m_utl_create_ddl.sas File Reference

Utilities

Utility macro to create a Data Definition Language DDL script

---

Description

This macro obtains all column, index and constraint information from a given table to be used to generate a Data Definition Language (DDL) file. The DDL file contains a SAS proc sql step to create a new instance of the given source table.

Note

This macro is able to generate DDL files to create tables for engine type of SAS or Oracle. Other engine types may be added in the future.

Autors

Paul Alexander Canals y Trocha (paul.canals@gmail.com)

Date

2020-09-23 00:00:00

Version

20.1.09

Link

https://github.com/paul-canals/toolbox

Parameters

Input	help	Parameter, if set (Help or ?) to print the Help information in the log. In all other cases this parameter should be left out from the macro call.
Input	libnm	Parameter to specify the SAS library reference. The default value for LIBNM is: _NONE.
Input	tblnm	Parameter to specify the SAS dataset or table. The default value for TBLNM is: _NONE_.
Input	creds	String containing the Oracle database credentials information. The following order of parameters is to be respected for the macro to work properly: user=, password=, path=
Input	ddl_file	Specifies the full path and name of the DDL file where the formatted output is written to. If the file that you specify does not exist, then it will be created for you. The default value for DDL_FILE is: _NONE_.
Input	prm_flg	Boolean [Y|N] parameter to specify wether the output DDL contains the identical libref, table name, and credentials as the input source table or to parameterize this information by using SAS macro variables instead. If PRM_FLG=Y, the libref, table name and optional credentials are replaced by the PRM_LIB, PRM_TBL, and PRM_CREDS values. The default value is: N.
Input	prm_lib	Optional. Parameter to specify the output DDL SAS library reference name or the SAS macro variable name value (e.g. &libref.).
Input	prm_tbl	Optional. Parameter to specify the output DDL SAS dataset or table name or the SAS macro variable name value (e.g. &table.).
Input	prm_creds	Optional. String containing the output DDL database credentials or SAS encryption key information. The following list of arguments is to be respected to work properly: Engine = ORA: user=, password=, path= Engine = SAS: encrypt=aes encryptkey= Alternatively a SAS macro variable name value can be used (e.g. &creds.).
Input	print	Boolean [Y|N] parameter to generate the output by using proc report steps with style HtmlBlue. The default value for PRINT is: N.
Input	debug	Boolean [Y|N] parameter to provide verbose mode information. The default value is: N.

Returns

- Returns all column index information on a given table.

Calls

- m_utl_chk_table_exist.sas
- m_utl_create_dir.sas
- m_utl_get_sashelp.sas
- m_utl_get_tbl_columns.sas
- m_utl_get_tbl_constraints.sas
- m_utl_get_tbl_indexes.sas
- m_utl_hash_lookup.sas
- m_utl_nlobs.sas
- m_utl_print_message.sas
- m_utl_print_mtrace.sas
- m_utl_printto.sas

Usage

Example 1: Show help information:

```
%m_utl_create_ddl(?)
```

For the next examples create a table with a couple of indexes:

```
proc sql noprint feedback stimer;
   CREATE TABLE WORK.BANKKONTO (
      MANDANT_ID           VARCHAR(8)     label='Mandant Identifier'
    , PARTNER_ID           NUMERIC(8)     label='Geschäftspartner Identifier'
    , KONTO_ID             NUMERIC(8)     label='Konto Identifier'
    , KONTO_WAEHRUNG_CD    VARCHAR(3)     label='Geschäftswährung'
    , KONTO_TYPE_CD        VARCHAR(32)    label='Konto Type Code'
    , SALDO_AMT            NUMERIC(8)     format=19.2 label='Saldo Betrag'
    , LOAD_DTTM            NUMERIC(20)    format=datetime20. label='Ladezeitstempel'
    , CONSTRAINT PRIM_KEY PRIMARY KEY (MANDANT_ID, KONTO_ID, LOAD_DTTM)
      );
   CREATE UNIQUE INDEX PARTNER_ID ON WORK.BANKKONTO (PARTNER_ID);
   CREATE INDEX IDX_BANKKONTO ON WORK.BANKKONTO (KONTO_ID, LOAD_DTTM);
quit;
```

Example 2: Create a Data Definition Language file for table bankkonto:

```
%m_utl_create_ddl(
   libnm    = WORK
 , tblnm    = bankkonto
 , ddl_file = %sysfunc(getoption(WORK))/bankkonto_work.sas
 , print    = Y
 , debug    = Y
   );
```

Example 3: Create a DDL file for table bankkonto with libref parameter:

```
%m_utl_ds2ddl(
   libnm    = WORK
 , tblnm    = bankkonto
 , ddl_file = %sysfunc(getoption(WORK))/bankkonto_libref.sas
 , prm_flg  = Y
 , prm_lib  = %nrstr(&libref.)
 , print    = Y
 , debug    = Y
   );
```

Example 4: Create a DDL file for table bankkonto with creds parameter:

```
%m_utl_ds2ddl(
   libnm     = WORK
 , tblnm     = bankkonto
 , ddl_file  = %sysfunc(getoption(WORK))/bankkonto_creds.sas
 , prm_flg   = Y
 , prm_lib   = %nrstr(&libref.)
 , prm_creds = %nrstr(&creds.)
 , print     = Y
 , debug     = Y
   );
```

Copyright