m_cst_describe_view.sas File Reference

Custom

Custom macro to export code to recreate a SAS DS or SQL view

Description

This macro can be used to transfer data step or SQL type views across different operating environments. The program uses the SAS dictionary tables to determine the view type, which can be a SAS data step or SQL type view, and then uses the describe statement accordingly to obtain the view code to recreate it.

Note

The macro returns the source path, view name, and view code information as result in a SAS dataset, which could be ported into a XPT type file, and transferred to another SAS system.

Autors

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

Date

2023-09-11 00:00:00

Version

23.1.09

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	pathnm	Parameter to specify the the full path and SAS view name including extension (.sas7bvew). The default value for PATHNM is: _NONE.
Input	viewnm	Optional. Full qualified name <library.view> for the source SAS view. The default value for VIEWNM is: _NONE_.
Input	libref	Parameter to specify the target library name that will be used for the view creation code. The default value for LIBREF is: _TMP_.
Input	outtbl	Optional. Full qualified name <library.view> for the result output table containing the view description information.
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 a table with SAS view type information and creation code.

Calls

- None

Usage

Example 1: Show help information:

```
%m_cst_describe_view(?)
```

For the next examples create a SAS data step and a proc sql type view:

```
data WORK._dsvview / view=WORK._dsvview;
   set SASHELP.class;
   where Sex eq "F";
run;

proc sql noprint;
   create view WORK._sqlview as
   select *
     from SASHELP.class
    where Sex eq "M"
   ;
quit;

proc print data=SASHELP.vview
   (where=(upcase(libname) eq 'WORK')) label noobs;
run;
```

Example 2: Obtain SAS data step view type code description information:

```
%m_cst_describe_view(
   pathnm   = %sysfunc(getoption(WORK))/_dsvview.sas7bvew
 , libref   = TMP
 , outtbl   = WORK.dsv_p_result
 , print    = Y
 , debug    = N
   );
```

Example 3: Obtain SAS proc SQL view type code description information:

```
%m_cst_describe_view(
   pathnm   = %sysfunc(getoption(WORK))/_sqlview.sas7bvew
 , libref   = TMP
 , outtbl   = WORK.sql_p_result
 , print    = Y
 , debug    = N
   );
```

Example 4: Obtain SAS view description information from existing library:

```
%m_cst_describe_view(
   viewnm   = WORK._dsvview
 , libref   = TMP
 , outtbl   = WORK.dsv_l_result
 , print    = Y
 , debug    = N
   );
```

Example 5: Obtain SQL view description information from existing library:

```
%m_cst_describe_view(
   viewnm   = WORK._sqlview
 , libref   = TMP
 , outtbl   = WORK.sql_l_result
 , print    = Y
 , debug    = N
   );
```

Copyright