Learning SAS: Extracting Numerical Data from Strings

Authored by

Mohammed looti

October 27, 2025

In the realm of data analysis, particularly when processing raw or poorly structured data, analysts frequently encounter the challenge of extracting specific data types from alphanumeric variables. Isolating numerical values embedded within a character **string** is a fundamental requirement for cleaning and preparing data for statistical modeling. **SAS**, recognized globally as a powerful statistical software suite, offers highly efficient tools for these data manipulation tasks. Among these tools, the **COMPRESS function** stands out as a straightforward and effective method for cleanly isolating digits from mixed alphanumeric data.

This comprehensive tutorial focuses on mastering the technique of extracting only numeric characters from a **string** variable using the **COMPRESS function** in **SAS**, specifically utilizing the powerful 'A' modifier. Furthermore, we will demonstrate the function's versatility by exploring how to extract only alphabetic characters using a different modifier. Developing proficiency in this essential **SAS** utility is indispensable for any data professional looking to streamline data cleaning workflows and ensure their **dataset** is ready for rigorous statistical analysis.

Deep Dive into the COMPRESS Function and Its Modifiers

The primary purpose of the **COMPRESS function** in **SAS** is to remove specific characters or categories of characters from an input **string**. While it can be used simply to remove explicitly defined characters, its true potential is unlocked through its optional third argument: the **modifiers**. These **modifiers** provide shorthand flags that specify entire classes of characters (e.g., all letters, all digits, all punctuation) to be either removed or kept, enabling far more complex and efficient character manipulation.

To successfully extract only the numerical components, we employ the 'A' modifier. The 'A' **modifier** stands for "alphabetic characters" and instructs the **COMPRESS function** to eliminate every letter (A-Z and a-z) present in the source **string**. By systematically removing all alphabetic characters, the remaining output consists exclusively of non-alphabetic characters, which usually include the desired numbers, as well as any existing symbols or spaces. When this process is executed using an empty second argument (meaning no user-defined characters are targeted for removal), the 'A' modifier acts as a powerful filter, leaving only the numerical components we need.

The fundamental **syntax** required to implement this extraction logic within a **SAS DATA step**, which is the standard environment for variable creation and transformation, is clearly defined below:

```
data new_data;
set original_data;
numbers_only = compress(some_string, '', 'A');
run;
```

In this code block, `new_data` represents the resultant **dataset**, `original_data` serves as the source **dataset** containing the raw variables, and `some_string` is the specific variable from which the numerical data must be extracted. The newly generated variable, `numbers_only`, will then hold the resulting numeric-only **string** value.

Demonstration: Isolating Numeric Identifiers in Course Data

To solidify the understanding of the **COMPRESS function**, let us work through a practical and highly relevant scenario. Consider a situation where a **SAS dataset** stores course information where the subject name and the corresponding numerical course identifier are concatenated into a single variable. The primary objective is to separate these numerical identifiers cleanly from the textual subject names, which is a necessary preparatory step for reporting or joining data tables.

We begin by constructing a sample **dataset** named `original_data`. This step simulates the real-world occurrence of mixed alphanumeric data, providing a tangible starting point for our manipulation exercise. The following **SAS DATA step** is used to define and populate this initial **dataset** with representative course entries:

```
/*create dataset*/
data original_data;
input course $12.;
datalines;
Stats101
Economics203
Business201
Botany411
Calculus101
English201
Chemistry402
Physics102
;
run;

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

Executing this code generates the `original_data` **dataset**. The output, typically viewed using **PROC PRINT**, confirms the mixed format of the course names, establishing the exact state of the data before the extraction process begins.

Obs	course
1	Stats101
2	Economics203
3	Business201
4	Botany411
5	Calculus101
6	English201
7	Chemistry402
8	Physics102

Implementing Numeric Extraction using the 'A' Modifier

With the sample data successfully prepared, the next step involves applying the **COMPRESS function** to isolate the numeric digits within the `course` variable. We will introduce a new variable, logically named `course_number_only`, to store the resulting extracted numbers. This variable creation and transformation logic is encapsulated within a subsequent **SAS DATA step**.

The core of this operation lies in the precise application of the 'A' modifier. As shown in the **SAS** code below, specifying `'', 'A'` within the function call instructs the software to remove all alphabetic characters from the `course` variable. This targeted removal effectively isolates and retains only the numerical digits, fulfilling the requirement for numeric extraction.

```
/*extract numbers from course column*/
data new_data;
set original_data;
course_number_only = compress(course, '', 'A');
run;

/*view results*/
proc print data=new_data;
```

Upon review of the resulting `new_data` **dataset**, the efficacy of this method becomes apparent. The newly established `course_number_only` column contains sequences such as "101" and "203", successfully demonstrating a clean, data-type specific separation from the original mixed **string** data.

Obs	course	course_number_only
1	Stats101	101
2	Economics203	203
3	Business201	201
4	Botany411	411
5	Calculus101	101
6	English201	201
7	Chemistry402	402
8	Physics102	102

Complementary Technique: Extracting Alphabetic Characters with the 'D' Modifier

While extracting numbers is a frequent requirement, data manipulation often demands the opposite: isolating only the textual, alphabetic components of an alphanumeric field. Fortunately, the **COMPRESS function** is equally adept at this task, requiring only a change in the **modifier** used.

To extract solely the characters (letters), we utilize the 'd' modifier. This lowercase 'd' signifies "digits" and instructs the function to remove all numeric digits (0 through 9) from the input **string**. By eliminating all numerical values, the remaining data will consist exclusively of the alphabetic characters, alongside any other non-digit characters that may be present, such as spaces or special symbols.

The following **SAS DATA step** demonstrates how to modify the previous logic to isolate the textual subject names from the course identifiers:

```
/*extract characters from course column*/
data new_data;
set original_data;
course_characters_only = compress(course, '', 'd');
run;

/*view results*/
proc print data=new_data;
```

Analyzing the output generated by this code reveals a new column, `course_characters_only`,

which now holds only the alphabetic components--the subject names--successfully fulfilling the requirement to extract textual data by removing the numerical identifiers.

Obs	course	course_characters_only
1	Stats101	Stats
2	Economics203	Economics
3	Business201	Business
4	Botany411	Botany
5	Calculus101	Calculus
6	English201	English
7	Chemistry402	Chemistry
8	Physics102	Physics

Conclusion and Further Exploration of COMPRESS Modifiers

The **COMPRESS function** is a highly flexible workhorse in the **SAS** programming environment, primarily due to its rich set of **modifiers**. As demonstrated, simple changes to the modifier argument (from 'A' to 'd') allow users to flip between extracting numbers and extracting text with minimal code alteration. Beyond 'A' (alphabetic) and 'd' (digits), SAS provides modifiers for targeting and removing punctuation, excess whitespace, control characters, and many other specific character groups.

Leveraging these advanced options can dramatically improve your ability to execute precise data cleaning and transformation tasks, ensuring your data is standardized and ready for any subsequent modeling or reporting task. For analysts seeking to exploit the full potential of this function, consulting the authoritative source is recommended.

For a comprehensive understanding and a complete inventory of all available **modifiers** for the **COMPRESS function**, we strongly advise referring to the official documentation. This resource offers in-depth explanations and varied examples for each modifier, allowing you to apply the function with unparalleled precision and confidence in any data scenario:

Note: You can find a complete list of **modifiers** for the **COMPRESS function** on this **official SAS documentation page**.

To further enhance your **SAS** programming skills and explore other common data manipulation techniques, consider reviewing the following tutorials, which address typical challenges encountered within the **SAS** environment:

How to Clean Data in SAS
Working with Dates and Times in SAS
Performing Conditional Logic with IF-THEN/ELSE in SAS

These resources provide essential knowledge for tackling diverse data-related challenges efficiently and effectively.