PharmaSUG China 2017 - Paper 31

Strip Special & Non-Printable Characters in Your Data Set

Ying Liu, MSD, Beijing, China

ABSTRACT

Clinical data can be collected and analyzed by internal and external vendors. Various ways of importing data may bring out special and non-printable characters in clinical data, accompanied with potential issues in developing quality data sets and deliverables. There are major issues such as incorrect statistical analysis in tables, listings or figures, or minor issues like incorrect breaks in lines or words. In order to manage data effectively, this paper will encode special and non-printable characters by ASCII code in clinical data to understand the essential reasons. Moreover, corresponding solutions to handle these issues will be provided in this paper.

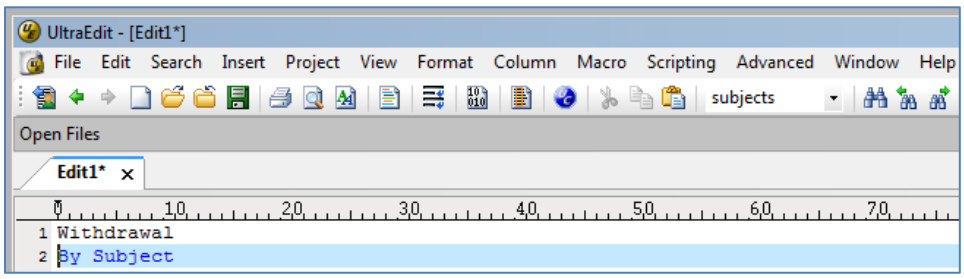
INTRODUCTION

The growth of clinical data is overwhelming not only because of the volume but also the diversity of data types from global market. Internal and external clinical data in the market are collected in various formats. Programmers have to derive the clinical data in format and make sure the result in the tables, listing or figures which can be traced back to the input data. Internal clinical data are in the standard format following the standard rules by most pharmaceutical companies in clinical trials. However, external clinical data in multiple formats are provided by the vendors to support for laboratory, biological samples and other analytics data. The risks are involved due to the extensive source of external clinical data. In the process of deriving variables and developing tables, programmers may encounter many special or non-printable characters which may result in incorrect breaks or huge deviations in statistical analysis. Considering the special and non-printable characters displayed as abnormal symbol or blank space, programmers may get confused when they are checking data between the display in table and variables in the SAS datasets. Nevertheless, these special and non-printable characters could be presentable in ASCII code by UltraEdit, which is a commercial text editor for Microsoft Windows, Linux and OS X. After tracking the special and non-printable characters, it would be easier to fix these issues through the methods mentioned in the paper. Moreover, striping the special and non-printable characters off in the process of deriving variables in the macro %ascii would be of high-efficiency and would avoid the vast major issues.

ASCII CHARACTER

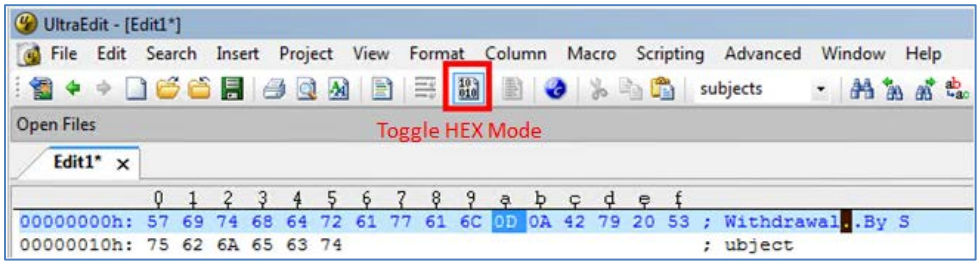
All of the internal and external clinical data are derived and generated into the tables, listings and figures in the computer by SAS. Encoded by the clinical data, these can be represented by using standardized numeric codes. The most widely acceptable code is called the American Standard Code for Information Interchange (ASCII). The ASCII code associates an integer value for each symbol in the character set, such as letters, digits, punctuation marks, special characters and control characters. ASCII code can be divided into two sets. One is Standard ASCII codes, which range from 0 to 127 in decimal or 00 to 7F in Hexadecimal. This set is mainly used for representing characters, such as characters “a” to “z”, numbers, and a part of non-printable characters with code from 0 to 31 (Decimal), which are used for controlling some peripheral devices. The other set is Extended ASCII codes, which range from 128 to 255 in decimal or 80 to FF in Hexadecimal. These meet the demand for more characters and symbols that are used for many languages. How to track the clinical data as characters in ASCII format? UltraEdit is convenient and helpful to review them.

First, open UltraEdit and paste the clinical data derived from SAS dataset or tables, such as Display 1.



Display 1. Clinical data from SAS dataset

Second, click the button “Toggle HEX Mode”. Then, the clinical data would be displayed as ASCII code, such as Display 2.



Display 2. Clinical data in ASCII format

In the Display 2, ASCII codes “0D” and “0A” mean carriage feed and line feed respectively. The full list of ASCII code is provided in Table 1 & 2 in the appendix. Tracking the characters is the first step to find out the targets, such as which one needs to be stripped off. After that, programmers can strip off the special or non-printable characters one by one or remove them in bulk.

NON-ASCII CHARACTER

Clinical trials across multiple regions of the world have become a common practice, with ultimate goal to bring good medicinal products to patients around the world. There are various languages in different regions, such as English in Western European, Chinese and Korean in Eastern Asia and so on. The written languages consist of thousands of characters, which can’t be covered by the limited ASCII code. In the paper, these uncovered characters are classified to “Non- ASCII” characters. According to the requirements from agency (U.S. Food and Drug Administration), the clinical data for submission are in the standard ASCII text formatted. Non-ASCII characters should be removed or converted. The methods to check non-ASCII characters and strip them off are also provided in the paper.

METHODS FOR ASCII CHARACTERS

As mentioned above, the special characters would track out by UltraEdit. SAS functions, such as “Translate”, “Transwrd” and “Compress”, can strip off these characters in a small range. Considering the vast of clinical data, it’s fussy to check all the data by UltraEdit and strip off the characters one by one. Based on the properties of characters in ASCII code or non-ASCII code, programmer can use Perl regular expressions with SAS functions to strip off special characters and non-printable characters in bulk.

METHOD I: TRANSLATE & TRANSWRD FUNCTION

TRANSLATE and TRANSWRD functions are used to replace specific characters in the strings one by one. Clarifying special or non-printable characters is the precondition to use these two functions. Here is an example to strip off the special character, carriage feed (ASCII code = “0D”), as an example.

```
newstring = translate(oldstring, "", "0D"x);
newstring = transwrd(oldstring, "0D"x, "");
```

METHOD II: COMPRESS FUNCTION

COMPRESS function can not only strip off the characters one by one in the strings, but also keep the characters by the K modifier in the third argument. Striping off the special and non-printable character, in words, means to keep the required characters. “w” is the most popular modifier for the printable characters.

1. Strip off the special characters, carriage feed (ASCII code = “0D”), in the following syntax:

```
newstring = compress(oldstring, "0D"x);
```

2. Keep printable characters, strip off not only carriage feed (ASCII code = “0D”) character but also other non-printable characters in the following syntax:

```
newstring = compress(oldstring, , "kw");
```

This powerful function with modifiers "k" and "w" can strip off the non-printable characters and vast majority of special characters. However, it's also restricted by the following conditions.

- Modifiers do not available before SAS version 9.
- Some special and printable characters cannot be stripped off by above syntax.
- ASCII codes from 128 to 159 are defined as printable or non-printable characters in different national language support. NLS (National Language Support) features in SAS system with "LOCALE = system options" instruct the SAS on how to represent the language currency by character sets. Latin-1 character set does not include ASCII code from 128 to 159. These ASCII codes are considered as non-printable characters in the Latin-1 character set. However, these ASCII codes are considered as printable characters in Windows Latin-1 character set. Considering the complex language support, COMPRESS function with "kw" modifiers commonly strips off special and non-printable characters, but could not cover all situations.

METHOD III: PRXCHANGE FUNCTIONS

Perl regular expression (PRX) functions enhance search-and-replace options in the string. Separate the ASCII code into two groups. One group is the lower codes from 00 to 19 (Decimal), which causes breaks, line feed or other abnormal display in the tables, listings and figures. The other is the higher codes after 7E (Decimal), which can vary from country to country. Character "µ" in ASCII code 230 in decimal may not be the same as "µ". PRXCHANGE function efficiently strips these two groups in bulk.

1. Strip lower codes (ASCII code from 00 to 19) in the following syntax:

```
newstring = prxchange('s/[\x00-\x19]//', -1, oldstring);
```

2. Strip higher codes (ASCII code after 7E) in the following syntax:

```
newstring = prxchange('s/[\x7F-\xFF]//', -1, oldstring);
```

Having checked the ASCII code in the Table 1 and Table 2, programmer can adjusting the ASCII code in the above syntax to strip series of ASCII codes.

METHODS FOR NON-ASCII CHARACTERS

Clinical data may be originated from another country or distributed across the global. Non-ASCII characters are likely to be included in with the national language. Although UltraEdit could track ASCII code in hexadecimal, it could not display the non-ASCII code accurately by "Toggle HEX Mode". NOTPRINT function is helpful to check the nonprintable characters in SAS and return the position of them, not only the characters in ASCII but also in non-ASCII. If there is no non-printable character in the string, it would return "0" by the following syntax:

```
position = noprint(oldstring);
```

Clarifying the non-printable characters is the first step to filter out the issue. The next step is to strip them off. Considering the imprecise non-ASCII code, it's better to keep all ASCII codes or adequate ASCII codes as required in the clinical data. As required by the agency (U.S. Food and Drug Administration), here are the steps to keep the printable ASCII characters.

Step 1, find out non-printable characters in ASCII characters. First, create the ASCII code in decimal. Then use HEXw. format to encode them as hexadecimal. Finally, NOTPRINT function would adjust the printable and nonprintable characters. The SAS code uses to track ASCII codes are as follows.

```
DATA notprint;
   do dec = 0 to 255;
   byte = byte(dec);
   hex = put(dec,hex2.);
   notprint = notprint(byte);
   output;
   end;
RUN;
```

Checked by NOTPRINT function, the non-printable ASCII characters are from 00 to 31, 129, 141, 143, 144, 157 and 173. The output is shown below:

ASCII values, and characters (Decimal, Hexadecimal and Actual character)

Obs	dec	byte	hex	notprint
130	129		81	1
142	141		8D	1
144	143		8F	1
145	144		90	1
158	157		9D	1
174	173		AD	1

Output 1. Output from tracking ASCII codes Statement

Step 2, strip off all special and non-printable characters (ASCII or Non-ASCII). First, combine all printable characters in the variable &ascii_chars by macro %ascii as below.

```
options minoperator mindelimiter=',';
%macro ascii();
    %local i asciichars;
    /* Adjust here to include any additional chars */
    %do i=32 %to 255;
    /* Adjust here to exclude any additional chars */
    %if &i in (129, 141, 143, 144, 157, 173) %then %do;
      %let asciichars=&asciichars%qsysfunc(byte(49));
    %end;
    %else %do;
      %let asciichars=&asciichars%qsysfunc(byte(&i));
    %end;
    %end;
    %str(&asciichars)
%mend;

/* Store reserved characters in macro variable */
%let ascii_chars=%ascii();
%put &=ascii_chars;
```

Second, COMPRESS function with “k” modifier can keep the printable characters from the variable &ascii_chars in the string and remove the rest special or non-printable characters in the macro language and data steps. Here is the macro as examples.

```
/* Example 1: Skip special and non-printable characters within macro language */
%put %sysfunc(compress(old - string,&ascii_chars,k ));

/* Example 2: Skip special and non-printable characters within data step */
data _null_;
    oldstring="old ›¡¢£¤¥œ □ª«¬-®¯°±²³´µ¶· •——˜™šžŸ ¦§¨© string";
    asciichars=symget('ascii_chars');
    newstring=compress(oldstring,asciichars,'k');
    put newstring=;
run;
```

Compared with checking and striping off the characters in the variable one by one, it’s of high-efficiency to strip off the special and non-printable characters in batch of variables by the macro %ascii. Moreover, Programmers can adjust the value of macro variable “i” to include or exclude any characters depending on the requirements in clinical trials.

CONCLUSION

With globalization of drug development, mulit-regional clinical trial (MTCT) has widely been conducted in clinical trials by global pharmaceutical companies. The approach of collection is one aspect result in the various formats of clinical data. In the other aspect, native language of the investigator in MTCT needs translation to enter the clinical database.

It would also affect the format of clinical data in diversity. National languages in computer code can be encoded by different character sets. For example, the Latin character set is used by English and most European languages, though the Greek character set is used only by the Greek language. With the CDISC submission to U.S. Food and Drug Administration (FDA), the computer code is most usefully formatted as standard ASCII text files. FDA published a report (*Test Report for DS-XML Pilot 2015*) in their pilot of the use of Dataset-XML as an alternative to SAS-XPT (SAS Transport5) for SDTM, SEND and ADAM submissions. There was a test batch with error "Some code points did not transcode" at the data conversion. It was due to non ASCII characters in the datasets and the Define.xml. Whether a Latin character set or Greek character set is used to collect clinical data is not important, it is best to strip non-ASCII characters and transcoding character without special characters in the developing, especially in cleaning up source data, to avoid the same error in CDISC submission.

If non-ASCII characters are never entered into the clinical data at the beginning, it could preclude the error at source. In the process of transcoding, the method mentioned in the paper can monitor the data to avoid the risk of creating special or non-printable characters. Not only can the methods in this paper be used to clarify the characters in ASCII code, but also evaluate and strip off special and non-printable characters. However, the methods of striping special and non-printable characters are not limited in this paper. Ultimately, solutions must be made based on the issue at hand. Understanding and handling the character issues is crucial to keep high quality of CDISC submission.

REFERENCES

Jasmin Johnson. (2014). "Third party laboratory data management: Perspective with respect to clinical data management". Perspectives in Clinical Research. 5(1): 41–44.

Premnath Shenoy. (2016). "Multi-regional clinical trials and global drug development". Perspectives in Clinical Research. 7(2): 62–67.

Michael Stackhouse. (2016). "UTF What? A Guide to Using UTF-8 Encoded Data in a SDTM Submission". PharmaSUG. Paper BB16.

Mark Tabladillo. (2012). "Regular Expressions in SAS® Enterprise Guide®". SAS Global Forum. Paper 299.

"UltraEdit". Available at https://en.wikipedia.org/wiki/UltraEdit

"4.1.1 The ASCII Character Set". Available at http://ee.hawaii.edu/~tep/EE160/Book/chap4/subsection2.1.1.1.html

"Clinical Data for Premarket Submissions". Available at https://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/HowtoMarketyourDevice/PremarketSubmissions/ucm136377.htm

Allan Bowe. "Stripping Non-ASCII Characters within Macro", Available at http://www.rawsas.com/2016/09/stripping-non-ascii-characters-within.html

RECOMMENDED READING

- SAS® 9.4 National Language Support (NLS)
- Test Report for DS-XML Pilot, April 8, 2015

CONTACT INFORMATION

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

Name: Ying Liu
Enterprise: MSD R&D (China) Co., Ltd.
Address: Rongda Road, Wangjing R&D Base, Zhongguancun Electronic Zone West Zone, Chaoyang District
City, State ZIP: Beijing 100012, China
Work Phone: +86 10 5860 9397
Fax: +86 10 5860 9397
E-mail: ying.liu9@merck.com

APPENDIX

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

Table 1. Standard ASCII Chart / ASCII Table - Hex to Decimal Code Conversion

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	▒	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▓	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	Ö	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	▒	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	Pts	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

Table 2. Extended ASCII Chart / ASCII Table - Hex to Decimal Code Conversion