Macros to get to “Know Thy Data”

Diane Foose, GlaxoSmithKline, Collegeville, Pa
Mike Gunshenanan, GlaxoSmithKline, Collegeville, Pa
Rick Morales, GlaxoSmithKline, Collegeville, Pa

ABSTRACT:
Every programmer at one time or another has heard the motto ‘Know Thy Data’. There have been many papers on this subject over the years at all the SUGs. Well, here is a practical paper to help you to get to know your data. In this paper, we will give you some macros to use and before you know it, you will Know Your Data. Starting with printing of the data through getting metadata about your data, hopefully, you will find something that will help you get your work finished faster. These macros are a stepping stone to use while you sift through the various layers of your data. As you use them you can enhance as you see fit. The key to these macros is that they are simplistic and general in scope to apply to many different areas.

INTRODUCTION:
Getting to know your data is an arduous task. Before you can begin manipulating, summarizing and reporting the contents of the collected data, you have to have a deeper understanding of what is contained in the data sets. SAS® software provides numerous tools in the form of data steps, data views and procedures to delve into the dissection of your data. We have chosen several of the macros we use to open the window into understanding what is contained in our data. These tools will allow the user to call a macro to produce a print out of the data, to view the contents of the data, to compare data sets to each other to determine differences in variable attributes, to produce a proc freq or proc means on the variables of interest in a data set as well as produce a combination of contents, freqs and means and to make comparisons on various dates to determine if a date is valid.

We are always debugging our code, testing our programs or just trying to look at our data. SAS is a very easy programming language to use for this purpose, but it can always be made simpler with the use of macros. The following is a description of each of these macros. Each macro can be found at the end of the paper.

%TESTPRINT:
The first of the several macros presented in this paper deals with displaying data and the metadata. %testprint macro will print the data using a PROC PRINT statement with options to print a designated number of observations, a subset of the data using a WHERE statement, break the data out using a BY statement, list only the variables of interest in a VAR statement and format those variables with a FORMAT statement, and print a meaningful TITLE. All, some or no options can be used. Finally, if you want to see the contents of the data set along with the PROC PRINT, there is an option to do a PROC CONTENTS.

If you need to look at one patient’s data from a data set, you can type in the PROC PRINT command yourself or use this macro. Both of these pieces of code will produce the same output::

```sas
   title 'lib1.demog dataset';
   proc print data=lib1.demog (where=(subjid=222));
   or
   %testprint(ds=lib1.demog,where=subjid=222);
```

Another example would be if you wanted to subset the data, only print certain variables and format the data to be readable. The following two versions of code would produce the same output.

```sas
   title 'Lib1.demog dataset';
   proc print data=lib1.demog;
   where sex = 'F'
   var subjid sex age race;
   format sex $sex. race $race.;
   or
```
%testprnt(ds=lib1.demog,where=sex='F',var=subjid sex age race,format=sex $sex. race $race.);

Let's say you needed to sort your data before listing the output, only want to show the first 100 observations and want a proc contents to view what variables are present on the data set. The following two pieces of code will produce the same output:

title 'ae dataset';
proc sort data=ae;
by subjid aept aestdt aesttm;
proc print data=ae (obs=50);
proc contents data=ae;
OR
%testprnt(ds=ae,sort=subjid aept aestdt aesttm, obs=100, contents=Y);

As you can see this macro is quick, easy and intuitive. It will not replace standard code, but is very useful when needing a quick printout of your data and can save time by allowing you to type less code.

%COMPARE

The second macro, %comparec, is used to compare two data sets and display the differences between the variables created in each of the data sets. If your task is to QC a data set by recreating all the variables, this is a simple check to compare the two data sets to affirm that both data sets have the same variables. A PROC COMPARE will tell you how many variables are in one data set but not the other, but does not tell you what the variables are. This macro performs a PROC CONTENTS on two data sets, creates an attributes data set of each and compares the differences between the two using PROC SQL. With this information you can make sure the data sets are properly validated.

%comparec(dsl=lib1.demog,ds2=work.qcdemog);

Output:
"Variables in lib1.demog and not in work.qcdemog";
Variable Name
CENTREID
INVID
"Variables in work.qcdemog and not in lib1.demog";
Variable Name
AGE
SEX

The next three macros all do similar tasks, but through a different process. You may find that you can use pieces of the code to perform other tasks of your choosing.

%CHECKTAB

The third macro, %checktab, will perform a PROC FREQ or PROC MEANS on a data set using the SASHELP.VCOLUMN in a DATA step. The macro creates a table of variable names and determines whether a PROC FREQ or a PROC MEANS is the applicable procedure to use based on the type of data (character or numeric). All variables (as a default) can be displayed or you can pick and choose which variables you want to investigate. Below is an example of using this macro to review the statistics of two variables (age & race) in the demog data set.

%checktab(library=lib1, table=demog, keepvar=%str('age','race'));

output:
Table LIB1.DEMOG
Obs name type format
 1 RACE char
 2 AGE num
Table LIB1.DEMOG, Variable Age
The MEANS Procedure
Analysis Variable : AGE Age in years
N Mean Std Dev Minimum Maximum
-----------------------------------------------
The FREQ Procedure

Race

<table>
<thead>
<tr>
<th>Race</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>African American/African Heritage</td>
<td>15</td>
<td>3.45</td>
<td>15</td>
<td>3.45</td>
</tr>
<tr>
<td>Asian - Central/South Asian Heritage</td>
<td>3</td>
<td>0.69</td>
<td>18</td>
<td>4.14</td>
</tr>
<tr>
<td>Asian - East Asian Heritage</td>
<td>121</td>
<td>27.82</td>
<td>139</td>
<td>31.95</td>
</tr>
<tr>
<td>Asian - South East Asian Heritage</td>
<td>122</td>
<td>28.05</td>
<td>261</td>
<td>60.00</td>
</tr>
<tr>
<td>White - White/Caucasian/European Heritage</td>
<td>174</td>
<td>40.00</td>
<td>435</td>
<td>100.00</td>
</tr>
</tbody>
</table>

The MEANS Procedure

<table>
<thead>
<tr>
<th>Variable</th>
<th>Label</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEIGHT</td>
<td>Height (cm)</td>
<td>669</td>
<td>163.2660688</td>
<td>7.4905769</td>
<td>140.0000000</td>
<td>196.0000000</td>
</tr>
<tr>
<td>WEIGHT</td>
<td>Weight (kg)</td>
<td>670</td>
<td>74.5983582</td>
<td>18.0798448</td>
<td>43.0000000</td>
<td>173.1000000</td>
</tr>
</tbody>
</table>

PART B:

DATASET "demo" DOES NOT EXIST IN CURRENT LIBRARY: library

%CHKALL

The fifth macro %chkall looks to a library to examine all of the data sets therein. Like the previous macros it runs through the list of data sets in the provided directory and produces one (sometimes two) procedures on every dataset. While similar to %checktab and %getdatadesc, this procedures is intended for use on directories where you are reviewing everything coming in (versus looking for anything in particular). The procedures available are FREQ, MEANS, CONTENTS & PRINT. The three input macro variables are LIBNAME for the location of your data, LOOKAT for the choice of procedure(s) and OBS to control the number of observations printed when PRINT is entered in LOOKAT. LOOKAT can be populated with

1. STATS to run FREQ on character variables and MEANS on numeric variables producing N, Mean, STD, Median, Min & MAX; or
2. FREQ to run a FREQ on all variables (which may be more useful than MEANS for examining things like race codes or visit numbers) in all data sets or PROC FREQ on the variables.

3. PRINT to PRINT a sample of observations (default is 10 observations, but can be controlled by &OBS in macrocall.)

4. PRINTALL to PRINT all observations.

5. CONTENTS to run a CONTENTS on all data sets.

PROC CONTENTS DATA=_ALL_ is used to get your list of all data sets within the directory of interest. After that DATA _NULL_ and SYMPUT are used to create macro variables for all data sets and for PROC MEANS when &LOOKAT=STATS the numeric variables within a data set.

A sample macro call:
%CHKALL ( LIBNAME = /prod/AB123456/XXX10001/data_look/rawdata/ , lookat=StatS)

The first portion of output lists the Data Sets with their Labels:

<table>
<thead>
<tr>
<th>Obs</th>
<th>MEMNAME</th>
<th>MEMLABEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AE</td>
<td>Adverse Events</td>
</tr>
<tr>
<td>2</td>
<td>BLIND</td>
<td>Treatment Blind</td>
</tr>
<tr>
<td>3</td>
<td>CMANAL</td>
<td>Conneds</td>
</tr>
<tr>
<td>4</td>
<td>CYCLE</td>
<td>Cycle</td>
</tr>
<tr>
<td>5</td>
<td>DEATH</td>
<td>Death</td>
</tr>
<tr>
<td>6</td>
<td>DEMO</td>
<td>Demography</td>
</tr>
</tbody>
</table>

The next portion will be the output from the chosen procedures (in this case STATS calling FREQ and MEANS).

Dataset: AE (Adverse Events)
Frequency of Character Variables

The FREQ Procedure

<table>
<thead>
<tr>
<th>AEACTRCD</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>29</td>
<td>1.08</td>
<td>29</td>
<td>1.08</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0.04</td>
<td>30</td>
<td>1.12</td>
</tr>
<tr>
<td>4</td>
<td>1502</td>
<td>56.11</td>
<td>1532</td>
<td>57.23</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0.19</td>
<td>1537</td>
<td>57.42</td>
</tr>
<tr>
<td>X</td>
<td>1140</td>
<td>42.58</td>
<td>2677</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Frequency Missing = 7

And for the MEANS procedure on numerics:

Dataset: AE (Adverse Events)
Stats for AEDUR: Duration of event

<table>
<thead>
<tr>
<th>Obs</th>
<th>n</th>
<th>mean</th>
<th>std</th>
<th>median</th>
<th>min</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1837</td>
<td>8.64453</td>
<td>10.6244</td>
<td>6</td>
<td>-58</td>
<td>99</td>
</tr>
</tbody>
</table>

The macro Puts the data set name (and label when not missing) in the title and includes the variable label for the MEANS output in the title.

When a data set in the directory is empty, it will output a note stating the data set has no observations:

Dataset: BLIND

<table>
<thead>
<tr>
<th>Obs</th>
<th>NOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data Set BLIND has no observations</td>
</tr>
</tbody>
</table>
CONCLUSION:
These 5 macros are just a brief look into an endless array of tools using SAS to delve into the complexities of the various types of data we encounter on a daily basis. Each one has been developed to determine the validity and scope of the data being utilized in our working environment. Any or all of these macros can be adjusted to your own environment or modified to assist you in getting to know your data. With these tools, we can better understand the various differences in the data and adjust our tools to further assist us to “Know Thy Data”.

REFERENCES

SAS is a registered trademark or trademark of SAS Institute, Inc. in the USA and other countries. ® indicates USA registration.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the authors at:

Diane Foose, Mike Gunshenan, Richard Morales
GlaxoSmithKline
Mailstop UP4310
1250 S Collegeville Rd
Collegeville, Pa 19426
diane.e.foose@gsk.com, michael.p.gunshenan@gsk.com, richard.j.morales@gsk.com
%macro testprint(ds=_last_, obs=, where=, var=, format=, title=&ds dataset, sort=, contents=N, nobs=Y, by=);

   %if "%sort" ^= "" %then %do;
       proc sort data=&ds;
       by &sort;
   %end;

   proc print data=&ds %if %upcase(&nobs) = N %then noobs; %if "%obs" ^= "" %then (obs=&obs);
   %if "%where" ^= "" %then where &where;=
   %if "%by" ^= "" %then by &by;
   %if "%var" ^= "" %then var &var;
   %if "%format" ^= "" %then format &format;
   title ":title";
   run;

   %if %upcase(&contents) = Y %then %do;
       proc contents data=&ds;
   %end;

%mend testprint;
%macro comparec(ds1=dmdata,ds2=ardata);
proc contents data=&ds1 out=dslcontents noprint;
proc contents data=&ds2 out=ds2contents noprint;
proc sql;
title "Variables in &ds1 and not in &ds2";
  select distinct name from dslcontents where upcase(name) not in (select
    upcase(name) from ds2contents);
title "Variables in &ds2 and not in &ds1";
  select distinct name from ds2contents where upcase(name) not in (select
    upcase(name) from dslcontents);
quit;
%mend comparec;

%macro checktab;
%* Macro:           checktab.sas          *;
%* SAS Version:    Release 8.2           *;
%* Author:         Diane Foose           *;
%* Date:           20-June-2000           *;
%* Purpose:        Macro to check all desired variables in a given dataset *;
%* by either a proc freq or proc means *;
%* Invocation:     %checktab(library=,table=,dropvar=,check=,checkcnt=) *;
%* Assumptions:    Works on any system for any dataset as long as library is *;
%* already defined and table exists before the macro is called. *;
%* Parameters:     (Keyword)             *;
%*                 library = Library name (default is work) *;
%*                 table   = Dataset name *;
%*                 keepvar = List of variables to be analyzed (default is all) *;
%*                 dropvar = List of variables not to be analyzed *;
%*                 check = To check for how many distinct values of each *;
%*                 variable, if > &checkcnt then do a proc means *;
%*                 (default is Yes) *;
%*                 checkcnt = Number to compare against to determine if a proc *;
%*                 means or Proc freq should be used. *;
%* Input:          &library..&table   *;
%* Output:         Proc Freq & Proc Means output *;
%* Parameters:     (Keyword)             *;
%macro checktab(library=work,table=,keepvar=,dropvar=,check=Y,checkcnt=10);

%_findfil (filen= checktab) ;

%let table = %upcase(&table);
%let library = %upcase(&library);
%let keepvar = %upcase(&keepvar);
%let dropvar = %upcase(&dropvar);
%let check = %upcase(&check);

/*** Pull all variables desired from table ***/
/*** Create incremental macro variables for each dataset variable ***/
data checktab (keep = name type format);
  set sashelp.vcolumn end=last;
  where mename = "&table"
    and libname = "&library"
  %if "&keepvar" ^= "" %then
    and name in (&keepvar);
  %else %if "&dropvar" ^= "" %then
    and name ^in (&dropvar);
  ;
  retain n 0;
  n = n + 1;
  call symput(compress("name"||put(n,8.)),name);
  call symput(compress("type"||put(n,8.)),type);
  call symput(compress("format"||put(n,8.)),format);
  if last then
    call symput("num",compress(put(n,8.)));
run;

/*** Print out List of Variables in Table to be analyzed ***/
title "Table &library..&table";
tproc print;

/*** Loop to perform either proc means or proc freq for each variable ***/
%do i = 1 %to &num;
  %if &check = Y %then %do;
    /*** Check to see how many distinct values for each variable ***/
    proc sql noprint;
    select count(distinct &name&i) into: count
    from &library..&table;
    %if &count > &checkcnt and &type&i = num %then %do;
      proc means data=&library..&table;
      var &name&i;
    %end;
  %else %do;
    proc freq data=&library..&table;
    table &name&i / missing;
  %end;
%end;
%else %do;
  proc freq data=&library..&table;
  table &name&i;
%end;
Program: getdatadesc.sas
Author: Rick Morales
Date: 06-MAR-05
Purpose: Macro to create a dataset of library dataset names to use in a call to produce a proc contents/freq/means of a dataset called:
Assumptions: library previously set-up
Invocation: %getdatadesc(lib=ardata, dtaset=demo, contents=N, freq=Y,
: frqvn=%str(agegrp), stat=Y, statvn=%str(height weight));
Parameters: lib = Name of library where datasets reside
: dtaset = Name of dataset used for output
: contents = Y/N to produce proc contents
: freq = Y/N to produce proc freq
: frqvn = names of variables for proc freq
: stat = Y/N to produce proc means
: statvn = names of variables for proc means
Input: All parameters must have a value
Output: Print out options of contents, freq and/or means

/* Get all dataset info from a library */

%macro getdatadesc(lib=, dtaset=,contents=,freq=, frqvn=, stat=, statvn=);
proc contents data=&lib.._all_ noprint out=all;
run;

/* Only keep name and one record per dataset */

proc sort data=all out=alll (keep=memname) nodupkey;
by memname;
run;

/* Transpose the dataset to create a variable name out of each dataset in the library */

proc transpose data=alll out=dset (drop=_name_);
   id memname;
run;

/* Count number of datasets in the library for a counter */

%let countn=0;

data _null_; 
   set alll end=eof;
   if memname ne '' then do;
      cnt+1;
   end;
end;
if eof then do;
call symput('countn',trim(left(put(cnt,8.))));
end;
run;

/* This macro will generate the dataset name for use as macro variable in the subsequent calls */
/* Run a proc contents/freq/means on the dataset you choose */

%if &countn ne 0 %then %do;
  %let dn=0;
  %do j = 1 %to &countn;
    %let ds=%sysfunc(open(dset));
    %let dsn=%lowercase(%sysfunc(varname(&ds, &j)));
    %let dc=%sysfunc(close(&ds));
    %if &dsn=&dataset %then %do;
      %let dn=1;
      %if &contents=Y %then %do;
        proc contents data=&lib..&dataset;
        run;
      %end;
      %if &freq=Y %then %do;
        proc freq data=&lib..&dsn;
        tables &frqvnn;
        run;
      %end;
      %if &stat=Y %then %do;
        proc means data=&lib..&dsn;
        var &statvn;
        run;
      %end;
      %end;
  %end;
%end;
%if &dn=0 %then %do;
  %put DATASET DOES NOT EXIST IN CURRENT LIBRARY: &lib;
%end;
%end;
%mend getdatadesc;
* STATS - provides a frequency on character*;
* variables and a means on numeric *;
* variables *;
* FREQ - provides a frequency on all *;
* variables (both character & numeric)*;
* PRINT - prints the number of observations *;
* specified by OBS (default is 10) *;
* PRINTALL - prints all observations *
* CONTENTS - provides a proc contents *
* (NOTE: LOOKAT can only contain one value per *
* submission of the macro. If more than *
* one of the previous is desired, the *
* macro must be called multiple times *)
* OBS = When PRINT is chosen, this is the number of *
* observations to be printed (default is 10) *
* FORMATS: None *
* INPUT: Datasets from the &LIBNAME directory *
* OUTPUT: CHKALL.LST containing the output of the procedure(s) *
* called *
* MODIFICATIONS:
* DATE PROG MOD # REASON *
* ------- ---- ---- ------------------------------- *
* options ps=50 ls=130 nofmterr mlogic mprint symbolgen msglevel=i;

* READ THE DATALIST FILE AND CREATE MACRO VARIABLES TO REPRESENT ALL *
* OF THE DATASETS IN THE DIRECTORY. [D1 THROUGH Dn] *
* ***************************************************************;
%MACRO CHKALL( LIBNAME = , LOOKAT=, OBS=10);
  libname dir ";&libname";
  proc contents data=dir._all_(DROP=FORMAT) out=datalist(keep=memname mem1:) NOPRINT;
  run;
  proc sort data=datalist NODUPKEY ;
    by memname ;
  run;
  proc print data=datalist ;
  run;
  run cancel ;
  %GLOBAL DSETCNTR ;
  %LOCAL I J K;
  DATA _NULL_;
    SET DATALIST END = FINAL;
    BY MEMNAME NOTSORTED;
    IF (LAST.MEMNAME);
    N+1;
IF (FINAL) THEN CALL SYMPUT( 'DSETCNTR', PUT( N, 8. ) );

RUN;

%DO J = 1 %TO &DSETCNTR;
   %global d&j label&j;
   %let dlabel&j=;
%END;

DATA _NULL_;  
SET DATALIST END = FINAL;  
BY MEMNAME NOTSORTED;

IF (LAST.MEMNAME);
N+1;

CALL SYMPUT( 'D'||LEFT( PUT( N,8. ) ), TRIM( MEMNAME ) );
IF MEMLABEL NE '' THEN CALL SYMPUT( 'DLABEL'||LEFT( PUT( N,8. ) ), TRIM( MEMLABEL ) );

RUN;

******************************************************************************
** MACRO CHECK RUNS THE ASSIGNED MACRO FOR ALL DATASETS. **
******************************************************************************

******************************************************************************
** CREATE MACRO TO OUTPUT PRINT/CONTENTS SEPARATELY BY DATASETS **
******************************************************************************

%MACRO CONT(DSIN = , LAB=, LOOKAT=);
title1 "Dataset: &dsin";
%if &d&label&j ne %str() %then
   %do;
      title1 "dataset: &dsin (&d&label&j)";
   %end;
%global dset anyobs;
%let ds= dir.&dsin;
%let dsi=%sysfunc(open(&ds));
%let anyobs= %sysfunc(attn(&dsi,nobs));
%let cr=%sysfunc(close(&dsi));
%if &anyobs = 0 %then
   %do;
      data empty;
      NOTE="Data Set &dsin has no observations";
      run;
      proc print data=empty;
      run;
      %end;

%if %upcase("&lookat")="FREQ" %then
%do;
   title2 'Frequency of All Variables';
   proc freq data = dir.&dsin ;
      tables _all_;
   run;;
%end;
%if %upcase("&lookat")="STATS" %then
  %do;
    title2 'Frequency of Character Variables';
    proc freq data = dir.&dsin ;
    tables _character_ ;
    run ;
  proc contents data = dir.&dsin out = contents (keep = memname label name type) NOPRINT;
    run ;
  proc sort data = contents ;
    by name ;
    where type = 1 ;
    run ;
  %global dset nobs ;
  %let dset = contents;
  %let dsid = %sysfunc(open(&dset)) ;
  %let nobs = %sysfunc(attrn(&dsid,nobs)) ;
  %let rc = %sysfunc(close(&dsid)) ;
  %if &nobs > 0 %then
    %do ;
    proc print data = contents ;
    run cancel ;
    run ;
  %end ;

  %LOCAL I J VARNTR ;

  DATA _NULL_ ;
    SET contents END = FINAL ;
    BY NAME NOTSORTED ;
    IF (LAST.NAME) ;
    N + 1 ;
    IF (FINAL) THEN CALL SYMPUT ('VARNTR' , PUT ( N , 8 . ) ) ;
  RUN ;

  %DO J = 1 %TO &VARNTR ;
    %local v&j label&j ;
    %END ;

  DATA _NULL_ ;
    SET contents END = FINAL ;
    BY NAME NOTSORTED ;
    IF (LAST.NAME) ;
    N + 1 ;
    CALL SYMPUT ( 'V' || LEFT( PUT ( N , 8 . ) ) , TRIM ( NAME ) ) ;
  

CALL SYMPUT('LABEL'|LEFT( PUT(N,8.) ) , TRIM(LABEL ) );

RUN;
%do k=1 %to &varcntr;
title2 'Means of Numeric Variables';
title3 "Stats for &&v&k: &&label&k";
proc means data = dir.&dsin noprint ;
  var &&v&k ;
  output out=chk n=n mean=mean std=std median=median min=min max=max ;
run;
proc print data=chk(drop=_:);
run;
%end;
%let varcntr=;
%end;
%if %upcase("&lookat")="CONTENTS" %then
  %do;
    proc contents data = dir.&dsin ;
    run;
  %end;
%end;
%if %upcase("&lookat")="PRINT" %then
  %do;
    proc print data = dir.&dsin(obs=&obs); run;
  %end;
%end;
%if %upcase("&lookat")="PRINTALL" %then
  %do;
    proc print data = dir.&dsin ;
    run;
  %end;
%end;
%MEND CONT ;
%MACRO CHECK(macname=, lookat=);
  %DO J = 1 %TO &DSETCNTR ;
    &&macname(dsins= &&d&j, lab=&&dlabel&j, lookat=&lookat )
  %END;
%MEND CHECK;
%CHECK(macname=CONT, lookat=&lookat)
%MEND CHKALL;
%CHKALL( LIBNAME = /arenv/arprod/gw679769/nkv101983/data_look/ardata/ , lookat=StatS)