m_utl_join_setkey.sas File Reference

Utilities

Utility macro to join two or more tables based on index values

---

Description

The macro can be used to join two or more tables by using index to perform table lookup, the process of locating observations in a table file based on values in a master file. The advantages of using an index for table lookup are that only the observations where a match occurs are read from the table, multiple values can be retrieved from the table, and the master observation is directly matched. This means that if guidelines are adhered to, access to the lookup table is made significantly faster. Possible disadvantages are that more CPU cycles and I/O operations are needed to create and maintain the index, and also that extra memory is needed to load the index pages during code processing, and to store the index on disk. The macro will set a CALL MISSING statement by default. If the DEFAULT_KEY argument is provided, the missing values will be set to the values of the default key value instead. This macro is based on the "Using KEY= to Perform Table Look-up" SAS paper from Sandra Lynn Aker (sandra.l.aker@chi.monsanto.com) and on the ut_hash_define.sas macro program from Dave Prinsloo (dave.prinsloo@yahoo.com).

Note

It is important to sort the master table records by the master key variable to avoid repeated look-ups of the identical record in the look-up tables!

Autors

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

Date

2020-03-23 00:00:00

Version

20.1.03

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	master_key	Name of the key column for the master table.
Input	mkey	Alias of the MASTER_KEY= parameter.
Input	trans_table	Full LIBNAME.TABLENAME name of the transactional table or SAS dataset, on which lookups are to be performed. This table has to contain an index on the TRANS_KEY column. The TRANS_TABLE parameter may not include contain a combination of SAS data step style statements between brackets! The default value for TRANS_TABLE is: _NONE_.
Input	table	Alias of the TRANS_TABLE= parameter.
Input	data	Alias of the TRANS_TABLE= parameter.
Input	trans_key	Name of the key column for the lookup table.
Input	key	Alias of the TRANS_KEY= parameter.
Input	key_num	An integer value to be added to the lookup table key column name to be able to distinguish the key column names when joining data from more than one lookup or transaction tables.
Input	include	Optional. Blank separated list of columns to limit the result varlist to only those columns mentioned in the include list parameter.
Input	exclude	Optional. Blank separated list of columns to limit the result varlist by excluding those columns mentioned in the exclude list parameter.
Input	rename_from	Optional. Blank separated list and ordered list of the original column names.
Input	rename_to	Optional. Blank separated list and ordered list of the new renamed column names.
Input	default_key	Optional. Specifies a default TRANS_KEY value when a lookupvalue could not be found. If this value is not set, lookup fail returns missing values for all columns.
Input	debug	Boolean [Y|N] parameter to provide verbose mode information. The default value is: N.

Returns

- Data step style key= code block for transaction table lookups.

Calls

- m_utl_chg_delimiter.sas
- m_utl_chk_table_index.sas
- m_utl_print_message.sas
- m_utl_varlist.sas

Usage

Example 1: Show help information:

```
%m_utl_join_setkey(?)
```

Example 2: Step 1 - Create a Master Table and three Lookup Tables:

```
data
   WORK.master(keep=mkey1 mkey2 mkey3 master_id master_txt ranuni)
   WORK.trans1(keep=tkey1 trans1_txt)
   WORK.trans2(keep=tkey2 trans2_txt trans2_txt2)
   WORK.trans3(keep=tkey3 trans3_txt trans3_txt2)
   ;
   * add a few rows that do not match ;
   do mkey1 = 111219 to 111221;
      master_id = mkey1;
      master_txt='nothing in trans tables' || put(master_id,z4.);
      output WORK.master;
   end;
   retain tkey1-tkey3 master_id;
   master_id = 0;
   tkey1 = 0;
   tkey2 = 0;
   tkey3 = 0;
   do loopi = 1 to 100000;
      do itrankey1 = 1 to 5;
         tkey1 = tkey1 + 1;
         trans1_txt = put(tkey1, z3.);
         output WORK.trans1;
         mkey1 = tkey1;
         do itrankey2 = 1 to 5;
            tkey2 = tkey2 + 1;
            if tkey2 > 10000 then tkey2 = 1;
            trans2_txt = '1'||put(tkey2, z3.);
            trans2_txt2 = '2'||put(tkey2, z3.);
            output WORK.trans2;
            mkey2 = tkey2;
            do itrankey3 = 1 to 5;
               tkey3 = tkey3 + 1;
               if tkey3 > 20000 then tkey3 = 1;
               trans3_txt = '1'||put(tkey3, z3.);
               trans3_txt2 = '2'||put(tkey3, z3.);
               output WORK.trans3;
               mkey3 = tkey3;
               master_id = master_id + 1;
               master_txt='mast ' || put(master_id,z4.);
               ranuni = ranuni(123);
               output WORK.master;
            end;
         end;
      end;
   end;
   * add a few rows that wont match to trans2 and trans3;
   do mkey1 = 999111997 to 999111998;
      master_id = mkey1;
      tkey1 = master_id;
      master_txt='nothing in trans tabs' || put(master_id,z4.);
      output WORK.master;
      output WORK.trans1;
   end;
run;
```

Example 2: Step 2 - Create a sorted and unsorted Master Table:

```
proc sort data=WORK.master out=WORK.master_unsort;
   by ranuni;
run;

proc sort data=WORK.master out=WORK.master_sorted;
   by mkey1 mkey2 mkey3;
run;
```

Example 2: Step 3 - Sort lookup tables and create simple indexes:

```
proc sort data=WORK.trans1 nodupkey
   out=WORK.trans1(index=(tkey1));
   by descending tkey1;
run;

proc sort data=WORK.trans2 nodupkey
   out=WORK.trans2(index=(tkey2));
   by descending tkey2;
run;

proc sort data=WORK.trans3 nodupkey
   out=WORK.trans3(index=(tkey3));
   by descending tkey3;
run;
```

Example 2: Step 4 - Join Master and Lookup tables by data step set key=:

```
%let _start = %sysfunc(datetime());

data WORK.master_keyjoin;
   set WORK.master_sorted;
   %m_utl_join_setkey(
      master_key  = mkey1
    , table       = WORK.trans1
    , key         = tkey1
    , key_num     = 1
    , default_key = 0
    , debug       = Y
      )
   %m_utl_join_setkey(
      master_key  = mkey2
    , table       = WORK.trans2
    , key         = tkey2
    , key_num     = 2
    , exclude     = trans2_txt
    , default_key = 0
    , debug       = Y
      )
   %m_utl_join_setkey(
      master_key  = mkey3
    , table       = WORK.trans3
    , key         = tkey3
    , key_num     = 3
    , rename_from = trans3_txt2
    , rename_to   = trans3_num
    , default_key = 0
    , debug       = Y
      )
   output;
run;

%let _stop = %sysfunc(datetime());
%let _time = %sysfunc(intck(SECOND, &_start., &_stop.), time10.);

data WORK.result;
   RunType="KEY Join";
   Message="Total Processing time for KEY join: &_time. (hh:mm:ss)";
run;

proc print data=WORK.result noobs;
run;
```

Example 2: Step 5 - Join Master and Lookup tables by proc sql join:

```
%let _start = %sysfunc(datetime());

proc sql noprint;
   create table WORK.mtemp_1 as
   select m.*
        , t.*
     from WORK.master m
          left join WORK.trans1 t
       on m.mkey1 = t.tkey1
          ;
   create table WORK.mtemp_2 as
   select m.*
        , t.*
     from WORK.mtemp_1 m
          left join WORK.trans2(drop=trans2_txt) t
       on m.mkey2 = t.tkey2
          ;
   create table WORK.master_sqljoin as
   select m.*
        , t.*
     from WORK.mtemp_2 m
          left join WORK.trans3(rename=(trans3_txt2=trans3_num)) t
       on m.mkey3 = t.tkey3
          ;
quit;

%let _stop = %sysfunc(datetime());
%let _time = %sysfunc(intck(SECOND, &_start., &_stop.), time10.);

data WORK.result;
   RunType="SQL Join";
   Message="Total Processing time for KEY join: &_time. (hh:mm:ss)";
run;

proc print data=WORK.result noobs;
run;
```

Copyright