

SM08

Unlocking the Enigma: Exploring SAS and R Strategies for Handling Non-Printable and Special Characters

PHUSE EU Connect 2024

November 10 – 13

Strasbourg Convention Centre



Agenda

- Introduction
- Non-Printable and Special Characters
- How do these characters step out of line?
- Handle these characters using:
 - SAS Macro
 - R Function
- Benefits and Conclusion



Introduction

- Invisible characters can significantly impact data integrity and report accuracy.
- Specific issues with hidden characters have been encountered in our work.
- Serious issues in reports, such as incorrect counts or statistics.
- Problems like incorrect line breaks or page breaks.
- Appearance of strange symbols in the reports.
- Importance of identifying and handling non-printable and special characters using SAS and R

What are Non-printable and special character?

- The first 32 ASCII codes (0-31) are non-printable control characters.

Examples:

- Carriage returns (CR)
- Line feeds (LF)
- Tabs that create indentation etc.

- The Non-ASCII (beyond 127) special Characters.
- Example: ‡, Š, etc

Code (dec)	Code (hex)	Value
128	80	€
129	81	
130	82	,
131	83	<i>f</i>
132	84	"
133	85	...
134	86	†
135	87	‡



How do these characters step out of line?

- Non-printable and special characters can create:
 - Incorrect counts/statistics
 - Improper line breaks
 - Page breaks
 - Appearance of strange symbols in reports.
- The following output gives an example of Line feed character.

Catalyst CR
Protocol: XX

Confidential

Page 1 of 2

Characteristic	Location	Trt A	Trt B	Total
Any TEAE [a]	Ocular-Any Treated Eye	2 (16.7)	0	2 (11.1)
Eye disorders	Ocular-Any Treated Eye	2 (16.7)	0	2 (11.1)
Corneal epithelium defect	Ocular-Any Treated Eye	1 (8.3)	0	1 (5.6)
Corneal opacity	Ocular-Any Treated Eye	1 (8.3)	0	1 (5.6)
Keratic precipitates	Ocular-Any Treated Eye	1 (8.3)	0	1 (5.6)
Visual impairment	Ocular-Any Treated Eye	1 (8.3)	0	1 (5.6)
Any TEAE [a]	Ocular-All Eyes	4 (33.3)	1 (16.7)	5 (27.8)
Eye disorders	Ocular-All Eyes	4 (33.3)	1 (16.7)	5 (27.8)
Visual impairment	Ocular-All Eyes	1 (8.3)	1 (16.7)	2 (11.1)
Corneal epithelium defect	Ocular-All Eyes	1 (8.3)	0	1 (5.6)
Corneal epithelium defect	Ocular-All Eyes	1 (8.3)	0	1 (5.6)
Corneal opacity	Ocular-All Eyes	1 (8.3)	0	1 (5.6)



Handling in SAS and R

- Developed a SAS macro and an R function to handle non-printable and special characters.
- Ensuring consistency and ease of handling across all outputs.
- The SAS macro utilizes the BYTE function to identify non-printable and special characters.
- Characters can be removed or replaced with a specified character, defined in an input Excel file that users can update as needed.
- A single run updates the entire dataset, outputs it to the specified location, and generates a detailed report on the changes made.

Handling in SAS

- The excel file plays key role in replacing the character.
- Either it can be user defined, or the macro creates its own.
- Below SAS code shows how the macro creates the file.

```
*Identifying non-printable/special characters using SAS when the excel is not available;
data excl_npsc;
  length schar rchar $7.;
  do i=130 to 140, 142, 145 to 156, 158, 159, 161 to 255;
    schar=byte(i);
    rchar=" ";
    output;
  end;
  schar="'09'x";
  rchar="nospace";
  output;
  schar="'0D0A'x";
  rchar="nospace";
  output;
  schar="'0D'x";
  rchar="nospace";
  output;
  schar="'0A'x";
  rchar="nospace";
  output;
  rchar="nospace";
  keep schar rchar;
run;
```

	A	B	C
1	RowNumber	schar	rchar
2		1 ,	
3		2 f	
4		3 „	
5		4 ...	
6		5 †	
7		6 ‡	
8		7 ^	
9		8 ‰	
10		9 Š	
11		10 ‹	
12		11 Œ	

Excel file with special character(schar) and replacement character (rchar)

Handling in SAS

- The SAS functions 'TRANWRD', TRANSTRN' and 'COMPRESS' are used for removing or replacing characters.
- Macro creates a report for detailed information regarding the update.

Non printable & Special Characters Report

05:24 Friday, October 11, 2024

Dataset Name	Variable Name	Non Printable & Special Character	Replaced Character	Excel Row Number
CM	CMTRT	@	(replaced with space)	40
DV	DVTERM	'	'	14
T14_3_1_2_V	CHARACTERISTIC	'0A'x	(replaced without space)	125
TS	TSVAL	'	'	14
TS	TSVAL1	½	(replaced with space)	55
TS	test	"	"	15

Handling in R

- Handling non-printable and special characters in R is relatively straightforward and can be approached in various ways.
- The user-defined R function `replace_special_characters` utilizes `lapply()` and `sapply()` to apply a replacement function to each column and check for the presence of special characters.

```
replace_special_characters <- function(dataset2, dataset1, dfName) {  
  
  replacements_made <- c()  
  
  # For each row in df1  
  for (i in seq_len(nrow(dataset1))) {  
    special_char <- dataset1$special_char[i]  
    replacement_char <- dataset1$replacement_char[i]  
  
    dataset2 <- as.data.frame(lapply(dataset2, function(column) {  
      sapply(column, function(x) {  
        if (!is.na(x) && grepl(special_char, x)) {  
          # Print  
  
          replacements_made <- c(replacements_made,  
                                paste0("Replaced '", special_char, "' with '", replacement_char,  
                                      "' in value: ", x))  
        }  
      })  
    })  
  }  
}
```

Handling in R

- If any special characters are found, they are replaced with the specified replacement character using the `gsub()` function.
- Another function, `convert_sas_datasets`, reads all the SAS datasets from the input folder, processes them, and returns the output to the target directory.

```
convert_sas_datasets <- function(source_dir, target_dir) {  
  # Get a list of all files in the source directory  
  all_files <- list.files(source_dir, full.names = TRUE)  
  
  # Filter the list to include only SAS datasets (.sas7bdat files)  
  sas_files <- all_files[grepl("\\.sas7bdat$", all_files)]  
  
  # Create the target directory if it doesn't exist  
  if (!dir.exists(target_dir)) {  
    dir.create(target_dir)  
  }  
  
  # Read and process each SAS dataset  
  result_dataframes <- list()  
  for (i in seq_along(sas_files)) {  
    file_path <- sas_files[i]
```



Benefits and Conclusion

- Standardizing the handling of these characters ensures consistency.
- Preserves data integrity.
- Streamlines data handling processes, saving time and resources.
- Meets regulatory standards.
- Ensures clear communication among all stakeholders.
- Produces more accurate and understandable results.
- Thereby advancing medical research and enhancing patient care.

Thank You

hardik.sheth@catalystcr.com

eldho.alias@catalystcr.com

vishal.nair@catalystcr.com