How to use the TODAY Function in SAS (With Examples)

Authored by

stats writer

November 19, 2025

The **TODAY function** in **SAS** is a powerful utility designed to retrieve and return the current system date. Understanding how SAS handles dates is critical for effective data manipulation; the TODAY function is fundamental to this process. It generates a **numeric SAS date value**, which represents the total number of days that have elapsed since the reference date of January 1, 1960. This base date (January 1, 1960) is the cornerstone of SAS date calculations.

The utility of the **TODAY function** extends far beyond simply displaying the date. It is an indispensable tool for time-sensitive analytical tasks. Data analysts frequently employ it to compute the span between two distinct dates, to extract specific date components (like the current day or year), or to dynamically generate new variables in a dataset that are contingent upon the execution date of the SAS program. Mastery of this function is a prerequisite for sophisticated date handling in SAS programming environments.

This comprehensive guide will demonstrate practical, step-by-step examples showcasing how to correctly implement the **TODAY function**. We will explore scenarios ranging from generating the raw numeric date value to applying standard SAS formats like DDMMYY10. and DATE9. to ensure human-readable output, thereby making your code and results clear and maintainable.

Understanding the Numeric SAS Date Value

Before diving into practical applications, it is essential to grasp how SAS internally manages dates. SAS stores all date values as simple numeric data. Specifically, a date is recorded as an integer representing the count of days since January 1, 1960. For instance, if the current date is determined to be 23,135 days after January 1, 1960, SAS stores the value 23135. This numeric representation is highly efficient for mathematical operations, such as calculating age or finding the difference between event dates.

The primary purpose of the **TODAY function** is precisely to retrieve this internal numeric representation of the current operating system date. When you invoke `TODAY()` within a DATA step, SAS fetches the current date and returns the corresponding **SAS date value**. While this numeric output is mathematically precise, it is usually meaningless to a non-programmer, necessitating the application of specific SAS formats for output and reporting.

The following examples illustrate how to utilize the **TODAY function** effectively, transitioning from the raw numeric output to clearly formatted date variables suitable for presentation.

Example 1: Retrieving the Current Date as a Numeric Value (Unformatted Output)

By default, when the **TODAY function** is used without applying any explicit format, it returns the

current date as the underlying numeric **SAS date value**. This value represents the total count of days since January 1, 1960. This raw output is crucial if you intend to perform immediate date arithmetic within the DATA step before any formatting occurs.

The following code snippet demonstrates the fundamental usage of the function. We create a simple dataset called `my_data` and assign the result of `today()` to a new variable named `today_date`. We then use **PROC PRINT** to display the resulting dataset, revealing the raw numeric value.

```
/*create dataset that contains current date*/
data my_data;
today_date=today();
run;

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

Obs	today_date
1	23135

As shown in the output above, the **TODAY function** generated the numeric value **23135**. This output signifies the numeric date value corresponding to the execution date. If this code were run on May 5, 2023 (the date associated with this specific output), the value 23,135 confirms that exactly 23,135 days had elapsed between January 1, 1960, and May 5, 2023. While useful for internal calculations, this numeric format is challenging for human interpretation, thus necessitating the use of SAS date formats.

The Necessity of Formatting SAS Dates

The raw numeric output from functions like `TODAY()` is optimized for machine processing but lacks readability. To present dates in a standard, easily understandable format (e.g., MM/DD/YYYY or 05MAY2023), we must apply a SAS format. SAS provides a comprehensive library of formats, allowing users to customize the display of date variables without altering the underlying numeric value stored in the dataset.

The primary mechanism for applying a format is the `FORMAT` statement within the DATA step or a relevant **PROC PRINT** statement. Selecting the appropriate format depends heavily on regional requirements and the desired level of detail. The following examples demonstrate two of the most

commonly used formats for displaying the current date.

Example 2: Applying the DDMMYY10. Format for Clarity

The `DDMMYY10.` format is frequently utilized in regions where the day precedes the month (e.g., European and many Asian conventions). The `10.` suffix specifies the width of the display, ensuring enough space for the day, month, year, and separators (typically slashes or dashes).

The code below modifies Example 1 by incorporating the `FORMAT` statement, instructing SAS to display the numeric value stored in `today_date` using the `DDMMYY10.` format. This immediately transforms the raw number into a structured date string.

```
/*create dataset that contains current date*/
data my_data;
today_date=today();
format today_date ddmmyy10.;
put today_date;
run;

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

Obs	today_date
1	05/05/2023

In this output, the **TODAY function** still returns the numeric value (23135), but the `DDMMYY10.` format overwrites the display, rendering it as the human-readable **05/05/2023**. Note that the underlying data in the SAS dataset remains the integer 23135; only the visual representation has been modified for reporting purposes.

Example 3: Utilizing the DATE9. Format for Standard Display

Another popular format for date representation in SAS is `DATE9.`. This format uses a fixed width of nine characters and typically displays the date using the standard format DDMMMYY (e.g., 05MAY23 or 05MAY2023, depending on width and century settings). This is often preferred in formal reports because the three-letter abbreviation of the month removes any ambiguity that might arise from strictly numerical formats like MM/DD/YY.

The following program demonstrates how to use the **TODAY function** in conjunction with the

`DATE9.` format to achieve this standardized display.

```
/*create dataset that contains current date*/
data my_data;
today_date=today();
format today_date date9.;
put today_date;
run;

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

Obs	today_date
1	05MAY2023

The resulting output clearly displays the date as **05MAY2023**, utilizing the standard `DATE9.` structure. This format provides an excellent balance between conciseness and clarity, making it a professional choice for many SAS projects.

Exploring Other Essential SAS Date Formats

The examples above only scratched the surface of date formatting options available in SAS. Since the **SAS date value** is highly flexible, it can be displayed in virtually any common date format required globally. Choosing the correct format is essential for meeting data governance standards and ensuring international compatibility.

Other widely used SAS date formats include:

`MMDDYY8.`: Displays the date as MM/DD/YY (e.g., 05/05/23).
`WEEKDATE.`: Displays the day of the week and full date (e.g., Friday, May 5, 2023).
`YYMMDD10.`: Displays the date in ISO standard format (YYYY-MM-DD), which is excellent for data exchange and sorting.

For complex applications, such as handling international time zones or specific regulatory reporting requirements, you may need to consult the complete SAS documentation. Refer to the official SAS documentation for a complete and comprehensive list of all available date and time formats that can be used with the **SAS formats** library.

Summary of Key Takeaways

The **TODAY function** is a straightforward yet critical tool for any SAS programmer needing to reference the current date. It operates by returning a numeric value counting the days since January 1, 1960. While this raw output is necessary for computational accuracy, applying a format is crucial for human readability and reporting. We demonstrated how to use the `FORMAT` statement with popular formats like `DDMMYY10.` and `DATE9.` to transform the numeric output into clear, structured date strings.

Mastering date handling in SAS involves not just knowing the utility functions, but also understanding the underlying numeric structure and the vast array of formatting options available. Incorporate `TODAY()` into your DATA steps to ensure your analyses are always reflective of the most current data execution context.