All Zipped Up and Nowhere to Go

Louise Hadden and Jim McIntosh
Abt Associates Inc., Cambridge MA

ABSTRACT

Computer programmers and analysts are frequently presented with disk space constraints, as our sources of data have become removable media. Cartridges, tapes and CD-ROMS can contain far more data than there is space available on the computer platform being utilized. This paper illustrates one method for dealing with the space quandary many programmers face on a daily basis. Most of the examples presented are specific to base SAS running on a UNIX operating system; however, with some modification, some of these techniques can be used with SAS for personal computers and mainframes. In the process of completing the task on which the examples presented are based, UNIX SAS versions 6.11 and 6.12 were used.

BACKGROUND

Abt Associates Inc. performed a contract to provide special analytic services to a pharmaceutical company. Our client was interested in developing a knowledge base about the market for a specific product. The analyses to be carried out included a longitudinal study of seven years worth of data from the Nationwide Inpatient Sample of the Healthcare Cost and Utilization Project (HCUP), among other data sources.

The HCUP data was provided in CD-ROM format. Five years of data (1988-1992) were presented as quarterly files of ASCII data (each quarter on a single CD-ROM containing up to 650 MB), while the 1993 and 1994 data appeared as several files containing selected variables for each year. The analysis design required selection of records (hospital discharges) by scanning fifteen diagnosis codes and fifteen procedure codes for particular diseases and outcomes, assigning records to designated market groups, and producing tabulations by market group for final presentation. The tabulations of discharges were to be weighted up to the national level, using weights carried on a separate file for each year. The weights were at the hospital level; therefore the weight files were comparatively small.

Our UNIX computing environment, which is a cost recovery system within our company, is heavily oversubscribed. The system contains 18 disk drives ranging from 2 GB to 9GB in capacity, with 47 file systems, each capable of holding 1-2 gigabytes of data for a total capacity of around 72 gigabytes of available storage, 4 8mm tape drives, 1 3480 cartridge drive, 1 9-track tape drive and 1 CD-ROM drive.

It was not possible to copy the HCUP data straight from the CD-ROMS to the UNIX file systems because of limited space availability and because the UNIX CD-ROM drive was not compatible with the format of the older HCUP CD-ROMS (1988-1992).

I. "I THINK I NEED A BIGGER SIZE . . ."

The first issue we addressed was the problem of CD-ROM incompatibility. We also found that the early CD-ROM format was incompatible with the HP CD recorder we have available. We had to find a CD reader that could read the older format, and at the same time, it seemed useful to compress the data in some fashion so that it was not necessary to mount multiple CD-ROMS for each year of data. We turned to the Info-ZIP utility, a freeware package written by Greg Roelofs, Onno van der Linden, Jean-loup Gailly, John Bush, Harald Denker, Hunter Goatley, Chris Herborth, Jonathan Hudson, Paul Kienitz, Johnny Lee, Steve P. Miller, Sergio Monesi, Kai Uwe Rommel, Steve Salisbury, Christian Spieler, Antoine Verheijen and Mike White, compatible with virtually any personal computer, mainframe and UNIX operating system in countries all over the world, and readily available on the Internet. In the U.S., Info-ZIP's home site is http://www.cdrom.com and in Switzerland, Info-ZIP's web site is sunsite.cnlab-switch.ch. Using INFOZIP for PC-DOS, we compressed the quarterly CD-ROMs into 4 zip files for each year. The "zipped" files were then ported over to a PC with a CD-ROM writer using the TCP/IP FTP protocol and a single CD-ROM, compatible with the UNIX CD-ROM drive and containing 3-4 zipped files, was produced for each year.

II. "HOW AM I SUPPOSED TO GET THIS ON?"

Once the HCUP data were on compatible CD-ROMS, we faced the challenge of reading in all seven years of data onto our RS/6000, extracting the desired discharge records. Always exploring efficient data processing methods, we had previously determined that using a pipe command in the filename was an excellent method of processing large amounts of data on 8mm cartridges, either cartridge to disk, or cartridge to cartridge. For example, to read a fixed-block file with a record length of 120 (that does not contain any record terminators/line feeds) from a tape or cartridge device using SAS (assuming that the tape or cartridge has already been positioned):

```
filename dat pipe 'tctl -nf /dev/rmt2.3 -b 32760 -p
32760 read';

data out.xxx;
        infile dat lrecl=120 recfm=f;
        input (normal SAS input specifications);
run;
```

It is also possible to position tapes or cartridges from within SAS using the CALL SYSTEM routine:

```
data _null_;
        call system('tctl -f /dev/rmt2.3 fsf 7');
run;
```

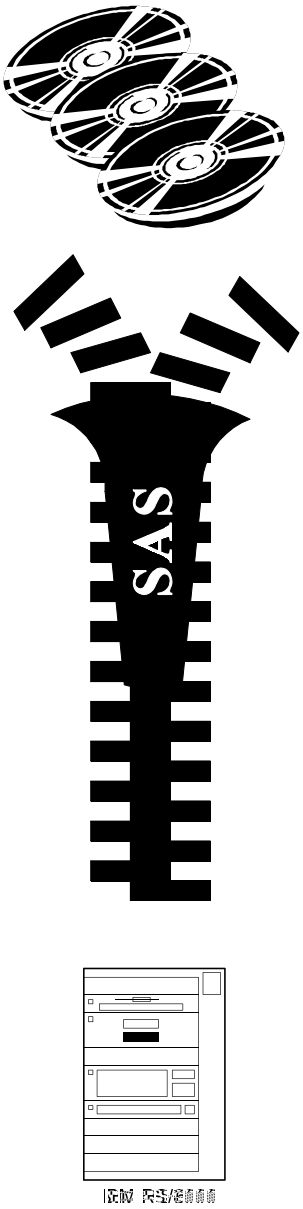
A SAS input statement with a trailing @ makes selection of records possible without actually reading (and writing) every entire record, thus minimizing the amount of available space needed for processing.

```
input @1 selvar $ebcdic2. @;
        if selvar ne '11' then delete;
        input @3 var2 s370ff1.
                .
                .
                .
        ;
run;
```

Using this same technique on zipped files on a CD-ROM required a small amount

of research, but proved both possible and efficient. This technique works equally well on UNIX compressed data files, substituting *uncompress* for *unzip -a -cqq*.

```
filename zip1 pipe 'unzip -a -cqq
/cdrom/nis88q1.zip';
filename zip2 pipe 'unzip -a -cqq
/cdrom/nis88q2.zip';
filename zip3 pipe 'unzip -a -cqq
/cdrom/nis88q3.zip';
filename zip4 pipe 'unzip -a -cqq
/cdrom/nis88q4.zip';
libname dd '/e20c/pharm/data';
libname ee '/tmp1/tempspace/pharm';
run;


title1 'Pharm';
title2 'HCUP 1988';
footnote "Program:  READIN88.SAS - &sysdate";
run;

%macro readit(year,qtr,filenum);

*********************************;
*  Data step                    ;
*********************************;

DATA yr&year._q&qtr. (drop=i);
        LENGTH statements;
        INFILE zip&filenum. lrecl=388;
*** Input the variables from the ASCII file;
                INPUT variables  @;
*** select based on constructed variables;
                if prim=1 or second=1 or
                badout=1;
                INPUT more variables;
                more SAS code
LABEL variables;
run;

*** create permanent file prior to merge so that if
sort fails, file is maintained ***;
*** use proc datasets to remove preliminary
quarterly files ***;

proc datasets library=dd;
   append base=hcup19&yr data=yr&yr._q&qtr;
   delete yr&yr._q&qtr.;
run;

%mend;

%readit(88,1,1);
%readit(88,2,2);
%readit(88,3,3);
%readit(88,4,4);


proc sort data=dd.hcup1988;
   by hospid;
run;

*** merge yearly file with appropriate weight file
***;
```



```
proc sort data=dd.niswt88 out=weights;
   by hospid;
run;


*** create permanent file with weights ***;

data ee.hcup1988;
   merge dd.hcup1988 (in=a) weights;
      by hospid;
   if a;
run;

*** remove preliminary permanent data set;
*** the use of 'options errorabend' will prevent
deletion of the prior permanent data set in the event
of a sort failure ***;

proc datasets library=dd;
   delete hcup1988;
run;
```

If the weight file was small enough, a format file could be created linking hospital id to the discharge weight, avoiding the cumbersome sort. Most systems seem to have a limit on how large formats can be, however, and we reached the limit on our system. We also required additional variables from the weight files, making it less efficient to process the weights in the manner.

An example of creating a format from a file follows.

```
data hospfmt;
   set dd.niswt88;
   label=put(weight,best12.);
   rename hospid=start;
   retain fmtname 'HOSPFMT';
run;

proc format library=library cntlin=hospfmt;
run;

and while you are reading in the large data sets. . .

   discwt_u=input(put(hospid,hospfmt.),12.);
```

All HCUP SAS analytic files were written onto disk (using several different file systems due to lack of available space) due to the necessity of sorting and merging with weight files for each year, but could also have been written out to an 8mm cartridge using the tctl pipe.

```
libname out tape '/dev/rmt1';
libname in '/e2/area';
run;

data out.yyy;
        set in.xxx;
        other SAS statements...
run;

        OR
proc copy in=in out=out;
        select yyy;
run;
```

However, processing from the CD-ROM drive and onto an 8mm cartridge drive each added processing steps and CPU time to the total. Along with the financial constraints on the project, there were also time constraints.

Each year's extract resulted in a file of 660 MB, well over half the size of many of our file systems. These files were compressed using the UNIX compress command while not in use, ready for subsequent analysis. Compressed files can be read using the CALL SYSTEM routine within SAS, and then recompressed in the same way.

```
data _null_;
        call system('uncompress
/e2/area/hcup1988.ssd01');
run;
```

We opted not to use SAS-based compression of the data sets, since we had experienced a mysterious incompatibility with our older UNIX file systems and SAS version 6.11 compression. We have since

upgraded both the file systems and SAS (to version 6.12) which has resolved the incompatibility.

III. A PERFECT FIT!

We contemplated creating a longitudinal file on a single 8mm cartridge, but the different structures of the 1988-1992 files versus the 1993 and 1994 files made this a difficult proposition. Such a longitudinal file could then have been processed directly from an 8mm cartridge using the tctl pipe within a SAS program.

```
libname tapelib tape '/dev/rmt1';
libname out '/e2/area';
run;

proc means data=tapelib.xxx;
title3 'Means on Data Set XXX';
run;
```

Since the files were on disk, albeit compressed, all analysis runs were performed in batch mode (using the UNIX at command), processing at night to reduce expenses and ensure timely completion of jobs.

```
at 23:59
uncompress /tmp1/tempspace/hcup1988.ssd01
sas anal8801
compress /tmp1/tempspace/hcup1988.ssd01
        .
        .
<cntl-D>
```

Since our system was overutilized, and there were often as many as ten SAS jobs running concurrently during the day, jobs were more likely to complete successfully at night. Because of space constraints (both in the SASUSER temporary directory and on each file system) all 7 years worth of analysis runs were done sequentially.

We also considered fficiency within each analytical SAS program important. Most reports consisted of summary information at the "market group" level, often with additional specifications. Sorting such large files was a daunting proposition, and since each record could be present in more than one market group, a minimal work data set containing only desired variables was created for each run, and then a PROC MEANS with an appropriate "WHERE" statement was performed, outputting a single record summary data set for each market group. The desired end product was a Lotus 1-2-3 spreadsheet with a line for each market group. A transport data set for each year and analysis run was created using output data sets from the means procedures, then FTPed to a personal computer, and transformed within the operating system there.

```
options ps=55 ls=170 errorabend errors=0
obs=max;

libname dd '/e20c/data';
libname library '/e2/area/codes';
libname trn sasv5xpt '/e4/data/anal8810.trn';
run;

data temp (keep=medicare los dischg totchg
discwt_u under65 proc);
        set dd.hcup1988 (keep=medicare los
                dischg totchg discwt_u dx1
                age cardiovs pr1 pr2 pr3 pr4
                pr5 pr6 pr7 pr8 pr9 pr10
                pr11 pr12 pr13 pr14 pr15
                where=(cardiovs=1));

        under65=(age lt 65);
        label under65='FLAG:  Age < 65';

        array prs {15} $ pr1 pr2 pr3 pr4 pr5
                pr6 pr7 pr8 pr9 pr10 pr11
                pr12 pr13 pr14 pr15;
```

```
do i=1 to 15;
        if substr(prs{i},1,2) in('35','36') then
        do;
                proc=prs{i};
                label proc='Procedure';
                output;
        end;
end;
run;

proc sort data=temp;
        by under65 proc;
run;

proc means data=temp noprint;
        var los dischg totchg;
        by under65 proc;
        weight discwt_u;
        output out=sumproc sum=sumlos
                sumdis sumchg;
run;

data sums;
        set sumproc;
        by under65 proc;
                avglos=sumlos/sumdis;
                avgchg_d=sumchg/sumdis;
                avgchg_l=sumchg/sumlos;

label
        sumdis='Total Discharges'
        sumlos='Total # of Hospital Days'
        sumchg='Total Charges'
        avglos='Average LOS'
        avgchg_d='Avg Chg Per Dischg'
        avgchg_l='Avg Chg Per Day';
 format sumchg dollar18.2 avgchg_d avgchg_l
dollar13.2 avglos comma15.2 sumlos sumdis
comma15.;
run;

proc print data=sums label uniform;
         var sumdis sumlos avglos sumchg
avgchg_d avgchg_l;
        by under65;
        id proc;
format proc $proccdn.;
title3 'Analysis of CV Procedure Groupings';
title4 'By Age Group';
run;

proc copy in=work out=trn;
        select sums;
run;
```

CONCLUSION

The SAS system provides many useful tools and interfaces for reading ‘difficult’ external files. The use of unnamed pipes in SAS running on UNIX operating systems in conjunction with such tools as **Info-ZIP** and the AIX UNIX **compress** and **tctl** commands allows the processing of data sets that otherwise would not ‘fit’ on the system.

REFERENCES

http://www.cdrom.com/pub/infozip/Info-ZIP.html

HCUP documentation, 1988-1994

SAS Companion for the UNIX environment: Language, Version 6

ACKNOWLEDGMENTS

The authors wish to thank our former colleague Ron Boheim and the many authors of Info-ZIP, particularly Greg Roelofs who was most helpful and maintains the Info-ZIP web page in the U.S.

AUTHOR CONTACT

Louise Hadden
Abt Associates Inc., HSRE
55 Wheeler St.
Cambridge, MA 02138
louise_hadden@abtassoc.com

KEYWORDS

SAS, UNIX, Info-ZIP, CALL SYSTEM, PIPE, TRAILING @, PROC DATASETS,

PROC MEANS, COMPRESS