m_utl_chk_table_index.sas File Reference

Utilities

Utility macro to check if a given index exists on a given table

---

Description

This macro works by opening a cursor on the VINDEX view subset for indexes on the passed table. This view contains a record for every variable in the index. The cursor returns a list of indexes and their indexed variables. Fortunately, this view returns records in order of INDXNAME which identifies the index name and INDXPOS which designates the order in which variables in a compound index are indexed. The program need only identify any index whose variables match those passed in the order they were passed. If the existing index contains both the correct variables and also some extras later in the sequence, that index will serve the purpose and is also considered a match. If a match is found the value of the output macro variable MVAR_MATCH is set to 1, otherwise the value is set to 0. This macro is based on a publication from 2006 called The Wealth of Information Found in the SASHELP Data Dictionary Views and how to Use It by Curtis Mack (curtis.mack@lgan.com).

Note

This macro can be used as inline code by setting the parameter GLOBAL_FLG to N. In this case both the MVAR_MATCH and MVAR_NAME variables are set to local macro variables.
MVAR_MATCH parameter returns either 1 or 0 and VAR_NAME the index name or null value depending on if a match could be found.

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 value is omitted the macro will use the INDEX_COLS to search for an index. The default value is: _NONE_.
Input	index_cols	Optional. List of column names separated by a blank character to search for an exising index. The default value is: _NONE_.
Input	global_flg	Boolean [Y|N] Parameter to specify wether the result value is to be declared as a global macro variable. If set to N, the result is only returned inline. The default value for GLOBAL is: N.
Output	mvar_match	Name of the global variable containing a boolean expression [0|1] representing the index match result. The default value is: _idx_match.
Output	mvar_name	Name of the global variable containing the index name result. The default value is: _idx_name.
Input	debug	Boolean [Y|N] parameter to provide verbose mode information. The default value is: N.

Returns

- Returns a 1 or 0 depending on index match.

Calls

- m_utl_print_message.sas

Usage

Example 1: Show help information:

```
%m_utl_chk_table_index(?)
```

For the next examples create a table with several SIMPLE and COMPOSITE indexes:

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

proc datasets library=WORK nolist;
   modify class;
   index create Name;
   index create comp_1 = (Name Sex);
   index create comp_2 = (Name Age);
   index create comp_3 = (Sex Age Name);
quit;
```

Example 2: check if a SIMPLE index exists (Result=1):

```
%let index_exist =
   %m_utl_chk_table_index(
      table      = WORK.class
    , index_name = Name
    , debug      = Y
      );
%put &=index_exist.;
```

Example 3: check if a SIMPLE index exists (Result=0):

```
%m_utl_chk_table_index(
   table      = WORK.class
 , index_cols = Sex
 , global_flg = Y
 , mvar_match = index_exist
 , debug      = Y
   );
%put &=index_exist.;
```

Example 4: check if a COMPOSITE index exists (Result=1):

```
%m_utl_chk_table_index(
   table      = WORK.class
 , index_cols = Name Sex
 , global_flg = Y
 , mvar_match = idx_match
 , mvar_name  = idx_name
 , debug      = Y
   );
%put &=idx_match.;
%put &=idx_name.;
```

Example 5: check if a COMPOSITE index exists (Result=1):

```
%m_utl_chk_table_index(
   table      = WORK.class
 , index_name = comp_2
 , global_flg = Y
 , mvar_match = idx_match
 , mvar_name  = idx_name
 , debug      = Y
   );
%put &=idx_match.;
%put &=idx_name.;
```

Example 6: check if a COMPOSITE index exists (Result=1):

```
%m_utl_chk_table_index(
   table      = WORK.class
 , index_cols = Name Age
 , global_flg = Y
 , mvar_match = idx_match
 , mvar_name  = idx_name
 , debug      = Y
   );
%put &=idx_match.;
%put &=idx_name.;
```

Example 7: check if a COMPOSITE index exists (Result=0):

```
%let index_match =
   %m_utl_chk_table_index(
      table      = WORK.class
    , index_cols = Name Sex Age
    , debug      = Y
      );
%put &=index_match.;
```

Copyright