Paper 1296-2014

What's on my Mainframe?
A macro that gives you a solid overview of your data on the mainframe.

Jesper Michelsen, Group Risk Management, Nykredit A/S

ABSTRACT

In connection with the consolidation of our SAS® platform at Nykredit the data stored on the Nykredit z/OS SAS® installation had to be migrated (copied) to the new x64 Windows SAS platform storage.

However, getting an overview of these data on the z/OS mainframe can be difficult, and a series of questions arise during the process. For example: Who is responsible? How many bytes? How many rows and columns? When were the data created? And so on.

With extensive use of filename FTP and looping, and extracting metadata, it is possible to get an overview of the data on the host presented in an Excel sheet.

INTRODUCTION

During 2013 we at Nykredit has consolidated our SAS hardware platform from 250+ local installations and a z/OS SAS and data server to an almost solely Windows Server x64 based platform.

This consolidation to Microsoft Windows and the migration of data from z/OS to a new data storage has reduced the 'analytic' workload on our z/OS with more than 80%. The reduction in the consuming of MIPS paid the project. As a side effect many of the jobs runs 50-70% faster on the new platform and it has become easier to automate and schedule jobs.

About 1.700 data catalogues at the z/OS data server had to be investigated and migrated to the new data storage.

We discovered that most users lacked the necessary skills to work in the 3270-enviroment, and it was difficult for them to get at total overview of all of their data catalogues in this environment.

The users could allocate the catalogues as a libref one by one in SAS® and investigate the content. But that is time consuming – and boring to do by hand.

A little creative thinking and the SAS® FILENAME statement, FTP Access method, and the included FTP Options provided us with the solution to this task.

With the SAS® FILENAME statement, using the FTP Access Method you can 'remote' control FTP commands sent to your host – in our case the z/OS mainframe – through SAS®.

You don't have to provide the FTP commands with a fully qualified "catalogue -path" – it is possible to construct a "search-string" including an asterisk '*'

And you can even read the information from the "FTP filename" into a dataset.

But - if your host or a firewall isn't opened for FTP you have to speak to your systems manager

STEP-BY-STEP

The first step is to construct the right FILENAME statement

- it is possible to construct a search string for the LSFILE= including an asterisk '*'

```
filename ftplist ftp '' ls cd='..' lsfile="&_ds..*" user="&_usr"
   host="&_host" pass=&_password;
```

The LS command is issued and that the LSFILE command controls for which part of the data catalogues the list should be made. This gives you the desired list of data catalogues on the z/OS mainframe.

The _ds macro variable contains either the fully qualified "DSN path" or a part of it.

The second step is to read the list into a SAS® dataset

- you probably have to know a little about how the list will look like

```
data DSNdata01;
       length DSNx $ 128;
       infile ftplist missover pad;
       input DSNx $128.;
Run;
```

Special characters like 'ÆØÅæøå' are important as they have a special meaning on a z/OS – you may need to translate in some way to get a reasonable result.

The third step is to loop through the dataset allocating the z/OS cataloges one by one

- we used some of the functions available to %sysfunc – open, fetchobs, getvarc, close
- in each loop we allocated a new libname and retrieved the desired information through the dictionary tables
- how many loops is decided by the number of records in the above dataset

We chose to extract the following information for each member in the catalogue

- name, memtype, crdate, modate, nobs, nvar, obslen, filesize and encoding
- for administrative use we also added some 'empty columns' e.g. (new)destination, status, who is responsible

The information is then appended to a master dataset which eventually is exported to an Excel workbook using a SAS® LIBNAME statement for Excel.

The whole sequence was built in to a macro to make it easier for users to retrieve the informations they needed

- actually we made two macros
- one which just gives a list of the catalogues on the z/OS from a certain search-point
- one which extracts the above mentioned information from a certain search-point

THE MACRO

The macro needs four parameters

- _host; which is the name of the mainframe server
- _ds; which is the "seach-string"; e.g. KAF.SAS.SASDATA.IAS*
- _complete; indicates whether the search-string is fully qualified or not
- _excelpath; the path to where the resulting excel workboos should be located. The macro is naming the workbook file by itself.

```
%macro listDSNdata2Excel(_host, _ds, _complete , _excelpath);
options msglevel=N;
%local _i _dsid _disp _excelpath _rc _rclib _dsn _host _complete _dsx _file;
%global _usr;

%readpwkrypt(&_host);

%if %upcase(&_complete) = C %then
   %do;
    filename ftplist ftp '' ls cd='..' lsfile="&_ds" user="&_usr" host="&_host"
        pass=&_password;
   %end;
%else
%if %upcase(&_complete) = U %then
   %do;
```

```
      filename ftplist ftp '' ls cd='..' lsfile="&_ds..*" user="&_usr"
             host="&_host" pass=&_password;
    %end;
 %if %eval(&sysfilrc=0) %then
    %do;
      proc datasets lib=work kill;
      run;quit;

      data DSNdata01;
      length DSNx $ 128;
      infile ftplist missover pad;
      input DSNx $128.;
      run;

      data DSNdata01(drop=DSNx);
      set DSNdata01;
      DSN=translate(DSNx,'Ø','9D'x);
      run;

      proc sort data=dsndata01;
      by dsn;
      run;
    %end;
  %else
  %do;
      %PUT WARNING: Cannot find any DSN with these qualifiers: %upcase(&_ds.);
      %put WARNING: Correct and try again;
  %end;
      filename ftplist clear;

/* Step 2: Allocate DSN and examine content */
  %if %sysfunc(exist(work.DSNdata01))=1 %then
     %do;
       %son(&_host);
       %nobs(work.DSNdata01);
       %let _dsid=%sysfunc(open(DSNdata01));
       %put _dsid: &_dsid;
         %do _i=1 %to &_nobs;
            %let _rc=%sysfunc(fetchobs(&_dsid,&_i));
            %let _DSN=%sysfunc(getvarc(&_dsid,1));
            libname HOST "&_DSN" server=SASNK02 access=readonly;
            %let _rclib=%sysfunc(libref(HOST));

            %if &_rclib=0 %then
               %do;
                  proc sql;
                  create table _dsnmembers&_i. as
                  select resolve("&_DSN") as DSNHOST length=128 format=$128.
                        ,memname as name
                        ,memtype as type
                        ,crdate
                        ,modate
                        ,nobs
                        ,nvar
                        ,obslen
                        ,filesize
                        ,encoding
                  from dictionary.tables
                  where libname='HOST';
                  quit;

            %if &sqlobs=0 %then
               %do;
```

```
              data _dsnmembers&_i.;
              length DSNHOST $ 128
                     navn $ 32
                     type $ 8
                     crdate
                     modate
                     nobs
                     nvar
                     obslen
                     filesize 8
                     encoding $ 256;
                     DSNHOST=resolve("&_DSN");
                     navn='DSN is empty';
                     type='N/A';
                     crdate=0;
                     modate=0;
                     nobs=0;
                     nvar=0;
                     obslen=0;
                     filesize=0;
                     encoding='N/A';
               run;
          %end;
      %end;
 %else
   %do;
       data _dsnmembers&_i.;
       length DSNHOST $ 128
               navn $ 32
               type $ 8
               crdate
               modate
               nobs
               nvar
               obslen
               filesize 8
               encoding $ 256;

       DSNHOST=resolve("&_DSN");
       navn='N/A';
       type='N/A';
       crdate=0;
       modate=0;
       nobs=0;
       nvar=0;
       obslen=0;
       filesize=0;
       encoding='DSN in use;
       run;
 %end;
       proc append base=dsnmembers01
                    data=_dsnmembers&_i force;
       run;
       libname HOST clear;
%end;
%let _rc=%sysfunc(close(&_dsid));
 %put RC fra 'CLOSE': &_rc;

 proc sql;
 create table dsn as
 select '' as plan label='Planned? Y/N'
       ,DSN label='DSN at Host'
       ,'' as Migrate label='Migrate Y/N?'
```

```
        ,'' as Method label='Method: Manuel/Batch'
        ,'' as Destination label='Destination'
        ,'' as Info_date label'Info on migration to (date)'
        ,'' as Info_init label='Info om migration mail to'
        ,'' as Info_OK label='OK (date)'
        ,'' as DSN_Read_Only_date label='DSN R/O (date)'
        ,'' as Control_date 'Migration controlled (date)'
        ,'' as NO_Read_date 'DSN no-read/no-write (date)'
 from dsndata01;

 create table datasets as
 select '' as plan label='Planned? Y/N'
        ,'' as ansv label='Responsebility for migration of DSN'
        ,DSNHOST label='DSN at Host'
        ,navn label='Member Name in DSN'
        ,type label='Member-type'
        ,crdate label='Create date'
        ,modate label='Modified date'
        ,nobs label='No of rows’
        ,nvar label='No of columns'
        ,obslen label='No of bytes per row'
        ,(filesize/1024) as filemb label='Filsize in Kb'
        ,encoding label='Encoding’
        ,'' as Migrate label='Migrate Y/N?'
        ,'' as Method label='Method: Manuel/Batch'
        ,'' as Destination label='Destination'
        ,'' as Info_date label'Info on migration mailed (date)'
        ,'' as Info_init label='Info on migration mailed to’
        ,'' as Info_OK label='OK (date)'
        ,'' as DSN_Read_Only_date label='DSN R/O (date)'
        ,'' as Control_date 'Migration controlled (date)'
        ,'' as NO_Read_date 'DSN no-read/no-write (date)'
 from dsnmembers01;
 quit;

 %let dato=%sysfunc(today(),date9.);
 %let time=%sysfunc(time(),hhmm8.);
 %let _file=&_excelpath.\DSNdata %upcase(&_ds.) pr. &dato..xlsx;
 %put NOTE: The EXCEL workbook is filed - &_file.;

 libname myxlbook "&_file.";

 %if %sysfunc(fileexist(&_file)) = 1 %then
   %do;
        proc datasets lib=myxlbook nolist;
        delete dsnmembers02 dsndata02;
        %_run(wq);
   %end;

 data myxlbook.DSN(dblabel=yes);
 set dsn;
 %_run(w);

 data myxlbook.Datasets(dblabel=yes);
 set datasets;
 %_run(w);

 libname myxlbook clear;

%end;
%soff(&_host);
%mend listDSNdata2Excel;
```

The readPWkrypt in-house Nykredit macro in line 5 reads a encrypted password stored in SASUSER. Initially the password was encrypted by the in-house Nykredit macro PWkrypt.

Line 6-16 is the FILENAME statements. Only one of them is executed, which depends on the value of the macro variable _complete. If _complete has the value U (for uncompleted)the macro variable _ds is extended with ".*" to indicate that the "search-string" is not fully qualified.

If the filename is a success (line 17) the sequence in line 21-32 is executed. If not a WARNING appears in the log. The rest of the macro fails to execute.

If the sequence in line 21-32 creates the dataset 'DSNdata01' then a signon to the host (the mainframe server) is executed. This is done through aa in-house Nykredit macro called 'son'. After this the in-house Nykredit macro nobs retrieves the number of records from the dataset DSNdata01 and put the value into a macrovariable called _nobs.

The looping through the dataset is then initialized and for each loop the name of the mainframe DSN is retrieved and used for allocation of a libname called HOST.

The information needed is then selected from dictionary.tables where the libame is equal to 'HOST'. If nothing ca be selected, a NULL record is constructed instead. If the DSN is in use or for some other reason not available a 'DSN in use'-record is constructed.

The records are appended to a master dataset in line 120-122 and the dataset 'DSNdata01' is closed. After this the extra administrative empty columns is created (line 127-164).

Finally the datasets are exported to Excel (line 169-182) and the connection to the host is disconnected using an in-house Nykredit macro called 'soff '.

CONCLUSION

SAS® FILENAME Statement with the FTP ACCESS Method made it easy for us to retrieve the needed information from the mainframe data server necessary for the migration work.

The resulting Excel workbooks proved to be a good tool for the users managing the migration work.

REFERENCES – AND RECOMMENDED READING

SAS Institute Inc., SAS® 9.3 Statements: Reference - FILENAME Statement, FTP Access Method
http://support.sas.com/documentation/cdl/en/lestmtsref/63323/HTML/default/viewer.htm#p0v0ijxl1k6d4bn16cshtic7u4i3.htm

SAS/ACCESS® 9.3 Interface to PC Files: Reference (SAS Institute Inc.), chapter on Libname Access and Excel Engines on Microsoft Windows.
http://support.sas.com/documentation/cdl/en/acpcref/66980/HTML/default/viewer.htm#titlepage.htm

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Senior Specialist
Jesper Michelsen
Nykredit A/S
Kalvebid Brygge 47
1560 Copenhagen V
Work Phone: +45 4455 1362
Email: jmic@nykredit.dk