MANAGING SAS DATA SETS
A DATABASE MANAGEMENT PERSPECTIVE
STEPHEN CLOS, ISSC
CAROLINE BAHLER, GIV

ABSTRACT

Permanent SAS® data sets can be used as a crucial part of the overall “database” environment. As a crucial part of the “database” environment, constant manual maintenance must be performed on the SAS data sets to optimize DASD usage and to prevent production job job abends due to out-of-space conditions. This presentation will discuss the various out-of-space conditions which can occur and the routines which can be developed to avoid these situations. In addition, this paper discusses how to use information from PROC CONTENTS, SAS Dictionary Tables using PROC SQL and a SAS routine which reads the Volume Table Of Contents (VTOC) of a DASD unit as part of a maintenance evaluation process. While this paper is directed toward the use of SAS on a host mainframe (our site is running a MVS operation system), with little or no modifications the same routines discussed could operate on other mainframe operating systems. Development of an evaluation process is the most important step in automating SAS data set maintenance freeing System Administrators to pursue other Database Management activities.

Introduction

The use of permanent SAS data sets can be a crucial part of an overall “database” environment. As a crucial part of the “database” environment, constant manual maintenance must be performed on the SAS data sets to optimize DASD usage and to prevent production job abends due to out-of-space conditions. SAS datasets on a single volume will expand to secondary extents as the dataset continues to grow. Since records are logically deleted in a SAS dataset, routine maintenance must be performed to clear the deleted records and reduce the allocation of the SAS dataset in order to optimize the DASD usage.

Out-of-space conditions can occur within single or multi-volume allocated data sets and can be triggered by multiple scenarios. For example, within a MVS mainframe system, a SAS data set has used all of the space allocated to that data set. This will generally only be a problem with data sets which have been allocated across multiple DASD units since these data sets will be unable to take secondary extents. Another problem which can be encountered within a MVS system is when a data set takes secondary extents but reaches the system maximum limit of 16. The last out-of-space condition occurs when a data set needs to take a secondary extent but the DASD volume does not have enough free space to allocate the secondary extent.

System Administrators need to develop processes to flag or automatically perform “reorganization” routines on data sets to prevent any or all of the out-of-space conditions which can occur. These routines need to evaluate SAS data set information and the Volume Table Of Contents (VTOC) of a DASD volume. These routines will analyze the data and based upon conclusions, the process will dictate which actions should be taken. This will ensure that data sets stay available to applications without running out-of-space or unable to take secondary extents. The objective of this paper is to discuss the data set attributes which need to be captured, the data set evaluation process, and the automation of the maintenance routines.

DATA COLLECTION

Data collection is the first step in the evaluation process. The following meta-data needs to be captured about each SAS data set:

- Host path name (the mainframe data set name);
- Number of Observations;
- Number of Deleted Observations;
- Total Library Blocks;
- Total Used Blocks;
- Total Free Blocks; and
- Number of Extents Allocated.

Multiple sources are used to obtain the data needed for the evaluations. There are different products which can also be used to gather the information which is needed. Sources for this data are the information found in the PROC CONTENTS listing for a SAS data set (the printed output written to a flat file); a VTOC listing (volume table of contents) of the DASD unit where the SAS data set resides (multiple products can be used to obtain this information), and information from the SAS Data set Dictionary Tables (Fig. 1).

When the data set is a multi-volume data set then from the PROC CONTENTS printed output the Total Library Blocks, Total Used Blocks, and Total Free Blocks is collected (Table 1). This data is used to compute the percentage of free space. The number of extents the data set has taken will be collected by reading the VTOC listings. The number of observations and the number of deleted observations is collected from the SAS dictionary tables using PROC SQL.
Table 1. Host specific information from PROC CONTENTS output.

<table>
<thead>
<tr>
<th>CONTENTS PROCEDURE</th>
</tr>
</thead>
<tbody>
<tr>
<td>-----Directory----</td>
</tr>
</tbody>
</table>

Libref:            SALES
Engine:            V608
Physical Name:     PSMO.SAS.CUSTMTH.DATA
Unit:              DISK
Volume:            DBON30, DB2P22, DB2P00
Disposition:       SHR
Device:            3390
Blocksize:         27548
Blocks per Track:  2
Total Library Blocks: 66000
Total Used Blocks:  137879
Total Free Blocks:  -71879
Highest Used Block: 143752
Highest Formatted Block: 143752
Members:           25

DATA EVALUATION
There are at least three evaluations that need to be performed.

- The first is to determine the number of deleted records in all members of a data set.
- The second evaluation is to determine the percentage of free blocks in a data set on the host mainframe. This will mainly be used for the multi-volume data sets since they are unable to take secondary extents. A data set allocated on a single volume is able to take secondary extents and the number of free blocks can vary.
- The third evaluation is to determine the number of extents a data set has taken and if that data set is in danger of causing an abend due to the number of extents. Although we do want data sets to be allocated in multiple extents, this must be monitored so that we don't reach the maximum limit of 16.

Using the output from PROC SQL, total records will be evaluated to determine the average number of records loaded per time frame in the data structure. The average number of records will be determined by dividing the number of records by the number of loads. The deleted records will be divided by the average number of records to determine how many load frames of deleted records exist in the data structure. If the number of load frames equal or exceed (a base number to be established), a "reorg" of the file will take place.

The percentage of free space is calculated and compared to a pre-established criteria. If those conditions meet or exceed the base criteria then a reorganization of the data set will take place.

The program will examine the number of extents in the VTOC listing for SAS data sets and if a data set is allocated in more than 6 extents, an alert message will be issued to the DBA staff. The DBA staff will then reallocate the data set making any changes needed to the primary or secondary space parameters.

Based upon the above criteria, a reorg of the file will be completed in order to purge the deleted records so that the space can be reused for new inserts. Based upon the percentage of free blocks, a reorg of a data set will be scheduled, or if a data set is in more than 6 extents a re-allocation of the data set will be performed. The application process will notify the DBA staff that a reallocation of the data set is required.

The reorg will consist of a run which will allocate a new data set on work volumes during off-peak hours. The job will copy the data, clear the permanent SAS data set, and copy the data back in, thereby dropping the deleted records. The temporary file allocated for the reorg will be deleted at the end of the run. The reason for not allocating the dataset on a new volume once and copying the data only there is that we want the SAS datasets to be resident on specific volumes which can be monitored. This helps data management in the management of the datasets and the DASD utilization at our site. Multi-Volume data sets will not be automatically "reorganized", but instead will issue an alert message to the DBA staff.

CONCLUSION
Permanent SAS data sets updated and used as a crucial part of the overall "database" environment require constant evaluation and maintenance to prevent production abends. We utilize SAS datasets by storing most of our historical data for sales information. Our datasets can grow beyond the
sizings based on sales growth or due to acquisitions which introduce new history on customers even though the work may not have been completed by our organization. We report history back to the end customer of the sales they did regardless of the company they were doing business with during a period of time. Developing an evaluation process is the most important step in automating SAS data set maintenance. Data availability is crucial not only to the applications but also to our users. When the data becomes unavailable to the applications or to our customers, this creates a hit to our service level agreements and to our customer satisfaction rating.

This routine also allows for the DBA staff to be freed for other Database Management activities.

References
(SELECT MAX(DATE_LOADED) FROM TSYS.TRASDSNS WHERE DSN = A.DSN)
ORDER BY DSN)
AS SASTUFF(DSN, SPACALOC, SCNDALOC, EXTENTS, NBRVOLS, BLKSIZE,
VOLSER, FREETRKS);
QUIT;
*PUT &SQLXMSG;

DATA _NULL_
SET SASTUFF END=EOF;
IF EOF THEN CALL SYMPUT('N', _._);
RUN;

*MACRO INFO;
*DO I= 1 %TO EN;

DATA _NULL_
SET SASTUFF;
IF _N_ = &I;
CALL SYMPUT('NAME', DSN);
RUN;

LISTINFO "NAME" DISP=SHR;

PROC SQL;
CREATE TABLE TOTOBJS AS
SELECT PATH AS DSN, SUM(NOBS-DELOBS) AS NOBS, SUM(DELOBS) AS DELOBS
FROM DICTIONARY.MEMBERS A,
DICTIONARY.TABLES B
WHERE PATH = "NAME" AND A.MEMNAME = B.MEMNAME;
QUIT;

DATA _NULL_
SET TOTOBJS;
WHERE DSN='';
FILE TEMP NOTITLES MOD;
PUT &001 DSN $44.
 &050 NOBS
 &070 DELOBS;
RUN;
*END;
*END;
*INFO;

DATA READTOT/V=>READTOT;
INFILE TEMP;
INPUT &001 DSN $44.
 &050 NOBS
 &070 DELOBS;
*SORTSUM(READTOT,DSN,NOBS DELOBS,SORTOUT);

DATA PRTRLINE(DROP=NOBS DELOBS);
MERGE SASTUFF SORTTOT;
BY DSN;
IF NOBS=0 OR DELOBS = 0 THEN FRENCH=0;
ELSE FRENCH = DELOBS / NOBS * 100;

PROC PRINT NOOOBS DATA=PRTRLINE SPLIT='*';
WHERE EXTENTS > 3 OR PRNT GE 20 OR SCNDALOC > FREETRKS;
TITLE 'SAS DATASETS WHICH NEED TO BE REORGED';
LABEL DSN = 'DATASET NAME'
SPACALOC = 'FRM*ALLOC'
SCNDALOC = 'SCNDRY*ALLOC'
NBRVOLS = 'NBR*VOLS'
VOLSER = 'VOLSER'
FREETRKS = 'FREETRKS*ON VOL'
FRNT = 'OF*DELOBS';
RUN;
/*
Figure 2. CLIST that fixes the secondary allocation information extracted to reflect DASD allocation in tracks.

```
PROC O DSN
  CONTROL NOLIST NOSYMLIST NOCONLIST NOMSG
  /* CONTROL LIST SYMLIST CONLIST MSG
  ISPEXEC CONTROL ERRORS RETURN
BEGIN: +
FREE F(INPUT1 OUTPUT1)
ALLOC DD(INPUT1) DS('DEV.L.SASR001S.DATA') SHR
ALLOC DD(OUTPUT1) DS('DEV.L.SASR001S.TEMP') SHR
SET &SPC4 EQ &STR(  )
ERROR +
  DO
    ERROR OFF
    GOTO FINIS
  END
OPENFILE INPUT1 INPUT1
OPENFILE OUTPUT1 OUTPUT1
LOCF: +
GETFILE INPUT1
SET OUTPUT1 EQ INPUT1
IF &SUBSTR(86:88,4INPUT1) EQ &STR(BLK) THEN +
  DO
    SET &TRKS EQ &SUBSTR(54:57,4INPUT1)
    SET &TRKS EQ &TRKS / 2
    SET &LEN EQ &LENGTH(&TRKS)
    SET &LEN EQ 4 - &LEN
    IF &LEN GT 0 THEN +
      SET &TRKS EQ &STR(&SUBSTR(1:4LEN,4SPC4)&TRKS)
      SET OUTPUT1 EQ +
      &STR(&SUBSTR(1:53,4INPUT1)&TRKS.&SUBSTR(58:85,4INPUT1))
    END
  END
IF &SUBSTR(86:88,4INPUT1) EQ &STR(CYL) THEN +
  DO
    SET &TRKS EQ &SUBSTR(54:57,4INPUT1)
    SET &TRKS EQ &TRKS * 15
    SET &LEN EQ &LENGTH(&TRKS)
    SET &LEN EQ 4 - &LEN
    IF &LEN GT 0 THEN +
      SET &TRKS EQ &STR(&SUBSTR(1:4LEN,4SPC4)&TRKS)
      SET OUTPUT1 EQ +
      &STR(&SUBSTR(1:53,4INPUT1)&TRKS.&SUBSTR(58:85,4INPUT1))
    END
PUTFILE OUTPUT1
GOTO LOOP
FINIS: +
CLOSEFILE INPUT1
CLOSEFILE OUTPUT1
FREE F(INPUT1 OUTPUT1)
DEL 'DEV.L.SASR001S.HOLD'
REN 'DEV.L.SASR001S.DATA' 'DEV.L.SASR001S.HOLD'
REN 'DEV.L.SASR001S.TEMP' 'DEV.L.SASR001S.DATA'
REN 'DEV.L.SASR001S.HOLD' 'DEV.L.SASR001S.TEMP'
EXIT
```

Fig. 3. CLIST that determine free space available to SAS dataset

```
PROC O DSN
  CONTROL NOLIST NOSYMLIST NOCONLIST NOMSG
  /* CONTROL LIST SYMLIST CONLIST MSG
  ISPEXEC CONTROL ERRORS RETURN
BEGIN: +
  ISPEXEC LMINIT DATAD0(SAS) DDFNAME(VTOC) ORG(PS)
  ISPEXEC EDIT DATA1(DAS) MACRO(EIGHT_FIX)
FINIS: +
  ISPEXEC LFREE DATA1(DAS)
EXIT
```