 'DSC_TENANT_ID' and 'DSC_SECRET_KEY' global variables defined in "config.sas" file to generate the access token for CI360.

This generated access token is then used to access the 360 API to fetch the schema JSON and the data from the JSON response is stored in a SAS Table.

3.2.3 Generate SAS Code

SAS Macro : generate_code

This macro is used for generating database specific DDL code as well as static ETL code

3.2.3.1 Generate Database Specific DDL Code

Macro generate_code with command line parameter 'CREATEDDL' and 'database' parameter set in 'config.sas' program.

This macro internally uses the 'TABLE_LIST' metadata dataset in the CDMCNFG library to generate the DDL code for the target tables for which the execution_flag field is set to 'Y'. This macro executes generate_table_code macro with table name as the parameter to generate DDL code for that table. It refers to metadata table ('METADATA_TABLE') in CDMCNFG library to pick up the details about the table's columns, their data types and primary key columns relationships. It also refers to DATATYPES metadata table in CDMCNFG library to map the base CI360 datatypes to database specific datatypes to generate the database specific DDLs. This code is then appended to a database specific static code file found in <DDL code> directory

3.2.3.2 Generate Database Specific Load Data Static ETL code

Macro generate_code with command line parameter 'CREATEETLCODE' and 'database' parameter set in 'config.sas' program

This macro internally uses the 'TABLE_LIST' metadata dataset in the CDMCNFG library to generate the static code for the target tables for which the execution_flag field is set to 'Y'. This macro executes generate_table_code macro with table name as the parameter to generate table specific load data code. It refers to metadata table ('METADATA_TABLE") in CDMCNFG library to pick up the details about the table's columns, their data types and primary key columns. It also refers to DATATYPES metadata table in CDMCNFG library to map the base CI360 datatypes to database specific datatypes to generate the database specific code. This code is then appended to a database specific static code file found in <static code> directory

3.2.4 DDL code generating Macros

SAS Macros : create_<database_name>_ddl

This set of macros are used to generate database specific DDL code for the target database. The 'create_macro_var' macro dynamically calls these macros for a specific target database based on the value of 'dbname' parameter defined in config.sas file.

3.2.5 Static ETL code generating Macros

SAS Macros : create_<database_name>_ code

This set of macros are used to generate database specific static ETL code that can be used to load CDM data into target database tables. The 'create_macro_var' macro dynamically calls these macros for a specific target database based on the value of 'dbname' parameter defined in config.sas file.

3.2.6 Run DDL Code to Create Target Tables.

SAS Macro : execute_ddl

This macro internally uses the static DDL code generated by this utility to create target tables. If you want to add tables to this code the static code needs to be regenerated by enabling the table in the 'TABLE_LIST' metadata table. The static DDL code generated is for a specific database as set in the database parameter in the config.sas file.

3.2.7 Run Static ETL Code to Load Data

SAS Macro : load_data

This macro internally uses the static ETL code generated by this utility to run incremental data load for the tables that are enabled in the 'TABLE_LIST' metadata table found in CDMCNFG library. If you want to add tables to this code the static code needs to be regenerated by enabling the table in the 'TABLE_LIST' metadata table. The static code generated is for a specific database as set in the database parameter in the config.sas file.

4. Metadata Tables

4.1 METADATA_TABLE

This table contains persistent metadata used by this utility. The metadata is created when the utility is run with GENERATEMETADATA command for first time this metadata information will be collated from UDM schema details. Whenever SAS provides new UDM Schema version you need to generate fresh metadata using same GENERATEMETADATA command. Manual inputs may be required to define information about Primary key relationships and NULL/NOT NULL constraints for each table.

The table contains following columns:

Column Name	**Values / Description**	**Additional Notes**
default_val	Default Value for the column data	This could be any literal string or SAS Function or Macro call e.g. CI360, datetime()
isnull	**T** - Null Values allowed in Target Table **S** – Null Values allowed in input tables **<Blank>** – Null values not allowed	**Possible values:** T S TS <BLANK> This needs to be manually defined
ispk	**T** – Column is part of Primary Key Target Table **S** – Column is part of Primary Key input tables **<Blank>** – Column is not part of Primary Key	**Possible values:** T S TS <BLANK> This needs to be manually defined
isfk	Column has Foreign key reference.	**Possible values:** T S TS <BLANK> This needs to be manually defined
fk_reference	Foreign Key reference into other table	This needs to be manually defined
table_name	Name of the Target Table	Pre-populated data from schema details json
column_name	Name of this column	Pre-populated data from schema details json
data_type	Datatype from Schema JSON from 360	No Manual Modification Required
data_length	Data Length from Schema JSON from 360	No Manual Modification Required
column_type	Combination of Datatype and Data Length	No Manual Modification Required
Staging_table	Temporary staging table name used by the script	This can be the table name suffixed by '_tmp'. Cannot be more than 32 characters.

4.2 TABLE_LIST

This table defines the tables which should be considered for DDL and ETL code generation.

Column Name	Values / Description	Additional Notes
table_name	Name of the Input Table	This value is populated in first run of the utility with UPDATEMETADATA command line parameter
execution_flag	If the table should be considered for code generation	Default value is 'N'. This needs to be set to 'Y' for the tables that need to be considered for DDL and ETL code generation
mart_type	Mart type of the Input Table	This value is populated in first run of the utility with UPDATEMETADATA command line parameter

4.3 DATATYPES

This table contains the data type mappings for all the supported target databases by this utility. In case you need to add support for additional databases you must add the datatype mappings here.

Column Name	Values / Description	Additional Notes
rdbms	Name of the Target Database	Oracle, Teradata, DB2, Redshift, MSSQL, Azure
schema_datatype	Datatype from 360 Schema API JSON	char date decimal int timestamp varchar
rdbms_datatype	Database Specific Data type	Defined for all the supported databases

5. Configuration Parameters

Following configuration parameters are used by all the macros in this utility and need to be set as variables in config.sas in the file {UtilityLocation}/config folder.

1. TENANT DETAILS

   Define variables for Tenant

```
%let TENANT_ID=tenant id value;

%let SECRET_KEY=secret key value;
```

%letExternal_gateway=https://<external gateway host>/marketingGateway;

2. SCHEMA DETAILS

   Define variables for Schema version.

   %let SCHEMA_VERSION=6;

3. DATABASE DETAILS

   Define the third-party database name and the credentials to access the database.

   %let dbname=;

   Defines the supported database name. Targeted supported databases for this utility are Oracle, RedShift, Teradata, DS2, MSsql and Azure.

   %let dbpath=;

   Defines the database service name (for example CMDM) which is used by the system to connect to database. E.g. TNS Names entry for Oracle database.

   %let dbschema=;

   Defines the target database schema name

   %let dbuser=;

   Defines the target database users name

   %let dbpass=" ";

   Defines the target database password

4. DATETIME FORMAT

   %let format=datetime27.6;

   User may change the format based on their requirement

5. BULKLOAD OPTIONS

   Defines the bulkload threshold option

   %LET DB_BL_THRESHOLD=100000;

```
%let DB_BL_OPTS= %str(BULKLOAD=YES);

%let DB_LD_OPTS =%str(INSERTBUFF=32767 DBCOMMIT=0);

%let cdmcodes_path=;
```

Location where generated static code and DDL is saved

```
%let udmmart=;
```

Location for downloaded tables

```
%let UtilityLocation=/<Location where utility is saved > /;
```