ABSTRACT
Whether you are attempting to figure out what you have when preparing for a migration or you just want to find out which files or directories are taking up all of your space, SAS is a great tool to inventory and report on the files on your desktop or server. This paper intends to present SAS code to inventory and report on the location you want to inventory.

INTRODUCTION
Trying to get a handle on the number of different types of files on a server or network drives can be daunting. Every year a server is in operation, there are opportunities for users to drop programs, data and other types of files across many directories. After a few years there can be thousands of files and many gigabytes of data. At some point the time will come for cleaning up or migrating to a new server. When this time comes, SAS can be a handy tool to inventory your files, summarize files by age, size, directory, etc. SAS can also be used to search through SAS programs for references to servers or statements that interact with Microsoft Excel that need to be changed. Depending on the details of the migration, there may be different results needed from the inventory.

The examples in this paper will mostly be using Microsoft Windows in the examples but the concepts will certainly extend to UNIX and LINUX.

CONCEPTS

Building a Database of Files
Often, the first step in getting a handle on what files are on your server is to load the files and attributes into a SAS data set. Using pipes and operating system commands, SAS can interrogate a server or network drive and load the files and their attributes into a SAS data set. The following code is an example of building a SAS data set containing all of the files on the c:\myfiles\ directory of a Microsoft Windows desktop.

```sas
filename pipedir pipe ' dir "c:\myfiles" /S' lrecl=5000;
data indata;	infile pipedir truncover;	input line $char1000.;	length directory $1000;	retain directory;	if line = "" or 		index(upcase(line),'<DIR>') or 		left(upcase(line)) =: 'VOLUME' then 	del;	if left(upcase(line)) =: 'DIRECTORY OF' then 
directory=left(substr(line,index(upcase(line),'DIRECTORY OF') +12));	if left(upcase(line)) =: 'DIRECTORY OF' then 
del;	if input(substr(line,1,10),?? mmddyy10.) = . then 		substr(line,1,10) ='12/31/2999';

date=input(substr(line,1,10),?? mmddyy10.);
format date mmddyy10.;
run;

proc sort data=indata;	by directory descending date;
run;
```
data Directory_Summary(drop=i line);
set indata;
by directory;
length filename $75;
retain number_of_files_in_directory directory_size;
if first.directory then
   do;
      number_of_files_in_directory=input(scan(line,2,' '),32.);
      directory_size=input(scan(line,4,' '),comma32.);
   end;
   file_size=input(scan(line,4,' '),comma32.);
filename='';
do i=5 to 100;
   filename=trim(left(filename))||' '|scan(line,i,' ');
   if scan(line,i,' ')='' then leave;
end;
if index(upcase(line),'FILE(S)') then delete;
if date ge '30DEC2999'd then delete;
run;

Figure 1: Excerpt from directory_summary data set

<table>
<thead>
<tr>
<th>Directory</th>
<th>date</th>
<th>filename</th>
<th>number_of_files_in_directory</th>
<th>directory_size</th>
<th>file_size</th>
</tr>
</thead>
<tbody>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>01/21/2011</td>
<td>archive2010.pst</td>
<td>9</td>
<td>5058871296</td>
<td>1165149184</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>01/21/2011</td>
<td>archive2009.pst</td>
<td>9</td>
<td>5058871296</td>
<td>878691328</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>01/21/2011</td>
<td>archive2008.pst</td>
<td>9</td>
<td>5058871296</td>
<td>754254848</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>01/21/2011</td>
<td>archive2007.pst</td>
<td>9</td>
<td>5058871296</td>
<td>660800512</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>01/21/2011</td>
<td>archive2006.pst</td>
<td>9</td>
<td>5058871296</td>
<td>573694976</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>02/24/2009</td>
<td>archive2005_2.pst</td>
<td>9</td>
<td>5058871296</td>
<td>511984640</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>08/30/2007</td>
<td>archive2004.pst</td>
<td>9</td>
<td>5058871296</td>
<td>507282432</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>03/21/2005</td>
<td>archive (bvarney v1).pst</td>
<td>9</td>
<td>5058871296</td>
<td>4964352</td>
</tr>
<tr>
<td>c:_mystuff\VTPUdrive\e-mail archive</td>
<td>12/20/2007</td>
<td>archive2005.pst</td>
<td>9</td>
<td>5058871296</td>
<td>2049024</td>
</tr>
</tbody>
</table>

Loading Programs into a SAS Data Set
Another useful thing to do is to load all of the SAS programs into a data set. This will allow us to search for strings and if brave enough, alter the programs programmatically. The code below will loop over each distinct directory from the directory summary data set created above (see Figure 1) and load the programs into one data set. Each line of the program will be a record in the data set.

proc sql noprint;
   select distinct directory into :dir1 - :dir9999
   from directory_summary
   where scan(lowcase(filename),-1,'.')='sas';
quit;
%let numdirs=&sqlobs.;
%put &numdirs.;
Paper BB-13 (continued)

```sas
proc datasets nolist lib=work;
delete allprgs;
quit;

%macro getsasprgs;
%do i=1 %to &numdirs.;
data prgs;
  length line sasprogram f $300;
  infile "&&dir&i.\.sas" FILENAME=f;
  input;
  sasprogram =f 
  linenum+1;
  if lag(sasprogram) ne sasprogram then linenum=1;
  line=_infile_
run;
proc append base=allprgs data=prgs;
run;
%end;
%mend getsasprgs;
%getsasprgs;
```

Figure 2: Excerpt from allprgs data set

<table>
<thead>
<tr>
<th>Linenum</th>
<th>SASprogram</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>c:\junk.sas</td>
<td>libname mydat &quot;\servername\shareddrive&quot;;</td>
</tr>
<tr>
<td>2</td>
<td>c:\junk.sas</td>
<td>proc options long.run;</td>
</tr>
<tr>
<td>3</td>
<td>c:\junk.sas</td>
<td>proc print data=sashelp.class;</td>
</tr>
<tr>
<td>1</td>
<td>c:\junkcsv.sas</td>
<td>data a;</td>
</tr>
<tr>
<td>2</td>
<td>c:\junkcsv.sas</td>
<td>infile &quot;c:\junk.csv&quot; dsd ;</td>
</tr>
<tr>
<td>3</td>
<td>c:\junkcsv.sas</td>
<td>input a b c;</td>
</tr>
<tr>
<td>4</td>
<td>c:\junkcsv.sas</td>
<td>run;</td>
</tr>
</tbody>
</table>

Application #1: Creating a list of unneeded files
After running the code for creating the database of files on the location that you passed into the program, you will have a data set with the fields listed below:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Filename</th>
<th>Date</th>
<th>Number_of_Files_in_Directory</th>
<th>Directory_Size</th>
<th>File_Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1000</td>
<td>$75</td>
<td>numeric w/ mmddyy10.</td>
<td>numeric (in bytes)</td>
<td>numeric (in bytes)</td>
<td></td>
</tr>
</tbody>
</table>

A simple SQL query can be specified to find the old (older than January 1st, 2009) and large (approximately larger than 10GB) files:
Paper BB-13 (continued)

```
proc sql;
    select *
    from directory_summary
    where file_size ge 10000000000 and
    date le "01JAN2009"d;
quit;
```

**Application #2: Searching for references to a macro call.**
After running the code for creating the database of SAS programs, you will have a data set with the fields listed below:

- SASProgram $300
- LineNum 8
- Line $300

A simple SQL query can be specified to find any references to a macro called experis:

```
proc sql;
    select *
    from allprgs
    where lowcase(line) ? '%experis' or
    lowcase(line) ? '%macro experis';
quit;
```

**Application #3: Find all of the SAS Enterprise Guide Projects**
After running the code for creating the database of files on the location that you passed into the program, you will have a data set with the fields listed below:

- Directory $1000
- Filename $75
- Date numeric w/ mmddyy10.
- Number_of_Files_in_Directory numeric
- Directory_Size numeric (in bytes)
- File_Size numeric (in bytes)

A simple SQL query can be specified to find the SAS Enterprise Guide projects:

```
proc sql;
    select *
    from directory_summary
    where scan(lowcase(filename),-1,'.') = 'egp';
quit;
```

**Application #4: Searching for programs related to protocol x/0001/0001.**
After running the code for creating the database of SAS programs, you will have a data set with the fields listed below:

- SASProgram $300
- LineNum 8
- Line $300
A simple SQL query can be specified to find any references to protocol x/0001/0001:

```sql
proc sql;
    select *
    from allprgs
    where lowcase(line) ? 'x/0001/0001' or
       (lowcase(sasprogram) ? 'x' and
        lowcase(sasprogram) ? '0001' and
        lowcase(sasprogram) ? '0001');
quit;
```

CONCLUSION
SAS is a great tool to interrogate and report on data retrieved from your operating system regarding to the files on your disk drives. Whether you are preparing for a migration, want to find older large files not needed any more or need to search through your SAS programs for specific strings, SAS is a powerful tool at your disposal.

REFERENCES
SAS Technical Support Website www.support.sas.com

CONTACT INFORMATION
Your comments and questions are valued and encouraged. The author may be contacted at:
Brian Varney
Experis Analytics Practice
a Manpower Company
Business Analytics Practice
5220 Lovers Lane Suite 200
Portage, Michigan  49002
Work Phone:  (269) 553-5185
Fax:   (269) 553-5101
Email: Brian.Varney@Experis.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.