Learning Guide: Removing Special Characters from Strings in SAS

Authored by

Mohammed looti

October 27, 2025

In the world of data analysis, ensuring the integrity and usability of your datasets is paramount. Unwanted elements, particularly special characters embedded within text fields--or strings--can severely hinder processing, matching, and reporting within the SAS environment. Fortunately, SAS provides highly efficient tools for rigorous data cleaning.

The most straightforward and robust method for systematically removing these extraneous symbols from a string is by utilizing the powerful `COMPRESS function` paired with specific modifiers, such as the widely used 'kas' combination. This approach allows developers to define what characters should be **kept**, implicitly removing everything else, thereby simplifying complex string manipulation tasks.

Understanding the SAS COMPRESS Function Syntax

The `COMPRESS function` is specifically designed to return a character string with specified characters removed. While it removes standard spaces by default when used minimally, its true strength emerges when leveraging modifiers to selectively remove or keep entire classes of characters, making it the ideal choice for eliminating unwanted punctuation or symbols from large datasets.

For the specific task of removing all non-alphanumeric special characters while preserving letters and spaces, we leverage the 'kas' modifier combination. This approach simplifies complex character removal into a single, highly readable operation that is executed efficiently within the SAS Data Step environment. The fundamental syntax structure required for this operation is shown below, demonstrating how a new variable is created based on the transformation of an existing one:

```
data new_data;
set original_data;
remove_specials = compress(some_string, , 'kas');
run;
```

In this syntax, `new_data` represents the output dataset, `original_data` is the source, and `some_string` is the variable containing the characters you wish to clean. It is important to note the placement of the second argument, which is intentionally left blank. This is because the 'kas' modifier tells SAS to use predefined character sets (alphabetic and space characters) as the basis for retention, rather than requiring the user to supply a custom list of characters to remove.

The Critical Need for Data Cleaning in SAS

In real-world data environments, data seldom arrives in a perfectly clean state. Issues such as

inconsistent data entry, corrupted imports, or limitations in source systems often introduce noise, frequently manifesting as embedded special characters. These characters--which might include symbols like `&`, `@`, `#`, `!`, or various brackets--can cause severe analytical headaches. For instance, attempting to merge datasets, perform exact matches, or apply complex statistical procedures often fails or produces incorrect results when text fields are polluted with unexpected symbols.

Effective data cleaning is, therefore, a mandatory preprocessing step, particularly when preparing data for statistical modeling or reporting within the SAS system. Ensuring that primary key fields or categorical variables are free of noise guarantees reliable data integrity and consistency across analytical stages.

While other SAS functions, such as `TRANSLATE` or `REPLACE`, can address specific character substitutions, the `COMPRESS function` stands out for its efficiency in handling large, undefined sets of unwanted characters. Instead of having to list every possible symbol for removal, `COMPRESS` allows the user to define what should be **kept**, simplifying the logic dramatically. This "keep list" approach, facilitated by the 'k' modifier, ensures that only valid characters remain, streamlining the data validation process considerably.

Practical Example: Defining the Dirty Dataset

To demonstrate the practical utility and necessity of the `COMPRESS` function, consider a hypothetical scenario involving employee sales data. We have captured employee names, but due to errors in data input or transfer, the names contain various extraneous special characters. Before we can accurately sort, group, or match these employee records, these symbols must be purged. The following SAS code block illustrates the creation and display of this initial, "dirty" dataset:

```
/*create dataset*/
data data1;
input name $ sales;
datalines;
Bob&%^ 45
M&$#@ike 50
Randy)) 39
Chad!? 14
Dan** 29
R 44
;
run;

/*view dataset*/
proc print data=data1;
```

Upon reviewing the output generated by the `PROC PRINT` statement, it becomes immediately apparent that the values stored in the **name** column are inconsistent. They contain numerous symbols like `&`, `%`, `^`, `$`, and various punctuation marks mixed in with the actual names. This visual verification confirms the immediate need for data transformation using robust SAS string manipulation tools to prepare the data for downstream processes.

Obs	name	sales
1	Bob&%^	45
2	M&$#@ike	50
3	Randy))	39
4	Chad!?	14
5	Dan**	29
6	R[on]	44

Implementing the 'kas' Modifier for Clean Output

To efficiently cleanse the **name** variable, we will apply the `COMPRESS function` within a new Data Step. We create a new variable, `new_name`, which will store the cleaned version of the original `name` string. The essential component of this transformation is the inclusion of the 'kas' modifier, which instructs SAS to only keep specific types of characters, implicitly removing everything else that does not fall into those specified categories.

The 'kas' modifier combination is highly efficient because it leverages predefined character sets for retention rather than removal. The individual components of the modifier string function as follows:

k specifies the 'Keep' action: This is the fundamental modifier that dictates that only characters belonging to the specified classes or listed in the character argument should be retained.
a specifies the 'Alphabetic' class: This ensures that all uppercase and lowercase letters (A-Z, a-z) are preserved in the output string.
s specifies the 'Space' class: This ensures that standard space characters, often important for readability in names, are also retained.

By using `compress(name, , 'kas')`, we are effectively telling SAS: "From the variable `name`, keep only the alphabetic characters and spaces, and remove every other character encountered." This operation systematically strips away all numerical digits, punctuation, and system symbols, guaranteeing a clean output ready for subsequent analysis. The resulting SAS code block that executes this cleaning process is structured as follows:

```
/*create second dataset with special characters removed from names*/
data data2;
set data1;
new_name=compress(name, , 'kas');
run;

/*view dataset*/
proc print data=data2;
```

The resultant dataset, `data2`, confirms the successful application of the function. The new **new_name** column now contains standardized, clean values. For example, 'Bob&%^' becomes 'Bob', and 'M&$#@ike' becomes 'Mike'. This demonstrates that the `COMPRESS` function successfully isolated and eliminated all extraneous special characters, leaving only the intended alphabetic content.

Obs	name	sales	new_name
1	Bob&%^	45	Bob
2	M&$#@ike	50	Mike
3	Randy))	39	Randy
4	Chad!?	14	Chad
5	Dan**	29	Dan
6	R[on]	44	Ron

Expanding the Power of COMPRESS Modifiers

While 'kas' is highly effective for basic name cleaning, the `COMPRESS function` offers a comprehensive suite of modifiers that allow users to manage almost any character removal scenario. Understanding these additional options is crucial for performing complex data cleaning tasks efficiently within the SAS environment. These modifiers can be combined in any required sequence within the quotes argument to define highly specific character retention or removal rules.

The following list highlights some of the most commonly used modifiers, which extend the capability of the `COMPRESS` function beyond simple alphabetical cleaning:

a (Alphabetic): Includes all English alphabetic characters (A-Z, a-z).
d (Digit): Includes all numerical digits (0-9).
n (Alphanumeric): Includes all letters and digits (this is functionally equivalent to combining 'ad').
p (Punctuation): Includes standard punctuation marks (e.g., periods, commas, semicolons,

exclamation points).
s (Space): Includes standard space characters.
c (Control): Includes control characters (e.g., tab, carriage returns, form feeds).
w (Whitespace): Includes horizontal and vertical tab, carriage return, and line feed characters.

For example, if your analytical goal required retaining only alphanumeric characters (letters and numbers) and removing all other symbols, including spaces, you would use the 'kn' modifier: `compress(variable, , 'kn')`. Conversely, if the requirement was to remove only punctuation and digits from a text field, but keep everything else (letters, spaces, and other symbols), you would use 'dp' **without** the 'k' modifier. When 'k' is omitted, the characters defined by the modifiers are removed, not kept. Mastering these versatile modifier combinations allows SAS programmers to handle virtually any text transformation requirement efficiently and robustly.

Additional Resources for SAS String Manipulation

The inherent versatility and efficiency of the `COMPRESS function` make it an indispensable tool in the SAS toolkit for cleaning character data. Its modifier system provides fine-grained control over which characters are retained or discarded, surpassing the capabilities of many simpler string functions when dealing with unknown or varied special characters.

For advanced scenarios and a comprehensive understanding of all available options, including multibyte character handling and locale-specific modifiers, it is highly recommended to consult the official documentation provided by SAS. A complete and authoritative list of modifiers for the `COMPRESS` function, along with detailed usage notes, can be found on this official SAS documentation page.