Paper TT21

Hyperlinks and Bookmarks with ODS RTF

Scott Osowski, PPD, Inc, Wilmington, NC
Thomas Fritchey, PPD, Inc, Wilmington, NC

ABSTRACT

The ODS RTF output destination in the SAS® System opens up a world of formatting and stylistic enhancements for your output. Furthermore, it allows you to use hyperlinks to navigate both within a document and to external files. Using the STYLE option in the REPORT procedure is one way of accomplishing this goal and has been demonstrated in previous publications. In contrast, this paper demonstrates a new and more flexible approach to creating hyperlinks and bookmarks using embedded RTF control words, PROC REPORT, and the ODS RTF destination.

INTRODUCTION

The ODS RTF output destination in SAS allows you to customize output in the popular file Rich Text Format (RTF). This paper demonstrates one approach to using the navigational options available in RTF files from PROC REPORT by illustrating its application in two examples, a data listing and an item 11 data definition table (DDT). Detailed approaches using the STYLE option available within PROC REPORT are widely known and have been well documented. These approaches are great if you want the entire contents of selected cells designated as external hyperlinks. However, the approach outlined in this paper allows you to have multiple internal and/or external hyperlinks embedded in any text in the report and offers options not available when using the style statement. This flexibility does come with a price, as the code necessary is lengthier than the alternative. It's up to you to decide which method is right for your output.

OVERVIEW OF THE RICH TEXT FORMAT SPECIFICATION

The RTF specification is a markup language broken up into groups marked by braces ({}) and control symbols/words marked by a preceding backslash (\) with no maximum line length. This paper does not attempt to provide an overview of various RTF control words but will briefly explain those related to creating bookmarks and hyperlinks.

`\bkmkstart`	– the start of the specified bookmark
`\bkmkend`	– the end of the specified bookmark
`\field`	– designates a field destination
`\fldinst`	– field instructions
`\fldrslt`	– field result

DEFINING BOOKMARKS

Although we use the hyperlink and bookmark mechanisms everyday as we navigate the world of electronic documentation, if you are new to the terms *bookmark* and *hyperlink* think of the hyperlink as the link you click on and the bookmark as the destination where that click takes you. Before you are able to use an internal hyperlink you must have an internal destination defined by a bookmark. You can manually create, examine, and delete bookmarks by selecting **Bookmark** from the **Insert** menu. Using SAS, these bookmarks can be created using RTF control words embedded in text to be displayed by PROC REPORT. It's important to note that PROC REPORT automatically makes a bookmark named IDX# for every output table produced. Thus, if you call PROC REPORT three times, your output RTF file will contain bookmarks IDX1, IDX2, and IDX3, marking the beginning of each output RTF table. What if you need additional bookmarks within an RTF table, or you desire more descriptive bookmark names? The following syntax allows you to create an RTF bookmark:

`{\*\bkmkstart` *bookmark_name*`}`*display_text*`{\*\bkmkend` *bookmark_name*`}`

The bookmark will have the name *bookmark_name* and have as its target the text *display_text*. Usually only the display text, not the bookmark, is visible when viewing an RTF document. However, you can instruct Microsoft Word to show bookmarks from the **View** tab under **Options** in the **Tools** menu. Bookmark names must begin with a letter and may not contain spaces – use the underscore character (_) instead. You should limit the length of bookmark names to 40 characters or less, and avoid special characters such as backslashes (\) and braces ({}).

As an example of one possible use for bookmarks and hyperlinks, let's produce a mock adverse event (AE) listing. Before presenting our mock data, let's output all our footnotes on the first page and mark each with an appropriate bookmark.

First, let's redirect output to ODS RTF and set our page orientation to landscape:

```
ods listing close;
ods rtf file = "AE_Listing.rtf";
ods escapechar = '^';
options orientation = landscape;
```

Second, we'll put our footnotes for the first page into a data set. There are several methods to accomplish this, but we'll use a CARDS statement. Note that there are no indented lines after the CARDS statement – lines are continuous and only appear wrapped due to width limitations.

```
data page1;
    length line $200.;
    infile cards missover length = lg dlm = '$';
    input line $varying200. lg;
    cards;
{\*\bkmkstart Outcome}[1] Outcome: 1=Resolved, 2=Resolved with sequelae, 3=Continuing,
    4=Fatal.{\*\bkmkend Outcome}
{\*\bkmkstart TreatmentRequired}[2] Any Treatment Required: 0=None, 1=Concomitant
    medications, 2=Non-drug therapies, 3=Concomitant medications and non-drug
    therapies.{\*\bkmkend TreatmentRequired}
{\*\bkmkstart Severity}[3] Severity: 1=Mild, 2=Moderate, 3=Severe.{\*\bkmkend Severity}
{\*\bkmkstart ActionTaken}[4] Action Taken: 0=None, 1=Study drug interrupted, 4=Study drug
    discontinued.{\*\bkmkend ActionTaken}
{\*\bkmkstart RelationToDrug}[5] Relation to Study Drug: 1=Not Related, 2=Unlikely,
    3=Possible, 4=Probable, 5=Definite.{\*\bkmkend RelationToDrug}
{\*\bkmkstart Serious}[6] Serious: 0=No, 1=Yes.{\*\bkmkend Serious}
;
run;
```

Now we'll output the first page of footnotes:

```
proc report data = page1 nowd missing;
    columns line;
    define line /display "Footnotes:" style = [cellwidth = 11 in just = left];
run;
```

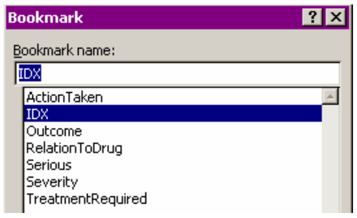
Finally, we need to close the ODS RTF output so that we can examine it:

```
ods rtf close;
```

Note the following important points about the above example:

- each line has been given a unique, descriptive bookmark name;
- the entire line is marked with a bookmark.; and
- any portion of each line could have been marked with a bookmark or could contain multiple bookmarks.

After submitting this code and opening the RTF output in Microsoft Word, click **Bookmark** from the **Insert** menu and you should see a list of our bookmarks.



DEFINING INTERNAL HYPERLINKS

As the name suggests, internal hyperlinks take you to a location within an RTF document by means of a previously defined bookmark. Similar to a bookmark, you can create an internal hyperlink in Microsoft Word by: 1) selecting **Hyperlink** from the **Insert** menu, 2) choosing **Place in This Document**, and finally, 3) choosing the destination bookmark. This can also be done using PROC REPORT giving you the flexibility to create internal hyperlinks almost anywhere in your RTF document. The following syntax allows you to create an internal hyperlink:

{\field {*\fldinst HYPERLINK \\l "*bookmark_name*"}{\fldrslt *display_text*}}

Let's break down this syntax into parts. First, the `\field` control word designs the entire string as something to be calculated or resolved by the RTF reader. Next, the `\*` control symbol instructs the RTF reader to ignore the next control word if it's not recognized. The `\fldinst` control word indicates the instructions for this field, and `HYPERLINK` tells the RTF reader that we're going somewhere. This next part is a little tricky. To let Microsoft Word know that our destination is an internal bookmark instead of an external file, we have to use `\l`. However, we know that the backslash (\) is reserved by RTF to mark control words and symbols. To protect our backslash, we simply use two of them, hence the `\\l` followed by the destination bookmark (*bookmark_name*) in double-quotes. Finally, the `\fldrslt` control word indicates the text that should appear in the document, which we've marked as *display_text*.

Although this may seem complicated at first, once understood it gives you the power to place hyperlinks almost anywhere. In fact, you can place multiple hyperlinks with different destinations within a single cell of text. Before we proceed to an example, let's first discuss an optional part of hyperlinks – screen tips.

If you hover your mouse over a hyperlink created in Microsoft Word you will notice that a screen tip appears. By default it will display the destination of the hyperlink with the bolded instruction **Click to follow link**. Although the instruction cannot be modified, the text above it can be by using the `\o` option, as demonstrated below.

{\field {*\fldinst HYPERLINK \\l "*bookmark_name*" \\o "*screen_tip*"}
    {\fldrslt *display_text*}}

Note that the screen tip must be enclosed with double-quotes, and, because it is an option within Microsoft Word, we must protect the backslash (\) by doubling it (\\). The length of screen tips should be limited to 256 characters.

Let's continue our AE listing example by using hyperlinks to link uncoded variable values to their respective footnotes containing the decoded values. Furthermore, let's use the screen tip option to allow the user to see the decoded value without requiring them to view the footnote page. The following code should be executed immediately before the final statement closing the ODS RTF output in the previous example.

First, let's set up our formats for uncoded variables to match our previous footnotes:

```
proc format lib = work.formats;
    value out 1 = 'Resolved'
        2 = 'Resolved with sequelae'
        3 = 'Continuing'
        4 = 'Fatal';
    value trt 0 = 'None'
        1 = 'Concomitant Medications'
        2 = 'Non-Drug Therapies'
        3 = 'Concomitant Medications and Non-Drug Therapies';
    value sev 1 = 'Mild'
        2 = 'Moderate'
        3 = 'Severe';
    value act 0 = 'None'
        1 = 'Study drug interrupted'
        4 = 'Study drug discontinued';
    value rel 1 = 'Not Related'
        2 = 'Unlikely'
        3 = 'Possible'
        4 = 'Probable'
        5 = 'Definite';
    value ser 0 = "Not a Serious Adverse Event"
        1 = "Serious Adverse Event";
run;
```

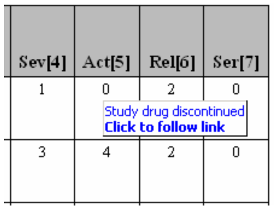
Second, we'll use a cards statement to input some dummy data. We'll also add our hyperlinks to each observation for each uncoded variable. Instead of doing this in separate statements, we'll use an array to handle all six variables at once. Notice that the statement assigning a value to the ith element of the charcodes array uses the bookmark name (bmkcodes array) as the destination, the formatted value of the variable (using the format from the fmtcodes array) as the screen tip, and the variable value itself (numcodes array) as the display text.

```
data ae;
    length asr $50.
        aeseq 8.
        prefterm $700.
        dt $30.
        aeout aecontrt aesev aeacttrt aerel aeser 8.;
    infile cards missover dlm = '$';
    input asr aeseq prefterm dt aeout aecontrt aesev aeacttrt aerel aeser;
    array numcodes{*} aeout aecontrt aesev aeacttrt aerel aeser;
    array charcodes{*} $200. aeoutc aecontrtc aesevc aeacttrtc aerelc aeserc;
    array fmtcodes{6} $3. _TEMPORARY_ ('out' 'trt' 'sev' 'act' 'rel' 'ser');
    array bmkcodes{6} $20. _TEMPORARY_ ('Outcome' 'TreatmentRequired' 'Severity'
        'ActionTaken' 'RelationToDrug' 'Serious');
    do i = 1 to dim(fmtcodes);
        charcodes{i} = '{\field {\*\fldinst HYPERLINK \\l "' ||
            compress(bmkcodes{i}) || '" \\o "' ||
            trim(left(putn(numcodes{i}, fmtcodes{i}))) ||
            '"}{\fldrslt ' || compress(put(numcodes{i}, 8.)) || '}}';
    end;
    cards;
103/ 2003/ 50/ Male/ White$2$Musculoskeletal and connective tissue disorders/^nPain in
        extremity/^nPAIN IN LEGS$14APR2005/^n30APR2005$1$0$1$0$2$0
103/ 2003/ 50/ Male/ White$1$Nervous system disorders/^nHeadache/^nINTERMITENT
        HEADACHES$14APR2005/^n30APR2005$1$0$3$4$2$0
103/ 2004/ 65/ Female/ Black or African American$1$Gastrointestinal
        disorders/^nDiarrhoea/^nDIARRHEA$25APR2005/^n26APR2005$1$1$2$0$1$0
103/ 2008/ 55/ Female/ Asian$2$Respiratory, thoracic and mediastinal
        disorders/^nEpistaxis/^nNOSE BLEEDING$30MAY2005/^n30MAY2005$1$0$2$0$2$0
103/ 2008/ 55/ Female/ Asian$1$Nervous system
        disorders/^nDizziness/^nDIZZINESS$31MAY2005/^n31MAY2005$1$1$2$0$2$0
;
run;
```

Finally, we'll use PROC REPORT to output the RTF table. Remember to close the ODS RTF output after this step.

```
proc report data = ae nowd missing split = '$' style(report) = [just = left];
    columns asr aeseq prefterm dt aeoutc aecontrtc aesevc aeacttrtc aerelc aeserc;
    define asr /order order = internal "Investigator/^nPatient No./^nAge/ Sex/ Race"
        style = [cellwidth = 1.6 in just = left];
    define aeseq /display "AE No." style = [cellwidth = .4 in just = center];
    define prefterm /display 'System Organ Class[1]/^nPreferred Term[1]/^nAdverse Event'
        style = [cellwidth = 3.4 in just = left];
    define dt /display "Start Date/^nStop Date" style = [cellwidth = 1.1 in just = center];
    define aeoutc /display 'Out[2]' style = [cellwidth = .55 in just = center];
    define aecontrtc /display 'Trt[3]' style = [cellwidth = .55 in just = center];
    define aesevc /display 'Sev[4]' style = [cellwidth = .55 in just = center];
    define aeacttrtc /display 'Act[5]' style = [cellwidth = .55 in just = center];
    define aerelc /display 'Rel[6]' style = [cellwidth = .55 in just = center];
    define aeserc /display 'Ser[7]' style = [cellwidth = .55 in just = center];
run;
```

Our listing should now contain our page of footnotes followed by our data. Furthermore, if you hover your mouse over any of the uncoded values in the data you should see a screen tip containing the formatted value. Clicking the uncoded value will take you to the associated footnote on the first page.



Sev[4]	Act[5]	Rel[6]	Ser[7]
1	0	2	0
3	4	2	0

Study drug discontinued
Click to follow link

DEFINING EXTERNAL HYPERLINKS

In addition to internal hyperlinks, your RTF output can also contain links to other files specified by absolute or relative paths. An absolute path is one that remains constant regardless of the document location. Examples of this would be "c:\mydocument.doc," "http://mysite.com/," etc. A relative path allows you to link to files in the same area relative to the original document, no matter where the file is located. This is very useful when you are distributing a group of files to other users and you don't know where the users might place them. If your links are relative and the files are located in the same folder or folder structure then the original links will still work. The following syntax allows you to create an external hyperlink with any destination and any display text you choose:

```
{\field {\*\fldinst HYPERLINK "filename_and_path"}{\fldrslt display_text}}
```

The hyperlink will open the file specified by *filename_and_path* and the actual link will read *display_text*. Remember that the backslash character (\) is used to denote RTF control words and symbols; any backslashes used in the file path must be protected by doubling them (\\) as previously mentioned.

For the final step in our AE listing example, let's suppose we also have created an AE table that summarizes the AE data. We plan to make reference to our AE listing as source data in a footnote, but let's take it a step further and create an external hyperlink to the listing file, assuming that the listing resides in the same folder as the table (so we'll use a relative path).

As before, we'll start with redirecting output to ODS RTF and setting our page orientation to landscape:

```
ods rtf file = "AE_Table.rtf";
options orientation = landscape;
```

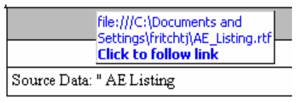
Next, let's use a dummy data set to represent our summarized data since we're only interested in the hyperlink.

```
data aetable;
    summary_data = '<Summarized Table Data>';
    output;
run;
```

Now we'll use a COMPUTE block in PROC REPORT to create our footnote containing the hyperlink, then close the ODS RTF output.

```
proc report data = aetable nowd missing split = '$';
    columns summary_data;
    define summary_data /display "Table Data" style = [cellwidth = 11 in just = center];
    compute after _page_/style = [protectspecialchars = off];
        line @1 'Source Data: {\field {\*\fldinst HYPERLINK "AE_Listing.rtf"}{\fldrslt AE
            Listing}}';
    endcomp;
run;
ods rtf close;
```

After executing this code, our mock AE table should contain a source data footnote and hyperlink. Since we didn't specify a screen tip, Microsoft Word defaults to show us the file to which the hyperlink is pointing. Although this looks like an absolute path, inspection of the hyperlink options in Word or the RTF control words will show that it is a relative path. Furthermore, if you move the document the default screen tip generated by Word will change.



file:///C:\Documents and Settings\fritchtj\AE_Listing.rtf
Click to follow link

Source Data: " AE Listing

But wait, something is amiss! The footnote reads: Source Data: " AE Listing. Where is the errant double-quote coming from? And another thing, why did we have to turn off the `protectspecialchars` option in the style section of the COMPUTE block?

THE COMPUTE BLOCK EXCEPTION

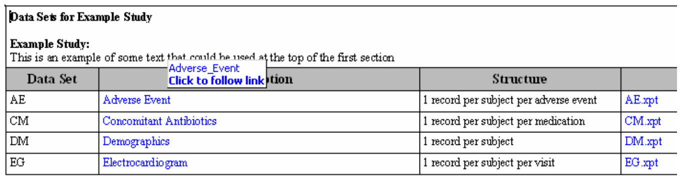
There is a distinct difference between the RTF output generated by a COMPUTE block and that generated by other aspects of PROC REPORT. Specifically, COMPUTE block output doesn't contain spaces. Instead, nonbreaking spaces fill the output, designated by the RTF control symbol \~. Indeed, when RTF code is embedded into COMPUTE block output, even the spaces separating control words from data are converted to nonbreaking spaces.

Since we know of no options that will force true spaces to be used in a COMPUTE block, our approach to overcoming this behavior is to post-process the RTF file and use the TRANWRD function to convert nonbreaking spaces (\~) to true spaces. We do not suggest embedding any RTF code in a COMPUTE block without post-processing the RTF file. Using Microsoft Word 2002, we have observed that bookmark names may be prefixed with "BM_" and hyperlink display text may include a double-quote, as in our previous example. However, because true spaces are expected as delimiters between RTF control words and data, different RTF readers and versions may handle this aberrant code very differently, perhaps even refusing to open the file.

One last thing to note about COMPUTE blocks is that the `protectspecialchars` option must be turned off if RTF code is to be successfully embedded. If this option is not set to off, the backslashes marking the beginning of control words/symbols will be internally doubled in the RTF output by SAS causing the RTF reader to interpret the RTF code as text.

EXAMPLE – ITEM 11

Now that you've seen how to create hyperlinks and bookmarks in a listing, we'll go through an example that creates the DDT section of an item 11 electronic case report tabulation (see Appendix for code). For this simple example we'll create some dummy data sets and we'll use the output from the CONTENTS procedure to populate the DDT. Using the output data set from PROC CONTENTS, you'll need to perform a few modifications and additions to produce the example DDT. First, you'll need to add the structure of each data set; this will populate the structure column and for this example is done manually with a CARDS statement. Next, you'll need to add the RTF control words to the variables that will enable you to hyperlink internally to a bookmark we'll create in a later section and to a variable that links externally to transport files listed in the location column. Now that we have all of the required variables for our database description table, we can open the ODS RTF destination, and output a table of the database metadata using PROC REPORT.

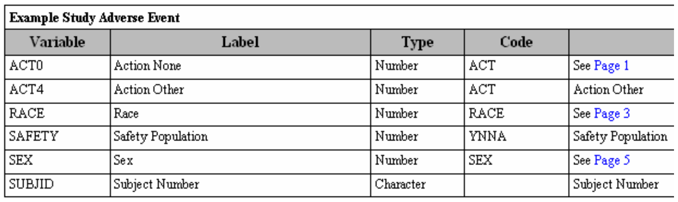


Data Sets for Example Study

Example Study:
This is an example of some text that could be used at the top of the first section

Data Set	tion	Structure	
AE	Adverse Event	1 record per subject per adverse event	AE.xpt
CM	Concomitant Antibiotics	1 record per subject per medication	CM.xpt
DM	Demographics	1 record per subject	DM.xpt
EG	Electrocardiogram	1 record per subject per visit	EG.xpt

The next step in the process is creating the data set description section of the DDT. For this section, you'll need to have a page variable that represents the CRF page on which the variable was annotated as well as the variable type from PROC CONTENTS. For the sake of simplicity, we use a counter to assign dummy pages. The next step in the example is to add the RTF code to create a relative external hyperlink using the aforementioned page variable to link to a specfic page on the blankcrf.pdf file. This example code assumes that destinations have already been created in this file using the form "Page_*pagenumber*", and that the file exists in your relative directory. Now that we have all of the required variables, we can use PROC REPORT to output to the same RTF destination opened before, except this time we'll use the COMPUTE block to create a bookmark using the data set label. This is the same variable used to create the hyperlink in the previous section of the SAS code. As described above we'll need to post-process the RTF file to remove the nonbreaking spaces from RTF code generated by this COMPUTE block.



Example Study Adverse Event

Variable	Label	Type	Code	
ACT0	Action None	Number	ACT	See Page 1
ACT4	Action Other	Number	ACT	Action Other
RACE	Race	Number	RACE	See Page 3
SAFETY	Safety Population	Number	YNNA	Safety Population
SEX	Sex	Number	SEX	See Page 5
SUBJID	Subject Number	Character		Subject Number

CONCLUSION

Since there is an ever-increasing demand to efficiently navigate electronic documents, the need for hyperlinks and bookmarks is evident. Though many options are available, few offer the immense flexibility of using embedding RTF control words and the ODS RTF destination. Though demonstrated using listing, table, and item 11 output, the value and utility of this approach is only limited by your imagination.

RECOMMENDED READING

Microsoft Corporation. "Microsoft Office Word 2003 Rich Text Format (RTF) Specification." April 2004. <http://www.microsoft.com/downloads/details.aspx?FamilyID=ac57de32-17f0-4b46-9e4e-467ef9bc5540&DisplayLang=en> (February 13, 2006).

Robert Walls (2005). "Using ODS RTF with Style!" *Proceedings of Pharmaceutical Users Software Exchange*, October 2005, SS03.

CONTACT INFORMATION

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

Scott Osowski
Senior Programmer Analyst
3151 South 17th St.
Wilmington, NC 28412
Work Phone: (910) 772-7546
Fax: (919) 379-6199
Scott.Osowski@wilm.ppdi.com

Thomas Fritchey
Senior Programmer Analyst
3151 South 17th St.
Wilmington, NC 28412
Work Phone: (910) 772-7148
Fax: (919) 654-8637
Thomas.Fritchey@wilm.ppdi.com

APPENDIX: EXAMPLE ITEM 11 DATA DEFINITION TABLE CODE

```
/* Close the listing output destination and set general options */
ods listing close;
options missing = '' orientation = landscape nofmterr nodate nonumber;
title;

/* Open the RTF destination */
ods escapechar = '^';
ods rtf file = "define.rtf";

/* Create Some dummy data sets */
/* Dummy AE */
data ae;
    attrib ACT0 label = "Action None" format = ACT. length = 8.
        ACT4 label = "Action Other" format = ACT. length = 8.
        RACE label = "Race" format = RACE. length = 8.
        SAFETY label = "Safety Population" format = YNNA. length = 8.
        SEX label = "Sex " format = SEX. length = 8.
        SUBJID label = "Subject Number" length = $3.;
    if _n_ = 0;
run;
/* Dummy CM */
data cm;
    attrib ANTISEQ label = "Sequence Number" length = 8.
        PREFTERM label = "Drug Name Preferred Term" length = $200.
        ROUTE label = "Route" format = $ROUTE. length = $2.
        STARTDT label = "Start Date" format = DATE9. length = 8.
        STOPDT label = "Stop Date" format = DATE9. length = 8.;
    if _n_ = 0;
run;
/* Dummy DM */
data dm;
    attrib AGE label = "Age (years)" length = 8.
        AGECAT label = "Age in Years - Categorical Grouping" length = $8.
        RACE label = "Race" format = RACE. length = 8.
        RACE_ label = "Race-Decode" length = $200.
        SEX label = "Sex" format = SEX. length = 8.
        SUBJID label = "Subject Number" length = $3.;
    if _n_ = 0;
run;
```

```
/* Dummy EG */
data eg;
    attrib ECGDT label = "Date ECG Performed" length = 8.
        ECGDTD label = "ECG Day" length = $2.
        ECGDTM label = "ECG Month" length = $3.
        ECGDTY label = "ECG Year" length = $4.
        ECGRSLT_ label = "ECG Result-Decode" length = $200.;
    if _n_ = 0;
run;

/* Get the contents of the dummy data sets */
proc contents noprint data = work._all_
    out = alldata (keep = memname name type length label format);
run;

/* Create a structure for each data set */
data struc;
    length memname $32 memlabel $50. struc $50.;
    infile cards missover dlm = '$';
    input memname $ memlabel $ struc $;
    cards;
AE$Adverse Event$1 record per subject per adverse event
CM$Concomitant Antibiotics$1 record per subject per medication
DM$Demographics$1 record per subject
EG$Electrocardiogram$1 record per subject per visit
;
run;

/* Merge structure onto the contents */
data alldata;
    merge alldata struc;
    by memname;
run;

/* Create the database description section of the DDT */
proc sort nodupkey data = alldata out = Sect1 (keep = memname memlabel struc);
    by memname memlabel struc;
run;

data sect1;
    length memlabel1 location $200.;
    set sect1;
/* Create an internal hyperlink displaying the data set label with a translated label destination
    */
    memlabel1 = '{\field {\*\fldinst HYPERLINK \\l "' ||
        translate(trim(left(substr(memlabel, 1, 40))), "_"," ") ||
        '"}{\fldrslt {\cs15\cf2 ' || trim(left(memlabel)) || '}}}';

/* Create an external hyperlink to each data set transport file using the data set name */
    location = '{\field {\*\fldinst HYPERLINK "\\' ||
        trim(left(memname)) || ".xpt" || '"}{\fldrslt {\cs15\cf2 ' ||
        trim(left(memname)) || ".xpt" || '}}}';
run;

/* Create the first page of the DDT */
proc report data = sect1 nowd;
    column memname memlabel1 struc location;
    define memname /left style = [cellwidth = 10%] "Data Set";
    define memlabel1 /left style = [cellwidth = 35%] "Description";
    define Struc /left style = [cellwidth = 22%] "Structure";
    define location / left display style = [cellwidth = 32%] "Location";
    compute before _page_/style = [protectspecialchars = off
        pretext = "\sl-220\slmult0\b\ql "];
        line @1 "Data Sets for Example Study";
        line @1 " ";
        line @1 "Example Study:";
        line @1 "\plain\fs20 This is an example of some text that could be used at the top of the
    first section";
    endcomp;
run;
```

```
/* Create some dummy page numbers and a decode variable */
data alldata;
    set alldata;
    retain pg;
    by memname;
    if first.memname then pg = 1;
    else pg = pg+1;
    if type = '2' then type1 = "Character";
    else if type = '1' then type1 = "Number";
run;

/* Create the text for the comment field and the variable for the bookmark */
data allpgs;
    length com1 $100.;
    set alldata;
    name1 = upcase(name);
    form = trim(left(format));
    if mod(pg,2) = 0 then com1 = trim(left(label));
    else com1 = '{See {\field {\*\fldinst HYPERLINK "\\blankcrf.pdf#PAGE_' ||
        trim(left(pg)) || '"}{\fldrslt {\cs15\cf2 Page ' ||
        trim(left(pg)) || '}}}}';
    newname = translate(trim(left(substr(memlabel, 1, 40))), "_", " ");
run;
proc sort nodupkey data = allpgs out = formac (keep = memname memlabel newname);
    by memname;
run;

%macro rept(dtname = , mmlab = , bkmname = );
proc report data = allpgs (where = (memname = "&dtname")) nowd;
    column name1 label type1 form com1;
    define name1 /left style = [cellwidth = 10%] order "Variable";
    define label /left style = [cellwidth = 25%] "Label";
    define type1 /left style = [cellwidth = 9%] "Type";
    define form /left style = [cellwidth = 10%] "Code";
    define com1 /left style = [cellwidth = 45%] "Comments";
    compute before _page_ /style = [protectspecialchars = off asis = on
        pretext = "\sl-220\slmult0\b\ql "];
        line @1 "Example Study {\*\bkmkstart %trim(&bkmname)}&mmlab. {\*\bkmkend
    %trim(&bkmname)}";
    endcomp;
run;
%mend rept;

/* Create the output for each data set */
data _null_;
    set formac;
    call execute('%rept(dtname = ' || trim(left(memname)) ||
        ', mmlab = ' || trim(left(memlabel)) ||
        ', bkmname = ' || trim(left(newname)) || ');');
run;

/* Close the RTF output */
ods rtf close;

/* Remove nonbreaking spaces entered by COMPUTE blocks */
data temp;
    length line $1500.;
    infile "define.rtf" lrecl = 1500 length = lg;
    input line $varying1500. lg;
    if index(line, '\~') then line = tranwrd(trim(left(line)), '\~', ' ');
run;
data _null_;
    set temp;
    file "define.rtf" lrecl = 1500;
    new_lg = length(line);
    put line $varying1500. new_lg;
run;
```