Paper 297

Untangling and Reformatting NT PerfMon Data to Load a UNIX® SAS® Database With a Software-Intelligent Data-Adaptive Application

Heather McDowell, Wisconsin Electric Power Co., Milwaukee, WI
LeRoy Bessler, Ph.D., Strong Smart Systems, Bessler Consulting and Research, Fox Point, WI

Abstract:
Input to the application is a concatenation of NT Performance Monitor data sets from several servers. Each server can present a different record layout, due to, e.g., differing number of processors, logical disks, paging files, etc. Rather than using a separate record type for each category of resource (processor, disk, etc.), all measurement data elements for a single time interval are strung out by NT PerfMon in one long observation. The collection of datetimestamped observations for a server is preceded by a header record of what are, in effect, textual column headings. The data set is like a collection of unloaded spreadsheets, where layout differs from server to server, and where layout for a single server can change from day to day. The target database has separate tables for each resource category (CPU, Logical Disk, Paging, etc.), each keyed by server name, resource ID (to distinguish processors from one another, etc.) and datetimestamp. The Software-Intelligent application solution uses Base SAS language and macro processing.

Data is useless if it is not presented in an informative, understandable, and accessible format. Transforming data into the right format for your targeted audience is sometimes quite a challenge. Performance data collected through the PerfMon utility of Windows NT presents that challenge to SAS* users.

The PerfMon utility has the ability to collect several hundred different statistics regarding the performance of the Windows NT servers. The data it produces consists of one heading record for the file, followed by one detail record for each time of collection. This flat file is useless if left in its original format. Through the use of SAS, however, the data can be untangled and transformed into some very useful information. This paper will show how, using base SAS and SAS macro language, one can go from the tangled mess of the flat file to a more useful and user-friendly format.

One must first get beyond the overwhelming, tangled big picture presented by the data and recognize the characteristics of the original data that will help in developing a program that will parse the data and enable it to be loaded into a SAS database.

A separate file of data is created on each NT server. The file consists of one heading record followed by the detail records. The various NT servers' PerfMon files are transmitted to a single UNIX server where they are concatenated, using the UNIX 'cat' command on the SAS filename statement, to make one large file (see Appendix 2) for use in the macro. When looking at the data, one can see its inherently useful characteristics. Both the heading record and the subsequent detail records have each piece of information delimited by commas. In addition, each piece of information is also contained in quotes. The heading record always starts with "Sample Time" and each subsequent statistic heading begins with the server name. Recognizing these few characteristics is the first step in developing the program that will untangle the data.

The approach used in this case was to first separate the heading records from the detail records and proceed to take the appropriate steps to be able to create the database variables. Usually every server is different, with a different number of processors, disks, networks segments, paging files, etc. And IDs for those resources may differ from server to server. This means that each server's portion of the concatenated file presents a different set of column headings to be parsed and then used to organize the statistical measurement data. In this case, four different tables were created to hold the four different types of data that was being collected.
The first step in this process was to indicate that the file was delimited by commas using 'dlm'

on the infile statement. When the data is parsed later in the macro, it is assumed that the quotes have already been removed. Therefore, the 'dsd' option is also used to remove the double quotes from the delimited data elements. (See Appendix 1 for the NT reformat macro and its invocation, with explanatory comments). 'Truncover' was used on the infile statement to allow for the reading of variable length records. A simple check was used to then recognize the heading records from the detail records. When reading in the flat file, a record starting with "Sample Time" indicated the start of a heading record. A record starting with anything else indicated the start of the detail record. The server name, which was present in each statistic heading, had to then be separated and used as a key to be associated with all its subsequent detail statistics.

It was necessary to ascertain the number of heading records, the number of columns within those heading records, and subsequently the number of columns in their respective sets of detail records. Each server collects (potentially different kinds of) data such as CPU Usage, Disk Free Space, etc., which we temporarily call resources. Each server also can have multiple processors and disks which we temporarily refer to as resource IDs. The sequence numbering of servers and columns, and associating server number with resource and resource IDs, take place throughout the parsing and input of the heading and detail records. It is of utmost importance to make sure that every statistic is being associated with its correct heading and server.

Once everything has been parsed and associated correctly, it is merged together using the appropriate match keys of server number and column number. The merged data is then separated in to four resource-specific tables where the attributes are applied, final sorting is done and variables that are no longer needed are dropped (see Appendices 3, 4, 5, 6). The data can then be easily used and worked with to produce whatever types of tabular or graphic reports desired.

Raw data is usually pretty ugly and many times it's hard to visualize a way to transform it into something useful. The key to working with tangled raw data is to break it up into small parts and look for commonalities and useful characteristics. It is also necessary to have an idea of the output desired in order to design a good program. If the raw data and desired output can be analyzed and understood, the path needed to get from one to the other can be a smoother one.

Notices

The code here was developed and verified with SAS version 6.12.

Authors

Heather McDowell
Wisconsin Electric Power Co.
Pewaukee Data Center
W237 N1500 Busse Road
Waukesha, WI 53188
heather.mcdowell@wepco.com
414-544-7234

LeRoy Bessler, PhD
Strong Smart Systems
Bessler Consulting and Research
PO Box 96
Milwaukee, WI 53201-0096, USA
bessler@execpc.com
414-351-6748

```
%macro ntrfmt(
 outlib=,      /* Library to hold output    */
               /* tables                    */
 slctmm=,      /* 2-digit month for input   */
               /* data selection            */
 slctdd=,      /* 2-digit day for input data*/
               /* selection                 */
 colwidth=80,  /* Maximum data column width */
               /* for the input data        */
 colcount=99) ;/* Maximum expected count of */
               /* data columns for any      */
               /* server in the input data  */

filename ntperf pipe "cat
  /vol1/ntperflogs/*_&slctmm&slctdd*.csv" ;

/* ************************************** */
/* Split the input into heading records and */
/* detail records. There is one heading     */
/* record per server.  Detail for each      */
/* server follows its heading record.       */
/* Develop a server sequence number for     */
/* each record group.                       */
/* ************************************** */

data headings detail ;
  retain srvrnum 0 ;
  infile ntperf dlm=',' dsd truncover recfm=v
     lrecl=1600 ;
  input col0 :$23. (col1-col&colcount)
    (:$&colwidth..) ;
  if index(col0,'Sample Time') > 0 then do ;
    srvrnum= srvrnum+1 ;
    output headings ;
  end;
  else output detail ;
run;

/* ************************************** */
/* Count the servers by counting the heading*/
/* records.  Create the SRVCOUNT global     */
/* variable as the server count.            */
/* ************************************** */

data _null_ ;
  %global srvcount ;
  call symput('srvcount',filesize) ;
  stop;
  set headings nobs= filesize ;
run;

/* ************************************** */
/* Count the column headings in each heading*/
/* record. Extract the server name from     */
/* col1.  Test the column variables in      */
/* sequence for blanks.  The first blank    */
/* column is one column past the last valid */
/* column.                                  */
/* ************************************** */

/* Output records that contain server name  */
/* and column count                         */

data counts(keep= server count) ;
  set headings ;
  idandmor= substr(col1,3,20) ;
  idlength= index(idandmor,'\') - 1 ;
  server= substr(idandmor,1,idlength) ;
  %do i = 1 %to &colcount %by 1 ;
    if col&i = ' ' then do ;
      count= %eval(&i - 1) ;
      output ;
      return ;
    end;
%end;
run ;
```

```
/* ************************************** */
/* Create a sequenced set of paired global  */
/* variables.  Server name (SRVR) and       */
/* associated count of columns (CNUM).      */
/* ************************************** */

%global %varlist(prefix=srvr,varcount=&srvcount);
%global %varlist(prefix=cnum,varcount=&srvcount);
data _null_ ;
  set counts ;
  %do i = 1 %to &srvcount %by 1 ;
    if &i = _N_ then do ;
     call symput("srvr&i",trim(left(server)));
     call sympút("cnum&i",count) ;
    end;
  %end;
run;

/* ************************************** */
/* Take apart the headings. There are       */
/* SRVCOUNT heading records. SAS observation*/
/* number _N_ is the same as server sequence*/
/* number. Each observation (heading record)*/
/* has a known column count (CNUM).  Each   */
/* column heading has to be parsed.  Each   */
/* column heading is a fully qualified      */
/* column content descriptor. It identifies */
/* server, resource, resource ID & column   */
/* name. Resources  are things like CPU     */
/* usage, Disk Free Space, etc  Each server */
/* has  multiple processors, disks, etc     */
/* Resource ID can be Processor  Number,    */
/* Disk Name, etc. The column name is the   */
/* name of the measurement variable.  Output*/
/* records associate server_number/column-  */
/* number with server  name, resource,      */
/* resource ID and column name.             */
/* ************************************** */

data parts(keep= srvrnum colnum server resource
resid colnam) ;
  %inc attr(server) ;
  length colnum $ 2 ;
  set headings ;
  %do i = 1 %to &srvcount %by 1 ;
    if _N_ = &i then do ;
      %do j = 1 %to &&cnum&i %by 1 ;
        colnum=   "&j" ;
        toparse=  col&j ;
        begin=    substr(toparse,3,(&colwidth-
                  2)) ;
        srvrlen=  index(begin,'\') - 1 ;
        server=   substr(begin,1,srvrlen) ;
        begin=    substr(begin,srvrlen
                  +2,(&colwidth-2-srvrlen-1));
        resrclen= index(begin,'(') - 1 ;
        resource= substr(begin,1,resrclen) ;
        begin=    substr(begin,resrclen+2,
                  (&colwidth-2-srvrlen-
                  1-resrclen-1)) ;
        residlen= index(begin,')') - 1 ;
        resid=    substr(begin,1,residlen) ;
        begin=    substr(begin,residlen+3,
                  (&colwidth-2-srvrlen-
                  1-resrclen-1-residlen-2)) ;
        cnamlen=  length(begin) ;
        colnam=   substr(begin,1,cnamlen) ;
        output ;
      %end;
    end;
  %end;
run;

/* ************************************** */
/* Take apart the detail.  For each server, */
/* there are multiple detail records, one   */
/* per datetimestamp.  Each observation     */
/* contains the server sequence number as   */
/* well as the original input columns.      */
/* First construct for each observation     */
/* from its variable col0 a SAS-standard    */
/* datetime variable called ‘ts’. For each  */
/* server sequence number we know column    */
/* count from above.  With each column      */
/* variable, associate a column number      */
/* COLNUM. Assign each subsequently         */
/* numbered column variable to a new        */
/* variable COLVAL.  Output records that    */
/* associate server-number/column-number    */
/* pairs with datetimestamp and column value*/
/* **************************************** */

data content(keep= srvrnum ts colnum colval) ;
  length colnum $ 2 ;
  set detail ;
  %inc attr(ts) ;
  ts=(mdy(substr(col0,1,2),substr
     (col0,4,2),substr(col0,7,4))*86400)
     + (substr(col0,12,2)*3600) +
     (substr(col0,15,2)*60)
      + round((1 * substr(col0,18,6)),1);
  %do i = 1 %to &srvcount %by 1 ;
    if srvrnum= &i then do ;
      %do j = 1 %to &&cnum&i %by 1 ;
        colnum=   "&j" ;
        colval=   col&j ;
        output ;
      %end;
    end;
  %end;
run;

proc sort data= content out= sortc ;
  by srvrnum colnum ;
run;

proc sort data= parts   out= sortp ;
  by srvrnum colnum ;
run;

/* **************************************** */
/* Pull together all the descriptor and key */
/* data with the measurements Server        */
/* sequence number and column number are the*/
/* match keys.  Output records that         */
/* associate datetimestamp, server, resource*/
/* ID,  resource, column name, & measurement*/
/* value                                    */
/* **************************************** */

data merged ;
  merge sortp sortc ;
  by srvrnum colnum ;
run;


/* **************************************** */
/* Split the data into resource-specific    */
/* files. Drop the no longer needed resource*/
/* and column number variable. Drop any     */
/* other no longer needed variables.        */
/* **************************************** */


data cputemp (drop= srvrnum colnum colnam
              resource)
     pagtemp (drop= colnum resource)
     dsktemp (drop= colnum resource)
     nsutemp (drop= srvrnum colnum colnam
              resource) ;
set merged ;
  if resource = "Processor"       then output
       cputemp ;
  if resource = "Paging File"     then output
       pagtemp ;
  if resource = "LogicalDisk"     then output
       dsktemp ;
  if resource = "Network Segment" then output
       nsutemp
run ;
```

```
/* **************************************** */
/* Apply variable attributes (label, length,*/
/* format).  Convert to resource-specific   */
/* files.  Extract any needed extra         */
/* variables. Filter out any bad data. Drop */
/* any no longer needed variables.          */
/* **************************************** */

data &outlib..cpun (drop= resid colval) ;
  set cputemp ;
  %inc attr(proc_num) ;
  %inc attr(cpu)      ;
  %inc attr(cpuda)    ;
  %inc attr(cpudw)    ;
  %inc attr(cpuhr)    ;
  cpu= colval ;
  if cpu ^< 0 ;
  if cpu ^> 100 ;
  proc_num= resid ;
  cpuda= put(put(day(datepart(ts)),z2.),$2.) ;
  cpudw=put(put(weekday(datepart(ts)),
        _weekday.),$3.);
  cpuhr= put(put(hour(timepart(ts)),z2.),$2.);
run;

/* **************************************** */
/* Apply variable attributes (label, length,*/
/* format).  Convert to resource-specific   */
/* files. Extract any needed extra          */
/*variables. Filter out any bad data. Drop  */
/* any no longer needed variables.          */
/* **************************************** */

data &outlib..nsun (drop= resid colval) ;
  set nsutemp ;
  %inc attr(netseg)   ;
  %inc attr(nsu)      ;
  %inc attr(nsuda)    ;
  %inc attr(nsudw)    ;
  %inc attr(nsuhr)    ;
  nsu= colval ;
  if nsu ^< 0 ;
  if nsu ^> 100 ;
  if resid =: '\Device\' then
    netseg= substr(resid,9,20);
  else
    netseg= resid;
  nsuda= put(put(day(datepart(ts)),z2.),$2.) ;
  nsudw= put(put(weekday(datepart(ts)),
         _weekday.),$3.) ;
  nsuhr= put(put(hour(timepart(ts)),z2.),$2.);
run;

/* **************************************** */
/* Input to here contains mixture of        */
/* multiple kinds of observations for       */
/* different measurement variables for the  */
/* same resource.  Filter out any bad data. */
/* Split the data into variable-specific    */
/* files keyed by server sequence number,   */
/* resource ID, and datetimestamp.  Do not  */
/* keep the no longer needed column number  */
/* variable.                                */
/* **************************************** */

data pagvar1 (keep= srvrnum server resid ts
          colval rename= (colval= _pfu  ))
     pagvar2 (keep= srvrnum server resid ts
          colval rename= (colval= _pfupk)) ;
  set pagtemp ;
  if colval ^< 0 ;
  if colval ^> 100 ;
  if colnam = "% Usage" then output pagvar1 ;
  else output pagvar2 ;
run;

proc sort data= pagvar1 ;
  by srvrnum resid ts ;
run;
```

```
proc sort data= pagvar2 ;
  by srvrnum resid ts ;
run;

/* **************************************** */
/* Pull together all the measurement        */
/* variables.  Server sequence number,      */
/* resource ID, and datetimestamp are the   */
/* match keys.  Apply variable attributes   */
/* (label, length, format).  Convert to     */
/* resource-specific variables.  Filter out */
/* any unwanted data.  Filter out any bad   */
/* data.  Drop any no longer needed         */
/* variables.                               */
/*  *************************************** */

data &outlib..pagn (drop= srvrnum resid _pfu
 _pfupk) ;
  %inc attr(pagfil)   ;
  %inc attr(pfu)      ;
  %inc attr(pfupk)    ;
  merge pagvar1 pagvar2 ;
  by srvrnum resid ts ;
  where index(resid, "_Total") = 0 ;
  pagfil=   resid    ;
  pfu=      _pfu     ;
  pfupk=    _pfupk   ;
  if pfu ^> pfupk ;
run;

/* ************************************* */
/* Input to here contains mixture of     */
/* multiple kinds of observations for    */
/* different measurement variables for   */
/* the same resource.  Filter out any bad*/
/* data.  Split the data into variable-  */
/* specific keyed by server sequence     */
/* number, resource ID, and datetimestamp*/
/* Do not keep the no longer needed      */
/* column number variable.               */
/* ************************************* */

data dskvar1 (keep= srvrnum server resid ts
          colval rename= (colval= _free   ))
     dskvar2 (keep= srvrnum server resid ts
          colval rename= (colval= _free_pc))
     dskvar3 (keep= srvrnum server resid ts
          colval rename= (colval= _dskt_pc)) ;
  set dsktemp ;

if colnam = "Free Megabytes" then output dskvar1
;
  else do ;
    if colval ^< 0 ;
    if colval ^> 100 ;
    if colnam = "% Free Space" then output
dskvar2 ;
    else output dskvar3 ;
  end;
run;

proc sort data= dskvar1 ;
  by srvrnum resid ts ;
run;

proc sort data= dskvar2 ;
  by srvrnum resid ts ;
run;

proc sort data= dskvar3 ;
  by srvrnum resid ts ;
run;


/* ************************************* */
/* Pull together all the measurement     */
/* variables.  Server sequence number,   */
/* resource ID, and datetimestamp are the*/
/* match keys.  Apply variable attributes*/
/* (label, length, format)  Convert to   */
/* resource-specific variables.  Extract */
/* any needed extra variables.  Filter   */
/* out any unwanted data. Drop any no    */
/* longer needed variables.              */
/* ************************************* */

data &outlib..dskn (drop= srvrnum resid _free
     _free_pc _dskt_pc) ;
  %inc attr(dsk)      ;
  %inc attr(date)     ;
  %inc attr(time)     ;
  %inc attr(free)     ;
  %inc attr(free_pc)  ;
  %inc attr(dskt_pc)  ;
  merge dskvar1 dskvar2 dskvar3 ;
  by srvrnum resid ts ;
  where index(resid,"_Total/_Total") = 0 ;
  time=     timepart(ts) ;
  date=     datepart(ts) ;
  dsk=      resid        ;
  free=     _free        ;
  free_pc=  _free_pc     ;
  dskt_pc=  _dskt_pc     ;
run;

%mend ntrfmt ;

%ntrfmt(outlib=tehold, slctmm=05, scltdd=01)
run ;
```

Appendix 1: NTRFMT Macro and Its Invocation

```
" Sample Time","\\PEDCAA31\Processor(3)\% Processor Time","\\PEDCAA31\Processor(2)\% Processor
Time","\\PEDCAA31\Processor(1)\% Processor Time","\\PEDCAA31\Processor(0)\% Processor
Time","\\PEDCAA31\Paging File(_Total)\% Usage Peak","\\PEDCAA31\Paging File(_Total)\%
Usage","\\PEDCAA31\Paging File(\??\E:\pagefile.sys)\% Usage Peak","\\PEDCAA31\Paging
File(\??\E:\pagefile.sys)\% Usage","\\PEDCAA31\Paging File(\??\C:\pagefile.sys)\% Usage
Peak","\\PEDCAA31\Paging File(\??\C:\pagefile.sys)\% Usage","\\PEDCAA31\Network Segment(\Device\bh_CpqNF31)\%
Network utilization","\\PEDCAA31\LogicalDisk(0/E:)\% Free Space","\\PEDCAA31\LogicalDisk(0/E:)\% Disk
Time","\\PEDCAA31\LogicalDisk(0/C:)\% Free Space","\\PEDCAA31\LogicalDisk(0/C:)\% Disk
Time","\\PEDCAA31\LogicalDisk(_Total/_Total)\% Free Space","\\PEDCAA31\LogicalDisk(_Total/_Total)\% Disk
Time"
"05/04/1998
05:39:16.750","0.24208425513182474","0.34625012319350335","0.075418866233134541","0.37749988361199804","44.01
7680354683201","42.9052599862259","31.007215711805557","29.905192057291668","93.977864583333343","92.82552083
3333329","0.0012929650700929099","18.735271013354282","0","47","0","22.007409122482056","0"
"05/04/1998
05:54:16.750","0.21878777710618857","0.40802052407956646","0.17191361042470099","0.18927441289933178","44.017
680354683201","43.353456439393938","31.007215711805557","30.40771484375","93.977864583333343","93.06510416666
6671","0.0012893776887431005","18.735271013354282","0","47","0","22.007409122482056","0"
" Sample Time","\\PEDCAA32\Processor(3)\% Processor Time","\\PEDCAA32\Processor(2)\% Processor
Time","\\PEDCAA32\Processor(1)\% Processor Time","\\PEDCAA32\Processor(0)\% Processor
Time","\\PEDCAA32\Paging File(_Total)\% Usage Peak","\\PEDCAA32\Paging File(_Total)\%
Usage","\\PEDCAA32\Paging File(\??\E:\pagefile.sys)\% Usage Peak","\\PEDCAA32\Paging
File(\??\E:\pagefile.sys)\% Usage","\\PEDCAA32\Paging File(\??\C:\pagefile.sys)\% Usage
Peak","\\PEDCAA32\Paging File(\??\C:\pagefile.sys)\% Usage","\\PEDCAA32\Network Segment(\Device\bh_CpqNF31)\%
Network utilization","\\PEDCAA32\LogicalDisk(0/E:)\% Free Space","\\PEDCAA32\LogicalDisk(0/E:)\% Disk
Time","\\PEDCAA32\LogicalDisk(0/C:)\% Free Space","\\PEDCAA32\LogicalDisk(0/C:)\% Disk
Time","\\PEDCAA32\LogicalDisk(_Total/_Total)\% Free Space","\\PEDCAA32\LogicalDisk(_Total/_Total)\% Disk
Time"
"05/05/1998
09:22:36.296","0.86058670394028036","0.73906041148386059","0.91787767038402635","0.78419874868196349","36.153
796487603309","34.830191115702483","22.761027018229164","21.935696072048611","87.58203125","84.34505208333334
3","0.002228973906967521","15.480941241227466","0","54.67153284671533","0","21.697348616417738","0"
"05/05/1998
09:37:36.296","0.74077444404000437","0.64702556902650477","0.86924512461405978","0.75813534682028783","36.153
796487603309","34.877539600550968","22.761027018229164","21.963840060763889","87.58203125","84.46614583333332
9","0.00224919078880034","15.480941241227466","0","54.67153284671533","0","21.697348616417738","0"
```

Appendix 2: Sample Input Data. The line feed separating the date and time in the “Sample Time” is not a true representation. It is a result of the formatting done to include the file in this paper.

```
OBS    SERVER          TS              PROC_NUM     CPU    CPUDA    CPUDW    CPUHR
1     PEDCAA31    01MAY98:06:09:17        3         28.33    01       Fri      06
.        .            .                 .           .      .        .        .
97    PEDCAA31    01MAY98:06:09:17        2         28.33    01       Fri      06
.        .            .                 .           .      .        .        .
672   PEDCAA32    02MAY98:06:01:25        1          0.20    02       Sat      06
```

Appendix 3: Sample Output CPU Table

```
OBS    DSK       DATE       TIME     FREE   FREE_PC  DSKT_PC    SERVER          TS
1      0/C:   05/01/1998   6:09:17     .     46.90     0.00     PEDCAA31  01MAY98:06:09:17
97     0/E:   05/01/1998   6:09:17     .     24.68     0.00     PEDCAA31  01MAY98:06:09:17
.       .         .           .        .       .        .           .           .
193    0/C:   05/01/1998   6:16:25     .     54.74     0.00     PEDCAA32  01MAY98:06:16:25
```

Appendix 4: Sample Output Disk Table

```
OBS    SERVER          TS                 NETSEG           NSU     NSUDA    NSUDW    NSUHR
1     PEDCAA31    01MAY98:06:09:17      bh_CpqNF31         0.00     01       Fri      06
2     PEDCAA31    01MAY98:06:24:17      bh_CpqNF31         0.00     01       Fri      06
.         .            .                  .                .       .         .       .
97    PEDCAA32    01MAY98:06:16:25      bh_CpqNF31         0.00     01       Fri      06
```

Appendix 5: Sample Output Network Utilization Table

```
OBS          PAGFIL              PFU      PFUPK     SERVER          TS
1      \??\C:\pagefile.sys       2.06     90.75    PEDCAA31     01MAY98:06:09:17
97     \??\E:\pagefile.sys       0.63     26.46    PEDCAA31     01MAY98:06:09:17
.            .                    .         .          .            .
193    \??\C:\pagefile.sys       1.73     91.97    PEDCAA32     01MAY98:06:16:25
```

Appendix 6: Sample Output Paging File Table