AUTOMATISATION OF COMPUTER PERFORMANCE EVALUATION WITH MXG AND SAS® SOFTWARE
By Jean-Paul TONNIEAU - La Garantie Mutuelle des Fonctionnaires

ABSTRACT

In the chapter 35 of the "MERRILL'S EXPANDED GUIDE TO COMPUTER PERFORMANCE EVALUATION USING THE SAS SYSTEM", Barry MERRILL shows us a way to manage the PDB with about ten jobs. Running these jobs could cause problems in operations; How should I know which job is to be run today if some of these jobs have to run every day and some jobs, every week and other jobs, every month, and in addition, 3 of these jobs did not ran correctly the night before? How should I proceed to add new reports in the weekly PDB JOB? To answer these questions, the GMF developed a SAS procedure with 2 SAS tables and a few MACROS. This subject has been discussed during Barry MERRILL's training, in PARIS 4, 5 and 6 july 1989. The french MXG club sent this procedure to Barry MERRILL, as a gift for his coming in FRANCE.

When I was a little MXG user, with my first MXG tape over my head like a saint and a big red bible under my arm, I didn't know very well what to do with my tape, how to strengthen my SMF datas and draw colored graphics about CPE for my hierarchy.

Now that I'm taller, I know how to do that, and to avoid new MXG users from sleepless and nightmares, let me explain the way I go about to get my reports from the SMF data with only one job once a day.

CONTENTS

WHAT IS A "MXG CYCLE"
THE STEPS BARRY MERRILL ADVICES
THE QUESTIONS
THE ANSWER
The NEEDS
The IDEA
PRINCIPLE OF THE AUTOMATE
THE AUTOMATE IN PRODUCTION
INSTALL THE AUTOMATE
WHAT IS A "MXG CYCLE" :

In the chapter 35 of MERRILL's bible (the red book!), Barry MERRILL shows us a way to manage the PDB with about ten jobs.

The user's target is to get reports from the PDB each day, week or month.

The steps to do this are shown on this picture.

First, you have to collect data from the SMF log or any other sources every day.

Then, you need to compute those data to get sums, averages and all new datas build from the source data.

At the end, you write or draw your report with the result of the compute step.

It seems to be very simple on this picture, but if you try to list on a picture all the JCL Barry MERRILL advices in his chapter 35, you will get the following picture.
AUTOMATISATION OF COMPUTER PERFORMANCE EVALUATION WITH MXG AND SAS SOFTWARE
By Jean-Paul TONNIEAU - La Garantie Mutuelle des Fonctionnaires

THE STEPS BARRY MERRILL ADVICES:
THE STEPS BARRY MERRILL ADVICES:

This is what Barry MERRILL advices us to do, in the chapter 35 of his red book.

First, you run the BUILDPDB job to build the Performance Data Base. Once the current PDB is built, you may want to save this today's base in a library whose name is the day of the week. The step BUILDMAC will generate this name by an old style sas macro named _DAY.

Then, you'll run BUILDDAY to copy the current PDB in the _DAY library.

Notice that this JCL and all other JCLs of th chapter 35 (except the TRENDJCL) are not supplied on the MXG tape, but are listed in the chapter 35. I provide those procedures with the code of the AUTOMATE on the exchange tape of the French MXG Club.

Third, after building a copy of the PDB for every day of the week, you'll have to keep the PDB datas for the weeks and for the month. This can be done using the GDG technic, with the WEEKJCL and the MONTHJCL.

The only one problem we usually have, is to decide when those JCL have to be executed.

Barry MERRILL solved this problem in chapter 35 by a small SAS step which will write the good JCL at the good execution time on the internal reader.

I provided this step on the tape under the name "LANCE".

At this point of the cycle, we only made collect steps. The first time we will strengthen data is when we'll run the JCLTREND job which is provided in MXG code with this note below:

```
//JOBNAME 'JOB (ACCOUNTING)', 'TREND TEST', RESIZN=4000K
//EXEC SAS
// OPTIONS='GEN=0 M:AUTOSOURCE S:SOURCE S:SOURCE M:WORK=38000'
//POS DD DSN=MXG.PDB,DISP=OLD /* INPUT AND OUTPUT FOR ASUM.... */
//WIN DD DSN=MXG.CWP,DISP=OLD /* INPUT FOR TREND.. */
//WIN DD DSN=MXG.CWP,DISP=OLD /* INPUT FOR ASUMCICS */
//WIN DD DSN=MXG.CWP,DISP=OLD /* CURRENT INPUT: LOCATED OUTPUT */
//OL:TREN DD DSN=MXG.TREN,DISP=OLD /* BACKUP COPY OF INPUT */
//SASLIB DD DSN=MXG.SASLIB,DISP=SHR
//SOURCE DD DSN=USER20...SOURCE,DISP=SHR
//SOURCE DD DSN=USER20...SOURCE,DISP=SHR
//SYSIN DD *
//OPTIONS SOURCE = SOURCES, MACROGEN = MACRINT;
//PROC COPY
// INCO=TREND OUTC=OLDTREN
// DD DSN=MXG.SOURCIE(ASUMCICS),DISP=SHR
// DD DSN=MXG.SOURCIE(TREN),DISP=SHR
// DD DSN=MXG.SOURCIE(TREN72),DISP=SHR
// DD DSN=MXG.SOURCIE(ASUM),DISP=SHR
// DATA CPUEUY (KEEP=SYSTEM DATE FCTCPUEY);
// SET TREND, M:INTRU;
// IF DATE >= START; /* F */
// DATE=DATEPART(START); FORMAT DATE DATE7;
// IF DATE GE TODAY() S48;
// PROC PLOT;
```

494
It is suggested you use PDBTREND to build a historical trend data base and then execute this job weekly after the weekly PDB has been created.

This example includes ASUMCICS and ASUMJOBS to create the PDB.CICS and PDB.JOBSKED data sets. It would be better to build those data sets along with your BUILDPDB, and then remove those members from this JCL.

The Questions:

We have seen several jobs.
Some of them are executed every day.
others have to be executed every week.
others have to be executed every month.
And if I need a 2 weeks report, I'll have to add a new JCL
which will have to be executed every 2 weeks! ...
In fact, the problem is that several JCLs have to be maintained in production and several users are able to modify those JCL.
Which JCL to run again in case of error?
Which JCL to modify to add a weekly report?
all those questions come from one:
HOW TO MANAGE JCL CHANGES IN PRODUCTION?

The Answer:

Try to have no JCL changes!

The Needs:
We need to have only one JCL to maintain in production.
We need to have restart points in case of error in this only one job.
We need to know the execution status of each SAS procedure called by this only one job. This point will allow several programmers to add their report procedures in this job without taking care of the JCL.

The Idea:
We will collect all the SAS procedures of the MXG CYCLE in one SAS table. A SAS procedure will execute those procedures and update this SAS table each time an observation of this SAS table has been executed.
This is the idea of the AUTOMATE.
PRINCIPLE OF THE AUTOMATE:

What we want to do is

$\text{\$INCLUDE &LIBR($\&PROC$);}$

for all the observations of the _PROCS dataset. We need to keep the return code and the execution date. The MACRO named PROCS does this function.

OPTIONS DQUOTE NOSOURCE NOSOURCE2;
%MACRO PROCS(LIBR,PROC,DATE);
OPTIONS OBS=MAX;
$\text{\$INCLUDE &LIBR($\&PROC$);}$
RUN;
DATA _CURR ;
  PROC = "$\&PROC$";
  STATUS = &SYSRC;
  DATELAST = &DATE;
DATA _RCPROC ;
  SET _RCPROC_ _CURR_;
RUN;
DATA _NULL ;
  CALL SYMPUT('PROC','$\&PROC$');
  CALL EXECUTE('$\&UPDATE($\&PROC$);');
RUN;
%MEND PROCS;

in addition, we have to update the _PROCS dataset for the current observation to set the next execution date, taking care of the execution PERIOD. The MACRO named UPDATE computes the next execution date.

%MACRO UPDATE(PROC);
DATA _UPDATE ;
  SET _PROCS;Y=YEAR(DATEDEM);J=JULDATE(DATEDEM);
  IF PROC ^= "$\&PROC$" THEN DELETE;
  IF PERIOD ^= 'DAY' THEN OFFSET = 1;
  IF PERIOD ^= 'WEEK' THEN DO;OFFSET = WEEKDAY(DATEDEM);END;
  IF PERIOD ^= 'MONTH' THEN DO;OFFSET = DAY(DATEDEM);END;
  IF PERIOD ^= 'YEAR' THEN DO;
    OFFSET = J - (Y-1900)*1000);OFF=MOD(Y,4);
    IF OFF=0 THEN OFFSET=OFFSET-1;
  END;
  IF PERIOD ^= '' THEN DO;
    OFFSET = SUM(OFFSET,-1,-MODDATE);
    MODDATE = 0;
    DATEDEM = INTNX(PERIOD,DATEDEM,1);
    DATEDEM = INTNX('DAY',DATEDEM,OFFSET);
    .../...
  END;

At the end of the MACRO named UPDATE, we need to update the _PROCS dataset. The MACRO named UPDATL does this function.

DROP OFFSET OFF Y J;
DATA _NULL ;
  CALL SYMPUT('PROC','$\&PROC$');
  CALL EXECUTE('$\&UPDATL($\&PROC$);');
RUN;
%MEND UPDATE;

%MACRO UPDATL(PROC);
DATA RC ;
  SET _CURR ;
DATA CURR ;
  SET _CURR ;
  DELETE;

496
DATA _UPDAT1_
  MERGE UPDATE _RC_
DATA _PROCS_
  UPDATE _PROCS _UPDAT1_
  BY SEQ_
RUN;
%MEND UPDAT1;

We also need to know if the procedure is executed, and if it is, was it correctly and at what time of which day?
The 1st DATA _NULL_ step will look if the date is a holliday or not.

DATA _NULL_
  DATE = _DATEEXEC;
  CALL SYMPUT('DATE',DATE);
RUN;

PROC SORT DATA= HOLLI;
  BY DATEDEM;
PROC SORT DATA= _PROCS;
  BY DATEDEM;

If it is, the execution date will be changed to the next date when execution will effectively happen.
This have to be done when the next execution date comes on a week-end day or a holliday.
The week-end day is known as the number of the day of week given by the WEEKDAY SAS function.
Two values are coded in the 2 SAS old style MACROS _D1 and _D2.
We will see again those MACRO within the procedure named IMACAUTO.
In this case, you will be able to specify if you want to run the procedure or not (because the initial execution date has been modified) by coding a value for the SHFTEXEC variable.
The DATA step UPDATE2 do these functions.

DATA _UPDAT2_
  MERGE HOLLI _PROCS;
  BY DATEDEM;
  IF ((PROC = '') OR (PERIOD = '') THEN DELETE;
  IF (((SHIFT = 1) OR (SHIFT = 3)) AND (DESC = '') AND
      (SHFTEXEC = 1)) THEN DO;
    SHFTEXEC = 0;
    MODDATE = SUM(MODDATE, 1); DATEDEM = DATEDEM + 1; END;
  END;
  IF (SHIFT = 2) OR (SHIFT = 3) THEN DO;
  IF ((((WEEKDAY(DATEDEM) = _D1) OR (WEEKDAY(DATEDEM) = _D2)) AND
      (SHFTEXEC = 1)) THEN DO;
    MODDATE = SUM(MODDATE, 1); DATEDEM = DATEDEM + 1; END;
  END;
  IF (SHIFT = 2) OR (SHIFT = 3) THEN DO;
  IF ((((WEEKDAY(DATEDEM) = _D1) OR (WEEKDAY(DATEDEM) = _D2)) AND
      (SHFTEXEC = 1)) THEN DO;
    MODDATE = 0;
    MODDATE = 0;
    SHFTEXEC = 0;
    END;
  END;
  IF PERIOD = 'DAY' THEN DO;
    MODDATE = 0;
    MODDATE = 0;
    SHFTEXEC = 0;
    END;
DROP DESC;
RUN;
PROC SORT DATA= _UPDAT2_
  BY SEQ;
PROC SORT DATA= _PROCS;
  BY SEQ;
DATA _PROCS_
  UPDATE _PROCS _UPDAT2_
  BY SEQ;
DATA _PROCS_
  SET _PROCS;
PROC SORT DATA= _PROCS_
  BY GROUP;
RUN;
At the end of this step, all the observations in the _PROCS datasets have execution dates which will effectively happen. A PROC PRINT is used just to show all the MXG CYCLE before beginning the execution of the AUTOMATE.

```sas
PROC PRINT DATA= _PROCS NOOBS LABEL SPLIT='*';
   VAR PROC LIBRARY DATEDEM PERIOD DATELAST SEQ SHIFT SHFTEXEC
       MODDATE STATUS;
   BY GROUP;
RUN;
DATA _RCPROC_;
RUN;
```

The 2nd DATA _NULL_ step will call the MACRO named PROCS for each observation only if the execution date is today and if the last execution date is not today. Notice that "today" is in fact the current execution date and can be changed for recovery, by using IMACAUTO.

```sas
DATA NULL;
   PUT PAGE;
   PUT 'EXECUTION DES PROCEDURES';
   PUT '====================================';
RUN;
DATA NULL;
   SET PROCS;
   IF ((DATEDEM ^= &DATE) AND ((STATUS ^= 0) OR (DATELAST ^= DATEDEM)
       AND (SHFTEXEC ^= 1)) THEN DO;
       PUT PROC=;
       CALL SYMPUT('LIB',LIBRARY);
       CALL SYMPUT('PROC',PROC);
       CALL EXECUTE('%PROCS(&LIB, &PROC, &DATE,);');
   END;
RUN;
```

At least, the 3rd DATA _NULL_ step will call the UPDATE MACRO to update the _PROCS dataset for each executed observation.

```sas
DATA RCPROC ;
   SET _RCPROC ;
   IF DATELAST ^= . THEN DELETE;
RUN;
DATA NULL ;
   PUT PAGE;
   PUT 'MISE-À JOUR DES PROCEDURES';
   PUT '====================================';
RUN;
DATA RCPROC ;
   SET RCPROC ;
   PUT PROC=;
RUN;
DATA NULL ;
   SET _PROCS;
   IF ((DATEDEM ^= &DATE) AND (SHFTEXEC ^= 1)) THEN DO;
       PUT PROC=;
       CALL SYMPUT('PROC',PROC);
       CALL EXECUTE('%UPDATE(&PROC);');
   END;
RUN;
DATA _PROCS ;
   SET _PROCS;
DATA _PROCS ;
   SET _PROCS;
PROC SORT DATA=_PROCS_;
   BY GROUP;
RUN;
```
Now we have only one SAS procedure executed every day by only one JCL.
AUTOMATE is a program written with SAS.
AUTOMATE is used to schedule the execution of MXG-SAS programs.
AUTOMATE is made of 3 parts:
- SAS DATASET (TABLES)
- SAS source programs
- SAS/AF&FSP programs

SAS DATA SETS.
--------------

AUTOMATE uses 2 SAS data sets:
- PROCs which contains the names of programs to be executed.
- HOLIDAYS which contains the holiday dates.

PROCS
-----
the PROCs SAS DATA SET contains the variables listed below:
the variables marked with an '* ' must have a non missing value.
LIBRARY * Name of the library where the program is stored.
PROC * Name of the program to be executed.
GROUP Description field: group of programs (COLLECT LIST ...)
DATEDEM * Date when the program must be executed. (DATE7. format).
DATELAST Date when the last execution of this program occurred. This variable is provided by AUTOMATE in DATE7. format.
PERIOD Period of execution of this program.
   Period can have one of the following values
   • (missing value) execution for this day only.
   DAY - Execution every days.
   WEEK - Execution every weeks.
   MONTH - Execution every month.
   YEAR - Execution every years.
SHIFT A flag used to shift the execution in case of week-end or holidays. SHIFT can have 1 of the 4 following values
   1 - Shift the execution in case of holidays only.
   2 - Shift the execution in case of week-end only.
   3 - Shift the execution on week-end and holidays.
SEQ * Execution sequence order. execution begin at SEQ=1.
STATUS Execution return code. Set by AUTOMATE.
MODDATE Number of days the execution has been shifted. MODDATE is set by AUTOMATE.
SHFTEXEC A flag used to know if you want the program for which the execution date has been shifted, to be executed or not. It allows the execution date to be set for next execution (taking care of PERIOD) without running the program.
   2 values are allowed:
   0 - Shift the date for execution on a working day.
   1 - Shift the date without execution.

HOLIDAYS
--------
This SAS DATA SET contains 2 variables. It can also be used with PROC CALENDAR.
DATEDEM Date of the holiday in DATE7. format.
DESC Description of the HOLIDAY.

SOURCES files.
--------------
Only 2 source files are provided: IMACAUTO and AUTOMATE.

JCL CONSIDERATIONS.
---------------------
Now, we have only one SAS procedure to execute. So the JCL to run the MXG CYCLE will be as simple as the following one:
AUTOMATE JOB X, CLASS=X, USER=XXXXXXXX, PASSWORD=XXXXXXXX,
GROUP=XXXXXXXX, MSGCLASS=X, NOTIFY=XXXXXXXX

COPYRIGHT (C) 1989 BY J-P TONNEAU & J-P LE GALLUDEC (GMF)

STEP1 -- BUILD SAS MXG PDB

THIS JOB MUST BE EXECUTED EVERY DAY JUST AFTER SMFDUMP PROCESS

BUILDpdb EXEC JCLPROC, DISPG=OLD, OPTIONS='GEN=0 NOSTIMER MACRO MCOMPILE'
SYSIN DD *
OPTIONS NOSOURCE NOSOURCE2 NOMACROGEN DQUOTE;
%INCLUDE SOURCLIB(IMACAUTO AUTOMATE);
SAVEpdb DD DSN=CPM.MOIS.PDBJOUR(+1),DISP=(,CATLG,DELETE),
UNIT=TAPE, LABEL=(1,SL), DCB=(SYS1.MODEL, BLKSIZE=32760)

This JCL uses a JCL procedure (JCLPROC) cataloged in a
PROCLIB partitionned dataset.
AS you can see, the JCL calls 2 SAS procedures by INCLUDE.
IMACAUTO which is used to initiate the environment.
AUTOMATE which is the procedure.
IMACAUTO allows you to run on different sets of
PDB and SAS TABLE$ for test or for recovery.

IMACAUTO
-------

IMACAUTO contains old style SAS MACROS used to customize AUTOMATE execution.

_DATEXEC  Execution reference date in DATE7. format.
     TODAY() is the default value.
_D1  Number of the first week-end day (1=sun 2=mon ...)
     0 when 1st week-end day is a working day.
_D2  Number of the 2nd week-end day.
     for _D1 and _D2, the values are like WEEKDAY result.
_PROCS  Name of the SAS DATASET PROCS described above.
_HOLLIDAY Name of the SAS DATASET HOLLIDAY described above.

How to use IMACAUTO for test or recovery?

Using for test:

You just have to set the old style MACROS of IMACAUTO to other values
and you 'll point to an other MXG CYCLE (changing _PROCS),
 an other execution date (changing _DATEXEC).
By adding DD card in the JCL you 'll point to an other SMF base
or an other PDB for output.

In case of Simple ERROR:

If an error occurs during the execution of JCLJOB, in this case,
the return code is not 0, the DATELAST is set and the DATEDEM is
not reset. So the next execution has not been done and all the
programs below have not been executed. Keep quiet! You have one
day long to correct the error and re-run JCLJOB without any other
changes to JCL or AUTOMATE params. AUTOMATE will re-start the
execution just at the point where the error has occurred and will
continue to the end of the PROCS SAS DATA SET.

Recovery in case of error during non-working days:

In case of error, the return code is not 0, the DATELAST is set
and the DATEDEM is not reset. So the next execution has not been
done and all the programs below have not been executed. in this
case, correct JCLJOB to point on good GDG versions then modify the
_DATEEXEC value in IMACAUTO and re-run JCLJOB.
SAS AF & FSP source files
--------------------------------

In addition to AUTOMATE, I provided SAS/AF programs on the MXG club tape. Those programs allow to display and modify
- the IMACAUTO source file.
- the PROCs SAS DATA SET.
- the HOLIDAYS SAS DATA SET.

Installing AUTOMATE.

After unloading the tape on disk, you first have to customize
1) the IMACAUTO file, 2) the JCLPROC JCL file,
3) the INITAUTO jcl file, 4) the JCLJOB file.
Do not forget to catalog the JCLPROC into your PROCLIB,
Then you just have to run INITAUTO and AUTOMATE is installed.
Ask your PRODUCTION staff to run JCLJOB every day.
THAT'S ALL FOLKS!

If you have SAS AF and FSP, you can customize IMACAUTO and
the PROC and HOLIDAYS SAS DATA SETS by issuing the statement:
PROC DISPLAY C=ddname.MXG.IMACAUTO.PROGRAM;RUN;
where ddname is the filename where the source has been unloaded.

Else, you can update the PROCs and HOLIDAYS SAS DATA SET by:

DATA _UPDATE_;
INPUT @01 PROC $8.
   @10 LIBRARY $8.
   @20 DATEDEM DATE7.
   @30 PERIOD $5.
   @40 SEQ 3.
   @50 SHIFT 1.
   @52 SHFTEXEC 1.
   @54 GROUP $CHAR16. ;
CARDS;
ANALXXXX SOURCLIB 01JAN90 DAY 15 0 0 LIST ;
DATA _PROCs; UPDATE _PROCs _UPDATE_ ;RUN;
DATA _UPDATE ;
INPUT @01 DATEDEM DATE7.
   @10 DESC $CHAR16. ;
CARDS;
01JAN90 NEWYEAR DAY ;
DATA _PROCs; UPDATE _HOLLI _UPDATE_ ;RUN;

Testing a new set of programs.
--------------------------------

Just customize with other values the installation of AUTOMATE
and run the JCLJOB2 every day...

Installing AUTOMATE: JCLPROC
--------------------------------

The JCLPROC can be called MXG. It is a SAS JCL PROC with
additional DD cards.

//JCLPROC PROC ENTRY=SASLPA,OPTIONS=,SORT=50,WORK='350,10',DISPG=SHR
//SAS EXEC PGM=SENTRY,REGION=4096K,TIME=1439,
// PARM='&OPTIONS SORTLIB='SYS1.SORTLIB' SORT=&SORT'
//LIBRARY DD ....
... SAS JCL PROC ...
//FT12FO01 DD SYSOUT=*  
//***************** MXG SOURCE LIB ******************************************  
//SOURCLIB DD DSN=YOUR.MXG.SOURCLIB,DISP=SHR  
// TESTLIB DD DSN=SAS.MXG.TESTLIB,DISP=SHR  
//***************** MXG AUTOMATE TABLE ******************************************  
//TABLES DD DSN=YOUR.MXG.TABLES,DISP=OLD  
//***************** INPUT DATABASES ******************************************  
//SMF DD DSN=SHR,DSN=YOUR.LOGSMF.JOURNAL  
//***************** OUTPUT SASBASES ******************************************  
'SPIN DD UNIT=WORK,SPACE=(CYL,(1,1))  
//***************** OUTPUT PDB'S ******************************************  
PDB DD DSN=CPM.PDB,DISP=OLD /* INPUT AND OUTPUT FOR ASUM....*/ 00270005  
//SUN DD DSN=CPM.SUN,DISP=OLD  
//MON DD DSN=CPM.MON,DISP=OLD  
//TUE DD DSN=CPM.TUE,DISP=OLD  
//WED DD DSN=CPM.WED,DISP=OLD  
//THU DD DSN=CPM.THU,DISP=OLD  
//FRI DD DSN=CPM.FRI,DISP=OLD  
//SAT DD DSN=CPM.SAT,DISP=OLD  
//WEEK DD DSN=CPM.WEEK(+1),DISP=(,CATLG),UNIT=TAPE,  
// DCB=SYS1.MODEL, LABEL=EXPDT=99000  
//WEEK1 DD DSN=CPM.WEEK(0),DISP=SHR  
//WEEK2 DD DSN=CPM.WEEK(-1),DISP=SHR  
//WEEK3 DD DSN=CPM.WEEK(-2),DISP=SHR  
//WEEK4 DD DSN=CPM.WEEK(-3),DISP=SHR  
//WEEK5 DD DSN=CPM.WEEK(-4),DISP=SHR  
//CICSTRAN DD DSN=MXG.CICSTRAN,DISP=OLD /* INPUT FOR ASUMCICS */ 00290005  
//IMSTRAN DD DSN=MXG.IMSTRAN,DISP=OLD 00290005  
//OLDTREND DD DSN=MXG.OLDTREND,DISP=OLD /* CURRENT INPUT, UPDATED OUTPUT */ 00300005  
//LDTREND DD DSN=MXG.LDTREND,DISP=OLD /* BACKUP COPY OF INPUT */ 00310005  
//***************** OUTPUT GRAPH ******************************************  
//MXGGGRAPH DD DSN=CPM.MXG.GRAPH,DISP=,DISPG  
//***************** WORKING FILES ******************************************  
//INSTREAM DD UNIT=SYSDA,SPACE=(TRK,(1,1)),DISP=(,DELETE)  

Installing AUTOMATE: INITAUTO  

The INITAUTO calls the JCLPROC JCL to create the default PROCS and  
HOLIDAY SAS tables.  

//INITAUTO JOB X,CLASS=X,MSGCLASS=X,NOTIFY=XXXXXXXX  
//**************************************************  
//* COPYRIGHT (C) 1989 BY J-P TONNIEAU & J-P LE GALLUDEC (GMF) *  
//**************************************************  
//*  
//INITAUTO EXEC JCLPROC,  
// OPTIONS='GEN=0 NOSTIMER MACRO MCOMPILE'  
//TABLES DD DSN=CPM.TABLES,DISP=(NEW,CATLG,),  
// UNIT=XXXX, VOL=SER=XXXXXX, SPACE=(CYL,(1,1))  
//WEEK DD DUMMY  
//WEEK1 DD DUMMY  
//WEEK2 DD DUMMY  
//WEEK3 DD DUMMY  
//WEEK4 DD DUMMY  
//WEEK5 DD DUMMY  
//INSTREAM DD DUMMY  
//SYSIN DD *  
// TIONS NOSOURCE NOSOURCE2;  
//INCLUDE SOURCLIB(IMACUTO);  
DATA PROCS;  
LENGTH SHFTEXEC 2. SHIFT 2. MODDATE 2. STATUS 5. GROUP S10.  
LIBRARY S8. SEQ 8.  
FORMAT DATEDEM DATE7. DATELAST DATE7.;  
INFORMAT DATEDEM DATE. DATELAST DATE.;  
LABEL
DATEDEM = 'Date Demande*Execution'
DATELAST = 'Date Derniere*Execution'
GROUP = 'Groupe de*Procedure'
LIBRARY = 'Bibliotheque'
MODDATE = 'Offset Modification*Date Execution'
PERIOD = 'Periodicite'
PROC = 'Procedure'
SEQ = 'Numero de*Sequence'
SHFTEXEC = 'Type d'*Execution'
SHIFT = 'Type de Periode'
STATUS = 'Code Retour'

INPUT
SHFTEXEC 2. SHIFT 2. MODDATE 2. STATUS 5. GROUP $10.
LIBRARY $8. SEQ 8.

RUN;

Installing AUTOMATE: INITAUTO
---------------------------------

The INITAUTO builds the MXG CYCLE with this INPUT STREAM:

CARDS;
0 0 0 . BUILDPDB .. BUILDPDBDAY SOURCLIB 1
0 0 0 . BUILDMAC .. BUILDMACDAY SOURCLIB 2
0 0 0 . COPY .. BUILDDAYDAY SOURCLIB 3
0 0 0 . COPY .. WEEKBld WEEK SOURCLIB 4
0 0 0 . COPY .. MONTHBldMONTHSOURCLIB 5
0 0 0 . CUMUL .. SAVETRNDWEEK SOURCLIB 6
0 0 0 . CUMUL .. ASUMCICSWEEK SOURCLIB 7
0 0 0 . CUMUL .. ASUMJOBSWEEK SOURCLIB 8
0 0 0 . CUMUL .. TRNDCICSWEEK SOURCLIB 9
0 0 0 . CUMUL .. TRNJOBSWEEK SOURCLIB 10
0 0 0 . CUMUL .. TRNDRMIWEEK SOURCLIB 11
0 0 0 . CUMUL .. TRND72 WEEK SOURCLIB 12
0 0 0 . CUMUL .. TRNDRPTWEEK SOURCLIB 13
0 0 0 . SAVE .. SAVEPDB DAY SOURCLIB 14
0 0 0 . LIST .. ANALDAILYDAY SOURCLIB 15
0 0 0 . LIST .. ANALCICSDAY SOURCLIB 16
0 0 0 . LIST .. ANALRMFIDAY SOURCLIB 17
0 0 0 . GRAPH .. GRAFCICSDAY SOURCLIB 18
0 0 0 . GRAPH .. GRAFRMFIDAY SOURCLIB 19
0 0 0 . GRAPH .. GRAFRNDRWEEK SOURCLIB 20

RUN;

Installing AUTOMATE: INITAUTO
---------------------------------

The INITAUTO resets the execution dates for tomorrow:

DATA _PROCS (KEEP =
SHFTEXEC SHIFT MODDATE STATUS GROUP
DATELAST DATEDEM PROC PERIOD
LIBRARY SEQ)
SET _PROCS;
DATEDEM = TODAY() + 1;
TODAY = TODAY();
IF PERIOD = 'WEEK' THEN DO;
    DD = WEEKDAY(TODAY);
    DD = 8 - DD;
    DATEDEM = DATEDEM + DD;
END;
IF PERIOD = 'MONTH' THEN DO;
    YY = YEAR(TODAY);
    MM = MONTH(TODAY);
    MM = MM + 1;
    IF MM > 12 THEN DO;
        MM = MM - 12;
        YY = YY + 1;
    END;
    DATEDEM = MDY(MM,1,YY);
END;
IF PERIOD = 'YEAR' THEN DO;
    YY = YEAR(TODAY);
    YY = YY + 1;
    DATEDEM = MDY(1,1,YY);
END;
RUN;

Installing AUTOMATE: INITAUTO
-----------------------------

The INITAUTO creates the HOLIDAY table with no holiday dates by default.

DATA _HOLL1;
LENGTH DATEDEM 8. DESC $30.;
FORMAT DATEDEM DATE7. DESC SCHAR30.;
INFORMAT DATEDEM DATE.;
INPUT DATEDEM 8. DESC $30.;
LABEL DATEDEM = 'Date*Jour Ferie'
DESC = 'Description'
;
CARDS;
;
RUN;
PROC SORT DATA= PROCS;
   BY GROUP SEQ;
PROC PRINT DATA= PROCS NOOBS LABEL SPLIT='*';
   VAR PROC LIBRARY DATEDEM PERIOD DATELAST SEQ SHIFT SHFTEXEC MODDATE STATUS;
   BY GROUP;
PROC SORT DATA= PROCS;
   BY SEQ;
RUN;

at least, the PROCS table is printed and sorted by sequence numbers.