# The SB+ Cookbook Reportwriter Recipes

(more attractive graphic heading discarded for quicker downloads)

A Collection of Recipes
For Creating Reports
Using the SB+ ReportWriter

Written By

Kevin King
Precision Solutions, Inc.
6/15/1998 3:28:00 PM

http://www.PrecisOnline.com
Kevin@PrecisOnline.com

# Table of Contents

# Welcome to the Cookbook!

When we released the book SB+ Solutions[1] in the beginning of 1997, we had no idea it would become such a popular reference. Copies of this book have now been sold to more countries than we ever knew existed, and the feedback we've received to date has been overwhelmingly positive.

Despite all the positive feedback we've received, there has been one criticism we have received a few times. Apparently, it's difficult for me to say what needs to be said in just a few words. SB+ Solutions is *only* 600 or so pages (counting the index, table of contents, and separator pages), so perhaps there is some truth to the allegations. (Actually, the original text was approaching 800 pages before we selected a smaller font.)

As a result, I've decided to try something new with this text. Those who have been in one of my SB+ training courses might remember my inclination towards "recipes" -- standardized approaches to producing a consistent result. Therefore, why not take a collection of recipes and create a "cookbook"?

This particular cookbook contains much of the material from Chapter 6 of SB+ Solutions, plus perhaps a few tips and tricks that may not be in the original text. Of course, if we're going to have a cookbook, each recipe will contain ingredients to use and tasks to follow. To keep the volume of this document to a minimum, however, there won't be a great deal of detail about why a particular setting or task is being done. Therefore, if you have any questions about why something is being done, you may wish to consult "the big green book".

In the end though, the goal of all of these books is the same: Using these techniques, we want you to be able to create better software faster and with less effort.

This text is formatted on an 8 1/2 x 11" page, though I do not intend to fill up each page entirely. Therefore, if you see a lot of blank space, don't worry -- I've simply left this space for you to write your own ideas and opinions. Of course, if you come up with something really, really cool, you'll have to share it with me...

Also note that this document is a work in progress, and will be updated as new techniques are discovered and time is kind enough to allow their introduction.


[1] Affectionately known as "the big green book".

# SB+ ReportWriter Report Sections

A report definition consists of different types of lines. Groups of lines are called a *section*. As a section is processed, the lines are executed one by one from the first line in the section to the last, unless a conditional field is used to skip a group of lines. The following paragraphs explain the different types of sections and when they are executed.

*Heading* — The heading section defines the lines that will be printed at the top of each page. Therefore, this section is executed once for each page that is printed.

*Column Heading* — The column heading is an extension to the heading lines, and will be output once for each page, following the heading lines.

*Detail* — The detail section defines the fields to be output for each selected report. This section will be output once for each record that has been read.

*Break* — If the report has break fields defined, each time the value in one of the break fields changes from one record to the next, the break section will be executed. If a report has, say, two breaks and the major break field changes, the break section will be processed once for the minor break first, then again for the major break.

The break section will be executed once for each break field at the end of the report. If the report has no grand total section, the break section will be processed one additional time for the major break. Therefore, if you have a report with any breaks, it is important to include your own grand total section, instead of allowing SB+ to reexecute the major break through the break section.

*Grand Total* — The grand total section defines the final totals for the report. It is executed once after all of the selected records have been processed.

*Footing* — The footing section defines the lines to appear at the bottom of the page and is executed once for each printed page.

# Creating a Simple Columnar Report

Ingredients

- File containing information to be reported
- Field definitions in the file
- Optional: Report template

Tasks

1. Start the Report Definitions tool and optionally copy in a template report. The template report should contain a standard heading, footing, and blank lines for detail and break sections.

2. Place the fields to be output on the report using the F5-Field key. Add column headings to the fields as appropriate.

3. Add graphics to the report as appropriate.

4. Using the F6-Params screen, add sorting and selection criteria as appropriate. If the report will run from a select list, enter the name of the select list in the Selection Criteria prompt on this screen. If the report needs additional values that are not in the main record, you may add a Process After Read to load the additional values.

5. Run the Report with F2-Save, None-Execute Now to ensure the right values are being printed in the detail section.

6. Using the F6-Params screen, add the break fields to the report as appropriate. Always remember that the first field name listed is referred to as break #1, the second field name is break #2, etc.

7. Add a break section to the report for the break fields. When laying out the break section, define the break section in order of the break fields, with the most minor break defined first. Allow three break lines for the most minor break, and two break lines for all other breaks.

8. Add the conditional fields to the report for the breaks. Conditional fields are most easily named "C.*x*" (where *x* is some number, such as 1 or 2).

9. Test the report thoroughly.

Always remember: You can run the report as many times as you want to test even the most minor changes. When in doubt about the effect of a change, run the report and see for yourself!

# Adding a Derived Value Field to the Report

Ingredients

- A report definition where you want to add a calculated value

Tasks

1. Start the ReportWriter tool and create or load the report you want to edit.

2. Move the cursor to any line on the report where such a field is needed.

3. Press F5 to add a new field. When adding a new field, be sure to use a name that does not reference a field in the dictionary. Prefixes such as "R." or "D." can be used to make the name unique to the report (and also provide a good visual tool for identifying derived value fields).

4. When you press <cr> on the field name, SB+ will tell you the field is not found. At this prompt, enter either "A" (create an alphanumeric field), "D" (a date field), "N" (a numeric field), or "M" (a monetary field) followed by the number of digits you want for the field. For example, for an alphanumeric field with a length of 10 characters, "A10" can be entered. For a numeric field of 12 characters, "N12" can be entered.

5. Add a derived value to the report. This can be any value that can be calculated by an SB+ expression, including calling processes and conditional expressions using functions such as IF(...) or CASE(...).

# Creating a Report Template

Ingredients

- A "template file" where your report template will be stored.

Tasks

1. In your template file, create a report with heading, column heading, and footing sections. Create a blank detail section and optionally create blank break and grand total sections as well.

2. Create derived value fields to include whatever information you want in the heading and footing. Here are some useful bits of information that may be included:

| Item | Derived Value |
|---|---|
| The current page number | @RV.PAGE |
| The current date | @DATE |
| The current time | @TIME |
| The time the report started | @RV.START.TIME |
| The company name | @CONTROL<1> |
| The report description | F(@SYSID:'PROCESS',@PROC.NAME<1>)<2> |
| The report process name | @PROC.NAME<1> |

# Adding Grand Totals to a Columnar Report

Ingredients

- A report definition with fields you want to total

Tasks

1. Start the ReportWriter tool and create or load the report you want to edit.

2. Move the cursor to the first character of a field that you want totalled (on the detail line) and press F5.

3. Change the report field type from "N" (Normal) to "A" (Accumulated).

4. In the grand total section, create a new total field named "GT.*field*" (where *field* is the name of the field from step 2 (above). SB+ will tell you the field is not found. Enter "N10" for a numeric field, 10 characters wide, or "M10" for a monetary field of the same width, or some combination of either "N" or "M" followed by a field width. (Of course, the field width can be adjusted later.)

5. Press F2 to save the field.

# Adding Break Totals to a Report

Ingredients

- A report definition with fields you want to total

Tasks

1. Start the ReportWriter tool and create or load the report you want to edit.

2. Move the cursor to the first character of a field that you want totalled (on the detail line) and press F5.

3. Change the report field type from "N" (Normal) to "A" (Accumulated).

4. In the break section, create a new total field named "T.*field*" (where *field* is the name of the field from step 2 (above). SB+ will tell you the field is not found. Enter "N10" for a numeric field, 10 characters wide, or "M10" for a monetary field of the same width, or some combination of either "N" or "M" followed by a field width. (Of course, the field width can be adjusted later.)

   Note: The total field (T.*field*) can appear any number of times in a break section.

5. Press F2 to save the total field.

# Adding Conditional Totals to a Columnar Report

Ingredients

- A report definition
- A full understanding of the conditions under which a value will be accumulated for total

Tasks

1. Start the ReportWriter tool and create or load the report you want to edit.

2. On a detail line, move the cursor to a blank spot and press F5 to add a new field. When naming the field, be sure to use a name that doesn't exist in the dictionary. SB+ will tell you the field is not found. Enter "A1" to create an alphabetic prompt that is 1 character in width. (This value will not be displayed on the detail line, so its type and length are insignificant.)

3. On this new field, the Report Field Type must be "A" (Accumulated). The Conversion should be quotes surrounded by parentheses: ("") -- this will suppress the field from outputting anything on the report.

4. In the derived value, use either the IF(...) function or a process call to check a condition and accumulate a particular value for total. For example, if you want to create a total of only people with STATUS = "A":

   IF(STATUS="A",1,0)

   If you want to create a total of the customer balances for people with status not equal to "A":

   IF(STATUS#"A",BALANCE,0)

   This will cause the *truevalue* in the IF(...) expression to be rolled up only when the condition is true.

   Note: For a record counter, simply use a derived value of "1" (quotes shown for clarity, not required for syntax).

5. Add either a break total or grand total as described earlier.

# Creating a Conditional Detail Line

Ingredients

- A report definition with multiple detail lines

Tasks

1. Start the ReportWriter tool and create or load the report you want to edit.

2. On each detail line that is to be printed only under certain conditions, move the cursor to a spot o the line where you can place a 1 character field and press F5. Create a field named "C.1"[1].

3. Press <cr> a couple extra times until a smaller window opens up on top of the F5-Field window. This will ask you to enter a conditional expression. In this expression, enter either an IF(...) function or process call which will determine the number of lines to be skipped.

   For example, if the current detail line is to be skipped when STATUS = "A":

   IF(STATUS="A",1,0)

   Or, if this and the following line are to be skipped when a balance exceeds 100.00:

   IF(BALANCE>10000,2,0)

Note: If accumulated fields follow a conditional field and the conditional causes the line to be skipped, the fields will not be accumulated. If the accumulated fields are before the conditional field, they will be accumulated.

[1] If C.1 has been used, try C.2, C.3, or whatever number has not been used.

# Creating a Columnar Report with a Single Break

Ingredients

- A report definition with a working detail section
- A field in the dictionary that you can sort and break on (not a work field)

Tasks

1. Start the ReportWriter tool and create or load the report you want to edit.

2. Press F6 to look at the report parameters. Add the appropriate field to the sort and break fields. (You may also specify additional sort fields.) Press F2 to save the F6-Params information.

3. Add a break section to the report (following the detail section) with three lines. The first and last lines will be blank, and the break and total field(s) (if applicable) will be placed on the second line.

4. Move the cursor where you want the break field[1] to be placed. Press F5. Enter the name "B.*field*" (where *field* is one of the fields shown in the Break Fields prompt on the F6-Params window). SB+ will tell you the field is not found. Enter "A10" for an alphabetic field that is 10 characters in length, or "D9" for a nine-character date, or any other combination of "A", "D", "N", or "M" followed by the length of the field.

5. Press F2 to save the break field.

6. Place total fields in the break (as required) as described earlier.

[1] For example, if you're breaking on a STATE field, you may want to show the state code next to the total for that state when the state changes. This state code field is the 'break field'.

# Creating a Columnar Report with Several Breaks

Ingredients

- A report definition with a working detail section
- Fields in the dictionary that you can sort and break on (not work fields)

Tasks

1. Start the ReportWriter tool and create or load the report you want to edit.

2. Press F6 to look at the report parameters. Add the appropriate fields to the sort and break fields, in major (least often) to minor (most often) order[1]. (You may also specify additional sort fields.) Take a mental snapshot of these fields, noting the first as break field #1, the second as break field #2, etc. Then, press F2 to save the F6-Params information.

3. Create a break section for the report. Multiple the number of breaks you want by 2 and add 1. This is the number of lines that should be created in your break section. (For example, if you have 4 breaks, you'll want 9 lines in your break section.)

4. Mentally picture the first three lines for the most minor break, the next two for the next intermediate break, continuing two lines at a time to the major break.

5. Setup each break as follows. For the minor break (the last break field on the F6-Params window and the first 3 lines of your break section), place the break and total fields (as described elsewhere) on the 2nd line. For all other breaks, place the break and total fields on the 1st of the two lines for the break "subsection".

6. On the first line of the break section (which is also the first line of the minor break subsection), move the cursor anywhere on the line and press F5 to add a conditional field. Enter the field name of "C.1"[2]. Press <cr> a couple of times until the little window pops up over the F5-Field window. In this window, enter the following conditional expression:

   IF(@RV.BREAK.FLD=*n*,0,3)

   In this syntax, *n* is the number of the break field you're currently working in. As this is the most minor break, if there are a total of 2 breaks for this report, then *n* = 2. Press F2 to save this field.

7. On the first line of the next break subsection (line 4, usually), move the cursor anywhere on the line and press F5 to add a conditional field. Enter the name of "C.2". Press <cr> a couple of times until the little window pops up over the F5-Field window. In this window, enter the following conditional expression:

   IF(@RV.BREAK.FLD=*n*,0,2)

   In this syntax, *n* is the number of the break field you're currently working in. If this is the major break, then *n* = 1. If this is an intermediate break (between the major and minor breaks), then *n* = the break number as noted on the F6-Params window[3]. Press F2.

---

[1] If you want a report which breaks on state and status within state, state is the major sort/break, status is the minor sort/break.
[2] If C.1 has been used, try C.2, C.3, or whatever number has not been used.
[3] The first break field noted on the F6-Params window is 1, the second is 2, etc.

# Creating a Columnar Summary Report

Ingredients

- A file containing fields to be summarized

Tasks

1. Start the ReportWriter tool and create a new report. You may want to load a report template if you have one.

2. In the F6-Params window, set "Totals Only" to "Y". This will suppress the detail lines from being printed.

3. On the detail line of the report, add all of your accumulated fields. Nothing else needs to be on this line, and the formatting of the line is insignificant (because it won't be printed). Also, be sure to use a Description Source of "N" so the accumulated fields aren't reflected on the column heading.

4. Create a break section for the report containing 1 line. Add the appropriate break and total fields (as explained earlier).

5. Create a grand total section for the report as required.

6. Type your own column headings onto the report above your break or grand totals.

# Creating a Summary at the end of a Detail Report

Ingredients

- A report definition with something that you can summarize
- A paragraph to do the summarizing (explained below)

Tasks

1. Start the ReportWriter tool and create or load a detail report that you want summarized at the end.

2. Add a paragraph in the Proc At Start as follows:

   @PARMS(2) = "" ;* The summary will be updated into @PARMS(2)[1]

3. Add a paragraph in the Proc After Read as follows:

   ```
   LOCAL NDX
   *
   NDX = LOC(summaryfield,@PARMS(2),@VM)
   IF NDX = 0 THEN
      NDX = DCOUNT(@PARMS(2)<1>,@VM) + 1
      @PARMS(2)<1,NDX> = summaryfield
   END
   *
   @PARMS(2)<2,NDX> = @PARMS(2)<2,NDX> + firstsummaryvalue
   @PARMS(2)<3,NDX> = @PARMS(2)<3,NDX> + secondsummaryvalue
   ...continue as necessary
   ```

   In this paragraph, fill in the italicized names with the appropriate fields. The field named *summaryfield* will be the field that you are summarizing based on, and *firstsummaryvalue, secondsummaryvalue,* etc., will be the first, second, and subsequent fields you want accumulated for a summary.

4. In the grand total section, add new derived value fields to output the values stored in @PARMS(2). For the *summaryfield*, create a new field for the report with a derived value of @PARMS(2)<1>. For the *firstsummaryvalue*, create a new field for the report with a derived value of @PARMS(2)<2>. On all fields, be sure to set Suppress Repetition to "L" and Max MV Lines to "0.M".

[1] If @PARMS(2) is unavailable, substitute with another @PARMS(...) element which is not being used for another purpose.

# Stripping Information From the Record

Ingredients

- A report against a file with controlling and dependent multivalued information
- A paragraph to strip information from the record (explained below)

Tasks

1. Start the ReportWriter tool and create or load a detail report with the multivalued information you want displayed.

2. To strip certain multivalues out of the record, call a Process After Read which removes the information from the record that you don't want shown on the report. For example, if the report is to strip out the first value in a PRICES field, create a paragraph as follows:

   ```
   @RECORD = DEL(@RECORD,POS(PRICES),1)
   ```

   You may also need to use the LOC(...) to find a particular multivalue position and then remove the appropriate values, such as in this example:

   ```
   LOCAL NDX
   *
   NDX = LOC('X',PRICE.CATEGORIES,@VM)
   IF NDX THEN
      @RECORD = DEL(@RECORD,POS(PRICE.CATEGORIES),NDX)
      @RECORD = DEL(@RECORD,POS(PRICES),NDX)
   END
   ```

   Or, if 'X' is the only price category to be shown:

   ```
   LOCAL NDX
   *
   NDX = LOC('X',<POS(PRICE.CATEGORIES)>,@VM)
   IF NDX THEN
      <POS(PRICE.CATEGORIES)> = <POS(PRICE.CATEGORIES),NDX>
      <POS(PRICES)> = <POS(PRICES),NDX>
   END ELSE
      EXIT 1 ;* Skip this record
   END
   ```

# Creating a Simple Form Report

Ingredients

- File containing information to be reported
- Field definitions in the file

Tasks

1. Start the Report Definitions tool and optionally copy in a template report. The report template should contain a standard detail section for the form.

2. Place the fields to be output on the report using the F5-Field key. Add labels to the fields as appropriate.

3. Add graphics to the report as appropriate.

4. Using the F6-Params screen, add sorting and selection criteria as appropriate. If the report will run from a select list, enter the name of the select list in the Selection Criteria prompt on this screen. If the report needs additional values that are not in the main record, you may add a Process After Read to load the additional values.

5. Run the Report with F2-Save, None-Execute Now to ensure the right values are being printed in the detail section.

6. Test the report thoroughly.

Always remember: You can run the report as many times as you want to test even the most minor changes. When in doubt about the effect of a change, run the report and see for yourself!

# Implementing Impossible Selection Criteria

Ingredients

- A report definition in which the selection criteria cannot be accomplished using the myriads of options available with the Selection Criteria prompt.

Tasks

1. Start the ReportWriter tool and load or create a report.

2. Use Selection Criteria (on the F6-Params window) to narrow down the list as much as possible. If it's simply impossible to use Selection Criteria, leave this prompt blank.

3. Using the Process After Read, call a paragraph or BASIC subroutine which determines if each record is to be printed. If the record is to be printed, set (or leave) @RTN.FLAG as zero. If the record is to be skipped, set @RTN.FLAG to 1. This allows you to use whatever conditions you require to determine if a single record should be printed or skipped.