

TS-610

Basics of Report Writing in Version 6

It's often useful to have the contents of a SAS data set output to an external file. Depending on the options used while creating the file, a report can be generated so the user has total control of the placement of variable values, column headings as well as other text. One way to generate a report from within SAS is using the DATA step.

This information below is intended as an overview of writing to external files with special emphasis on report writing within the DATA step. An example is given at the end that illustrates many of the options discussed in the paper. For brevity, the information in this paper is specific to the MVS and PC operating systems only.

The essential components of the DATA step needed for creating a report are the following:

- `_NULL_` keyword on DATA statement prevents a new SAS data set from being created
- FILE statement specifies the external file that will be created
- PUT statement writes lines to the file
- PRINT option on FILE statement

Basic Questions to Consider Prior to Creating A File:

1. Writing information to a new file or an existing file?
2. Is the file to be fixed or variable length?
3. If fixed length, what's the length of each record? If variable length, what's the length of the longest record?
4. Is the file going to be a text file or a report?

Writing information to a new file or an existing file?

The location of the external file that will be written to can be referenced either in a FILENAME statement

```
FILENAME fileref <device type> 'external file specification' <host options>;
```

or FILE statement

```
FILE 'external file specification' <options>;
```

Under batch processing in MVS, the FILE statement can reference a pre-existing data set or a new data set allocated by the FILENAME statement or DD statement in the JCL. If referencing a new data set on the FILE statement not allocated by the FILENAME statement or TSO ALLOC statement while running interactively, you are prompted for allocation of a new data set.

The PC operating system allows referencing a pre-existing or new file on the FILE statement. If the FILE statement references a new file, it is automatically created in the specified path.

If the new information is appended to the file, the MOD option is placed on the FILE statement. Without MOD, the contents of an existing file with the same name are overwritten.

Is the file to be fixed or variable length?

Variable length files with RECFM=V are created by default on the PC unless the PRINT option is specified. In that case it creates a file with RECFM=P with carriage control characters in column 1. Fixed block files are created by default on MVS unless the PRINT option is specified, which will produce a VBA (variable block print) file.

To change the default file type, specify the RECFM= option on the FILE or FILENAME statement on the PC. Under MVS allocate the file using the RECFM= option on the FILENAME statement or use the RECFM= sub-parameter of the DCB parameter of the DD statement specifying the record format. When creating a print file, the file will be allocated as VBA or FBA. See the appropriate SAS Companion for your operating system for valid values for the RECFM= option.

It's important to remember that specifying RECFM=F along with the PRINT option on the FILE statement is incompatible on the PC. If you want the records to be fixed length but also want the file to be a PRINT file, then allow the file to default to variable length, specify the longest record length with the LRECL= option, and specify the PAD option. These options are specified on the FILE statement. This causes the records that aren't as long as the value of the LRECL= option to be padded with blanks.

If fixed length, what's the length of each record? If variable length, what's the length of the longest record?

The LRECL= option specifies the logical record length of the output file. If you do not use the LRECL= option, the SAS System chooses a value based on the host system file characteristics.

Under MVS, the first byte is reserved for the ANSI defined printer control characters when the RECFM=VBA or FBA. The LRECL must be defined equal to the length of the longest line plus one for the printer control characters for an FBA file. The LRECL must be defined equal to the length of the longest line plus four bytes for the RDW (record descriptor word) and one byte for the printer control character of a VBA file. The RDW is a four byte field at the beginning of each record that is hidden to the user but understood by the operating system.

The LINESIZE option is very important when creating files so make sure it is set appropriately when writing to a PRINT file. The maximum value of this option is 256 when creating a PRINT file. Under MVS if the LRECL= option is not specified, the value is determined by the current setting of the LINESIZE= option. If the PRINT option is used, a VBA file is allocated with an LRECL value of LINESIZE plus 5.

PAD is the default option when writing to a fixed length file. This causes all records to be padded with blanks to the length specified by the LRECL= option. On MVS, the option NOPAD is invalid. NOPAD is the default for variable-length files.

Is the file to contain carriage control characters?

Besides the end-of-record markers and RDW information, there are situations when other carriage control characters are needed. If the file you're creating is to be printed, the printer needs to know things such as: 1) how many lines to advance the paper, 2) when to begin a new page, 3) when to skip lines, and 4) when to hold the current line for overprint. These carriage control characters are inserted into the file when the option PRINT is used on the FILE statement.

The most typical way to create a report is by writing to an external file. The FILE statement contains the fully qualified path and filename, multilevel name, or previously defined fileref. If the file is to contain carriage control characters, the option PRINT is placed on the FILE statement after the fileref or file specification. When writing directly to an external file, the PRINT option causes the new file to contain carriage control characters. When creating a report with the MVS operating system, the file is allocated with an A suffix (FBA,VBA) to indicate the file is a print file. Otherwise the file will not contain carriage control characters even though the PRINT option is specified.

It's often helpful to see a preview of the report periodically during the creation phase without having to print the actual file (report). When PRINT is the fileref, the SAS System uses carriage control characters and writes the lines with the characteristics of a print file to the same print file as the output produced by SAS procedures (typically the output window if using SAS in the Display Manager mode). Remember you cannot view the results of OVERPRINT when writing to the OUTPUT window.

Various options are available on the FILE statement when creating PRINT files. Some of the most frequently used options are discussed below:

- *N=available lines*
specifies the number of lines you want available to the output pointer in the current iteration of the DATA step. *Available-lines* can be expressed as a number (n) or as the keyword PAGESIZE or PS.

n specifies the number of lines at a time that are available to the output pointer.
PS (PAGESIZE) specifies that the entire page is available to the output pointer.

There are two ways to control the number of lines available to the output pointer:

- The N= option
- The #n line pointer control in a PUT statement

If the N= option is not specified and no # pointer controls are used, one line is available (N=1). If N= is not used but there are # pointer controls, N= has the highest value specified for a # pointer control in any PUT statement in the current DATA step. This can cause unusual results in the output

file. It's recommended that if # pointer controls are used, an appropriate value for N= is selected and specified.

It's important to remember that when writing to a print file, the value of the N= option must be either 1 or PAGESIZE.

- **HEADER=label**
defines a statement label that identifies a group of SAS statements that you want to execute each time SAS begins a new output page. The first statement after the label must be an executable statement; otherwise, you can use any SAS statement in the group of statements labeled for execution with the HEADER= option.

To prevent the statements in this group from executing with each iteration of the DATA step, use two RETURN statements...one precedes the label and the other appears as the last statement in the group.

- **LINESLEFT=variable**
defines a variable whose value is the number of lines left on the current page. You supply the variable name; SAS assigns that variable the value of the number of lines left on the current page. This option is often used in conjunction with PUT _PAGE _ so that the page ejects occur at specific intervals.
- **LRECL=logical-record-length**
specifies the logical record length. If you do not use the LRECL= option, SAS chooses a value based on the host system's file characteristics.

When creating a print file on MVS operating system, it's important to remember that the first column is reserved for carriage control characters. This means that the first column of data begins in column 2 or after. Therefore the LRECL value must be the length of the longest line you are writing with a PUT statement plus one for the printer control character or if the file is a VBA file, add five instead of one.

When creating a PRINT file, if the LRECL= value is smaller than the value of the LINESIZE option, a warning will be issued as follows:

WARNING: Truncated record.

- **PAGESIZE=value**
sets the number of lines per page. After the value of the PAGESIZE= option is reached, the output pointer advances to line 1 of a new page.

PRINT is required on the FILE statement when customizing your report with any of the following:

- HEADER= option on the FILE statement
- TITLE statement
- OVERPRINT option on the PUT statement
- Specifying a page eject with PUT _ PAGE _

HEADER= (discussed on previous page)

TITLE statement

specifies title lines to be printed on SAS print files and other SAS output. Once you specify a title, it's used for all subsequent output until you cancel the title or define another title for that line. A TITLE statement for a given line cancels the previous TITLE statement for that line and for all lines with larger n numbers. The following statement suppresses a title on line *n* and all lines after it:

```
titlen;
```

The line reserved for TITLE1 also contains date and page number information. If you do not want this information printed on your report, specify the system options NODATE and NONUMBER.

OVERPRINT option

causes whatever follows the keyword OVERPRINT to be printed on the output line most recently written to by a PUT statement. It is not necessary to use a trailing @ to keep the output pointer on the same record. Overprint is only applicable when the N= option on the FILE statement is set to 1 AND you are writing to an external file; therefore, this option is not applicable when # line pointer controls are used. OVERPRINT will not give correct results if you're writing to the output window. It's necessary to print the file in order to see the correct line spacing that's produced from using OVERPRINT. Browsing the file with an editor will indicate incorrect line spacing unless you interpret the carriage control characters.

PUT _ PAGE _

advances the pointer to the first line of a new page. Although a new page automatically begins when a line exceeds the current PAGESIZE= value, it's often applicable to conditionally advance the pointer to a new page. Consecutive PUT _ PAGE _ statements will not cause multiple blank pages to be inserted in the report. Once the pointer is resting at the top of a blank page, it will not honor additional PUT _ PAGE_ statements. An additional line pointer control, _BLANKPAGE_, is applicable

in this instance. It advances the pointer to the first line of a new page, even when the pointer is currently located at the top left position of a new page.

All of the information contained in this paper is documented in "SAS Language Reference Version 6 First Edition" with the exception of PUT `_BLANKPAGE _`. This is documented in "SAS Technical Report P-222" on page 42.

EXAMPLE

An example of creating a multi-panel report is on the following page. Many of the features discussed in this paper are demonstrated in the example.

```

data a;
  do i=1 to 500;          /* sample data set */
    do j=1 to 2;
      output;
    end;
  end;
run;

```

```

options linesize=80 pagesize=60;

```

```

data one;
  set a;
  by i;
  retain column 0;
  file 'file specification goes here' print linesleft=num n=ps header=pagetop lrecl=80;
  /* COLUMN variable is initialized to 0 so until it changes, the following PUT statement will execute */
  if column=0 then do;
    put @7 i @22 j;
  end;
  /* when COLUMN=1 the following PUT statement will execute */
  else if column=1 then do;
    put @46 i @61 j;
  end;
  /* if there are 2 lines remaining on the page and COLUMN=0 then do the following */
  if num =2 and column=0 then do ;
  /* change the value of COLUMN to 1 */
    column=1;
  /* move the pointer to the top of the next column to position 43 */
    put #3 @43;
  end;
  /* if there are 2 lines remaining on the page and COLUMN=1, then we have written to the entire page.
  We now want to do a page eject and reset COLUMN back to 0 */
  else if num=2 and column=1 then do;
    put _page_;
    column=0;
  end;
return;
  /* the following "header" information will occur at the top of each new page */
pagetop:
  pagenum +1;
  do hcol = 4,43;
    put #1 @78 pagenum 2.;
    put #2 @hcol 'Value of I' +4 'Value of J';

```

```
    put #3 @hcol '_____' +4 '_____' ;  
end;  
run;
```