

Using SAS to Help Manage ROSCOE Members and PDS Files

Clarence Wm. Jackson
City of Dallas, TX

I run SAS on an IBM MVS mainframe machine using a variety of text editors to submit SAS jobs, one of which is Computer Associates' Roscoe. Over the years, I have written a lot in Roscoe, and the number of members I have created under several programmer names had grown to over 2000. No way would I be able to manage this, but at the time, there was no need. Then came the Year 2000 situation, and I needed to make space for doing the testing. And moving around as I do with the City, it was becoming increasingly difficult to copy the members from one programmer ID to another, so that I had my favorite SAS jobs on hand.

I recently accepted another position, and I needed a way to export my current 600+ CA Roscoe members to a PDS file as PDS members, and didn't want to go thru and do the Roscoe copy and export commands for each member. As I got into this process, I found that there were other Roscoe members under my other ID's that I also needed to move for later release of the Roscoe space.

Using SAS, I was able to use the Roscoe member directory information, and RPF (Roscoe Program Facility) to build a RPF file that I imported back into Roscoe and executed. This worked so well, that I was able to export all the members in about 5 minutes, and pass the code on to others needing to do some Roscoe member management. The steps to get it done follows:

1. Get the names of the Roscoe members to be exported into a flat file or PDS member using the Attach Library Roscoe command, then use the Copy Library command to put the list into your AWS:

```
ATT LIB  
COPY LIB *
```

2. Delete the first 3 lines of the library copy, then use the Roscoe Export command to copy the library list to an external file or partitioned data set:

```
EXP DSN=FLATFILE(OR_PDS)
```

3. Read the exported file into SAS, reading thru the list of member names and build 2 RPF statements for each member. The first statement is to Fetch the member, and the second statement will Export the member into a PDS by the same name as the Roscoe member being exported.

```
F membername  
EXP DSN=EXPORT.TO.PDS(membername)
```

where the membername is the Roscoe member. In this scheme, the PDS member and Roscoe member will have the same name. This makes it easier to find the members later.

3. Save and submit your SAS job. Remember to have a flat file ready to receive the SAS output, which will be two cards per member, plus 2 extra cards.

4. In Roscoe, import the flat file containing the results of the SAS job, and EXECute the RPF you just created by bringing it up into your AWS, and submitting the EXEC RPFname command in Roscoe.

Things to do before you start:

1. Make sure you have a PDS, and it is large enough to hold all members. The normal size of Roscoe partitions is about 60,000 lines at 80 bytes each line, although some partitions can be twice that size.

2. Allocate enough directory entries, one per Roscoe member.

The following SAS program was used to build this file, then import the file back into Roscoe. You'll need to copy your Roscoe lib to a file, then EXEC the program below. You should change it per your needs. This program works if the member name is the first variable in the saved directory member.

```
FILENAME INSTUFF "CLJACKS.PDSLIB(ROSMEMS)" SHR;  
FILENAME EXP2PDS "TAX7.JACKSON" LRECL=80 BLKSIZE=23440 RECFM=FB SHR;
```

```
DATA _NULL_;  
LENGTH MEMNAM $12. MEMNAM1 $14. MEMNAM2 $12.;  
INFILE INSTUFF END=ALLDONE;  
INPUT MEMNAM @8 MEMPFX $8.;  
MEMNAM1='(!!TRIM(LEFT(MEMPFX))!!)';  
MEMNAM2=TRIM(LEFT(MEMNAM));  
FILE EXP2PDS;  
IF _N_=1 THEN PUT @1 '<<EXPDS>>';  
PUT @1 'EXP DSN=CLJACKS.PDSLIB' MEMNAM1 ' INPUT=' MEMNAM2;  
IF ALLDONE THEN PUT @1 'RETURN';  
DELETE;  
RETURN;  
RUN;  
  
ENDSAS;
```

Roscoe to PDS Updates

Now, after doing this a few times, you really don't want to export each and every member to a

PDS, so how do you just export only those you have stored yet? Use SAS PROC SOURCE to get a list of your PDS members, compare it to your Roscoe member list, then make the RPF just for those members. I used the following program to do just that. The Temporary file will contain the list of PDS member names.

```
OPTIONS='MPRINT DQUOTE ';

FILENAME PDSLIST "&&TEMPLIST" NEW UNIT=SYSDA
  SPACE=(TRK,(15,1),RLSE)
  DCB=(LRECL=80,BLKSIZE=23440,RECFM=FB);

FILENAME EXP2PDS "TAX7.JACKSON" LRECL=80 BLKSIZE=23440 RECFM=FB SHR;

FILENAME PDSINFO "CLJACKS.PDSLIB" SHR;

PROC SOURCE DIRDD=PDSLIST INDD=PDSINFO NODATA NOPRINT;
RUN;

FILENAME PDSINFO CLEAR;

DATA PDSLIST;
  INFILE PDSLIST;
  INPUT @1 MEMNAME $8.;
RUN;

PROC SORT;
  BY MEMNAME;
RUN;

FILENAME INSTUFF "CLJACKS.PDSLIB(ROSMEMS)" SHR;

DATA ROSMEM;
  INFILE INSTUFF;
  INPUT @8 MEMNAME $8.;
RUN;

PROC SORT;
  BY MEMNAME;
RUN;

FILENAME INSTUFF CLEAR;

DATA MISSTHEM;
  MERGE ROSMEM (IN=A)
    PDSLIST (IN=B)
  END=ALLDONE;
  BY MEMNAME;
  IF A AND NOT B;
  FILE EXP2PDS;
  IF _N_=1 THEN PUT @1 '<<EXP2PDS>>';
```

```

MEMNAM2=('()TRIM(LEFT(MEMNAME))))');
PUT @1 'F' MEMNAME $8.
  /@1 'EXP DSN=CLJACKS.PDSLIB' MEMNAM2;
IF ALLDONE THEN PUT @1 'RETURN';
RUN;

PROC PRINT;
RUN;

ENDSAS;

```

So How do I Know What My Roscoe Members Are?

I use SAS to get a listing of what's on my Roscoe partition because I can group the data however I need. One of fun things about Roscoe is the ability to 'stack' Roscoe members to make one logical program or job. This is one way to have parts of your code in different members, and use only what you need. To submit jobs in multiple members, use the following command style:

```
SUB mem1 mem2 mem...
```

Roscoe will submit all of the jobs to MVS, and MVS will read each one as the same job until it reads the '/' in column 1-2. So using this scheme, you can submit SAS statements from one Roscoe member, datacards in another Roscoe member, and the rest of the SAS statements in a third Roscoe member. With this in mind, I setup three Roscoe members with the 1st member containing the SAS statements to create a SAS DataSet from the Roscoe Library copied and saved to the 2nd member, and the 3rd member will create a listing by various sorts, including date last accessed.

The first member will look something like this:

```

//CJROSCOE JOB (DS07,XXXXXXXX),CJAC.4BS.CLASS=S
//S1 EXEC SAS,HOLD=NO
//SYSIN DD *

DATA;
INFILE CARDS;
INPUT @1 RECS $72. @8 JOBNAME $8.
      @18 DESC $30. @50 CREATE $CHAR8. @56 YEARMAD $2.
      @70 ACCESSE $CHAR8. @60 UPDATE $CHAR8.;
IF ACCESSE='08/11/97' /* TO FIND STUFF I USED THAT DAY */
THEN LIST;
CREATED=INPUT(PUT(CREATE,$8.),MMDDYY8.);
ACCESSED=INPUT(PUT(ACCESSE,$8.),MMDDYY8.);
UPDATED=INPUT(PUT(UPDATE,$8.),MMDDYY8.);

LABEL RECS="ROSCOE*MEMBER*INFO"
      ACCESSED="LAST*ACCESSED"

```

```

UPDATED="LAST*UPDATED"
DESC="COMMENTS*ABOUT*MEMBER"
JOBNAME="ROSCOE*MEMBER*NAME"
CREATED="DATE *CREATED";
CARDS;

```

The second member will be the lines following the CARDS statement in the job stream (these are also the cards used in earlier programs for the Roscoe member list:

QA6.BILAC	08/28/97	08/29/97	08/29/97
QA6.BIRTHCOM	09/02/97	09/02/97	09/03/97
QA6.BIRTHDEP	08/06/97	08/06/97	08/14/97
QA6.BIRTHDIV	08/06/97	08/06/97	09/02/97
QA6.BIRTHEQU	09/02/97	09/02/97	09/02/97
QA6.BIRTH911	09/02/97	09/02/97	09/02/97
QA6.BUDGET	09/08/97	09/09/97	09/09/97

The third and last member will process the card input, and create various reports and listings:

```

*****

```

```

TITLE "CITY OF DALLAS, ECI (OPERATIONS AND PLANS) DEPT, PRODUCTION CONTROL
DIVISION";
FOOTNOTE "CLARENCE JACKSON >> ROSCOE DIRECTORY << &SYSJOBID";

```

```

PROC FREQ;
TABLES YEARMAD;
RUN;

```

```

PROC SORT;
BY ACCESSED UPDATED CREATED JOBNAME;
RUN;

```

```

PROC PRINT N D NOOBS SPLIT="";
ID JOBNAME;
VAR DESC ACCESSED UPDATED CREATED;
FORMAT ACCESSED UPDATED CREATED MMDDYY10.;
RUN;

```

```

ENDSAS;
//

```

Using these programs, I'm in more control of what's on my Roscoe members, and have built a way to backup the Roscoe members to PDS files. I can then take those PDS members and put to tape, download to PC file format for use in my SAS Windows95 sessions, etc. I now found that I don't have to have multiple versions of SAS programs in Roscoe as I can keep them on PDS files, which reduces my Roscoe partition overhead. I can now manage what's out on my PDS

files, and my Roscoe files using SAS.

Clarence Wm. Jackson
City of Dallas
214-670-4876

email contact:
cljacks@ci.dallas.tx.us (work)
Cjac@compuserve.com (home)