

PP07

Compare program header with program syntax – A check program to create and compare the program header- and section comments

Adrian Czaban¹; Charlotte Bergknut²

¹Novo Nordisk A/S, Søborg, Denmark (adcz@novonordisk.com); ²Orphazyme A/Z, Copenhagen, Denmark, former employed at Novo Nordisk A/S, Søborg, Denmark;

Goal

- Electronic check to ensure header of the program corresponds to body text (syntax) in the program
- Generate the header by information that already exist, that is from global macro variables and program body text

Motivation

- Efficiency:
 - Saves time
 - Reusing already existing information by captured by a program
- Quality:
 - By using data from the programs to generate header, the header and body text in the program will always be aligned

Components

Header template:

```
*****
Description :
Input       :
Output      :
Usage notes :
-----
Programmer  :
*****
```

Example of a filled out header:

```
*****
Description : This program generates the table ADAM.ADSL.
               The program goes through the following steps:
               LIB: Setup libraries
               DEM: Extract demographics data
               DIS: Extract disposition data
               USE: Extract substance use data
               DER: Merge subject level data and do cross domain derivations
               FIN: Finalize dataset

Input       : SDTM.DM
               SDTM.SUPPDM
               SDTM.DS
               SDTM.SU
               SDTM.SUPPSU
               METADATA.MDCOL

Output      : ADAM.ADSL
Usage notes : This program should run before BDS and OCCDS programs
Programmer  : XXXX
*****
```

The prerequisite is that the comments in the program uses standard names for the sections.

Example of a section in the program:

```
*****
/* BAS: Mark baseline values */
*****

* Assign baseline flag;
proc sql;
  create table bas_1 as select
    a.*
  ,case
    when a.aeefl = 'Y' and b.abfl = 'Y' then 'Y'
  end as abfl
  from eli_f as a
  left join metadata.mdflow as b
    on a.avisitn = b.avisitn
    and a.topiccd = b.topiccd;
quit;
```

Program to check/generate header

```
*****
Description : This program generates the table ADaM.ADAE.
               The program contains the following sections:
               LIB: Setup libraries
               DAT: Retrieve SDTM datasets
               TRT: Retrieve treatment variables
               DTM: Create date/time variables
               DER: Data derivations
               FIN: Finalise dataset

Input       : adam.adsl
               sdtm.supdae
               sdtm.ds
               sdtm.ae
               sdtm.relrec

Output      : ADAM.ADAE
Usage notes :
Programmer  : CLBG
*****

/* LIB: Setup libraries */
*****

%access_v04();

*****

/* DAT: Retrieve SDTM datasets */
*****

%adam_add_supp_v01(domain = ae);

data dat_1;
  set ae;
  keep studyid usubjid aeseq aeterm aetech1 aesev aesor aeout aeterm
    aebodsys aere1 aeacn dictver aestdtc aeendtc aerefid aecat
    aedecod aeptcd aehtl aesoc;

run;

*****

/* TRT: Retrieve treatment variables */
*****

/* Considering variables from ADSL */

proc sql noprint;
  create table dat_f as
  select a.*
    ,b.subjid, b.siteid, b.sex, b.age, b.ageu, b.race, b.complfl
    ,b.randfl, b.ethnic, b.fasfl, b.saffl, b.trt01p as trtp
    ,b.trt01pn as trtpn, b.trt01a as trta, b.trt01an as trtan
    ,b.cohort, b.trtsdt, b.trtsdtm, a.aeacn as aacn, a.aere1 as arel
    ,a.aetech1 as atech1, b.bmibl, b.bmiblu, a.dictver as meddrav
  from dat_1 as a
  left join
    adam.adsl as b
    on a.usubjid = b.usubjid
  ;

quit;
```

Program header

Program body text

Make a sas-dataset from the SAS program, use to compare header with bodytext in the program

```
*****
FOUND_LINE
1  -----
2  Description : This program generates the table ADaM.ADAE.
3  The program contains the following sections:
4  LIB: Setup libraries
5  DAT: Retrieve SDTM datasets
6  TRT: Retrieve treatment variables
7  DTM: Create date/time variables
8  DER: Data derivations
9  FIN: Finalisedataset
10 Input       : adam.adsl
11 sdtm.supdae
12 sdtm.ds
13 sdtm.ae
14 sdtm.relrec
15 Output      : ADAM.ADAE
16 Usage notes :
17 Programmer  : CLBG
18 -----
19
20
21 -----
22 /* LIB: Setup libraries */
23 -----
24
25 %access_v04();
26
27
28 -----
29 /* DAT: Retrieve SDTM datasets */
30 -----
31
32 %adam add supp v01(domain = ae);
33
34 data dat_1;
35 set ae;
36 keep studyid usubjid aeseq aeterm aetech1 aesev aesor aeout aeterm aebodsys aere1 aea...
37 dictver aestdtc aeendtc aerefid aeacn aedecod aeptcd aehtl aesoc;
38 run;
39
40 -----
41 /* TRT: Retrieve treatment variables */
42 -----
43
44 /* Considering variables from ADSL */
45
46 proc sql noprint;
47 create table dat_f as
48 select a.*
49   ,b.subjid, b.siteid, b.sex, b.age, b.ageu, b.race, b.complfl, b.randfl, b.ethnic
50   ,b.fasfl, b.saffl, b.trt01p as trtp, b.trt01pn as trtpn, b.trt01a as trta, b.trt01an as trtan
51   ,b.cohort, b.trtsdt, b.trtsdtm, a.aeacn as aacn, a.aere1 as arel, a.aetech1 as atech1
52   ,b.bmibl, b.bmiblu, a.dictver as meddrav
53 from dat_1 as a
54 left join
55   adam.adsl as b
56   on a.usubjid = b.usubjid
57 ;
58 quit;
59
```

Example of syntax used to read the sas program into a sas-dataset and compare/generate the header out of the program and macro variables

```
*****
/* SRC: Save the program as a sas dataset */
*****

filename src "/path/folder/programname.sas";

data search_1;
infile src truncover;
input found_line $200.;
run;
```

Next step is to save the lines between the two first ***-lines into a variable called 'header_text' and below the 'header'- observations will be saved into the variable called 'body_text' and then the two columns could easily be compared with SAS syntax.

Example of syntax used to search for section names and description of section in the comments

```
*****
/* PRT: Find the section names for the
   different parts of programming syntax
   used in the program */
*****

data prt_1;
  set sec_f;
  found_line_1=compress(found_line, '/* ');
  found_line_2=substr((found_line_1),4,1);
  if found_line_2=":" then output;

run;

data prt_2(drop=header_text)
  prt_3(drop=body_text body_no);
  set prt_1;
  body_text=left(compress(body_text, '/* '));
  header_text=left(compress(header_text, '/* '));
  sect_abbr=substr(found_line_1,1,4);
  body_no=line_no;
  if section="body_text" then output prt_2;
  else output prt_3;
  keep section sect_abbr body_text
    header_text body_no;

run;

proc sort data=prt_2; by sect_abbr; run;
proc sort data=prt_3; by sect_abbr; run;

data prt_f(drop=section sect_abbr);
  length category $20.;
  retain sect_abbr header_text body_text;
  merge prt_2 prt_3;
  by sect_abbr;
  category="Sections";

run;

proc sort data=prt_f; by body_no; run;
```

Result file below where a finding is found, we have created a compare program where the result is presented in Excel format.

HEADER_TEXT	BODY_TEXT
LIB: Setup libraries	LIB: Setup libraries
DAT: Retrieve SDTM datasets	DAT: Retrieve SDTM datasets
TRT: Retrieve treatment variables	TRT: Retrieve treatment variables
DTM: Create datetime variables	DTM: Create datetime variables
DER: Data derivations	DER: Data derivations
FIN: Finalise dataset	FIN: Finalise dataset

Alternative approach: regular expressions

An alternative approach has been used, by implementing regular expressions in Perl.

```
while (defined (my $line = <IN>)) {
  chomp $line;
  $elements = split " ", $line;
  foreach $element ($elements) {
    if ($element =~ m/(ADAM\.[A-Z0-9]*)|(SDTM\.[A-Z0-9]*)|
      (DM\.[A-Z0-9]*)|(ADM\.[A-Z0-9]*)|(ADS\.[A-Z0-9]*)|
      (EXT\.[A-Z0-9]*)|(GCMD\.[A-Z0-9]*)/i) {
      $matched = $0;
      $matched =~ s/['a-zA-Z0-9.]/g;
      $input{uc($matched)} = 1;
    }
  }
}

print "Identified the following input datasets:";

foreach my $key (keys %input) {
  print " $key";
}
```

This method is very powerful, with a potential of advance usage opportunities. For example:

- it could be employed to search for more complex patterns, allowing for more flexibility in the .sas program.
- it could identify macro programs used, and scan those for any relevant information as well
- it can easily make a comparison and summary of header information across multiple programs