m_utl_create_index.sas File Reference

Utilities

Utility macro to create an index on a given table or dataset

Description

The macro creates a new index on a given table or dataset by using the PROC DATASETS procedure. This macro can be included inline into an existing PROC DATASETS step by setting the INLINE_FLG parameter to Y. Also there is a build in check on INDEX_COLS to verify that the index does not exist yet for the given table. If this is the case, the macro returns silently. This macro is based on the ut_create_index.sas macro program by Dave Prinsloo (dave.prinsloo@yahoo.com)

Autors

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

Date

2020-02-02 00:00:00

Version

20.1.02

Link

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

Parameters

Input	help	Parameter, if set (or ?) to print the Help information in the log. In all other cases this parameter should be left out from the macro call.
Input	table	Full LIBNAME.TABLENAME name of the or SAS dataset. The default value is: _NONE.
Input	index_name	Name of the index. If omitted the macro will set a default name for composite index, or use the value of INDEX_COLS for simple index with only one column.
Input	index_cols	List of one or more column names to be indexed separated by a blank.
Input	inline_flg	Boolean [Y|N] Parameter to declare if the macro is to be run inline inside a PROC DATASETS step. The Default value is: N.
Input	debug	Boolean [Y|N] parameter to provide verbose mode information. The default value is: N.

Returns

- New created index on table

Calls

- m_utl_chk_table_index.sas
- m_utl_print_message.sas

Usage

Example 1: Show help information:

```
%m_utl_create_index(?)
```

For the next examples create a table for the index creation:

```
data WORK.class;
   set SASHELP.class;
run;
```

Example 2: Create simple index on Name (Result: index Name created):

```
%m_utl_create_index(
   table      = WORK.class
 , index_cols = Name
 , inline_flg = N
 , debug      = Y
   )

proc print data=SASHELP.vindex;
   where libname='WORK' and memname='CLASS';
run;
```

Example 3: Create simple index on Sex (Result: index Sex created):

```
proc datasets lib=WORK nolist;
   %m_utl_create_index(
      table      = WORK.class
    , index_cols = Sex
    , inline_flg = Y
    , debug      = Y
      )
quit;

proc print data=SASHELP.vindex;
   where libname='WORK' and memname='CLASS';
run;
```

Example 4: Create simple index on Sex (Result: index not created):

```
%m_utl_create_index(
   table      = WORK.class
 , index_name = Sex
 , inline_flg = N
 , debug      = Y
   );

proc print data=SASHELP.vindex;
   where libname='WORK' and memname='CLASS';
run;
```

Example 5: Create composite index on Name Sex (Result: index idx_class created):

```
%m_utl_create_index(
   table      = WORK.class
 , index_cols = Name Sex
 , inline_flg = N
 , debug      = Y
   );

proc print data=SASHELP.vindex;
   where libname='WORK' and memname='CLASS';
run;
```

Example 6: Create composite index on Name Sex (Result: index class not created):

```
%m_utl_create_index(
   table      = WORK.class
 , index_name = class
 , index_cols = Name Sex
 , inline_flg = N
 , debug      = Y
   );

proc print data=SASHELP.vindex;
   where libname='WORK' and memname='CLASS';
run;
```

Copyright