How to Easily Concatenate Datasets in SAS

Authored by

stats writer

December 1, 2025

The process of combining data files is a fundamental task in data management and analysis. In SAS, achieving this combination is straightforward, particularly when stacking observations one after the other--a process known as Concatenation.

Unlike merging, which typically combines variables horizontally based on a key identifier, concatenation stacks entire records vertically, resulting in a single, unified data structure. This is essential when dealing with input datasets collected over different time periods or from different branches that share an identical variable structure. Mastering this technique ensures that data preparation is efficient and accurate for subsequent statistical procedures.

The core mechanism for this operation is the SET statement, which is executed exclusively within the DATA step. When multiple input datasets are listed after the SET statement, SAS automatically reads all observations from the first listed dataset, followed sequentially by all observations from the second, and so on, until all listed input files have contributed their records to the newly created output dataset.

Understanding the Concatenation Principle

Concatenation is conceptually simple: it is additive along the row axis. If you have Dataset A with 10 rows and Dataset B with 15 rows, concatenating them will produce a new dataset, Dataset C, with 25 rows, assuming both datasets have the same columns. This differs significantly from data merging (using the `MERGE` statement), which aligns records side-by-side based on common keys.

To successfully perform this vertical combination, the fundamental requirement is that the schemas of the source datasets must align. Specifically, they should share the same variable names, and ideally, these variables should be defined with the same data type (e.g., character or numeric) and length. While SAS is relatively flexible and attempts type conversion when necessary, strict structural uniformity prevents unexpected warnings or truncation errors in the resulting data.

This technique is vital for tasks like compiling monthly sales reports into an annual summary or combining patient data collected across various clinical sites. It transforms disparate, but structurally identical, data files into one cohesive unit ready for analysis within the DATA step environment.

Prerequisites for Vertical Stacking

Before initiating the concatenation process using the SET statement, it is critical to confirm that the input datasets meet specific structural criteria. Adhering to these requirements ensures a clean, error-free combination and prevents unexpected data issues in the output file.

The primary prerequisite is that all source files must contain the same variables, spelled and cased identically. If variable names differ slightly (e.g., `ID_Number` vs. `IDNumber`), SAS will treat them as separate variables, and the resulting dataset will contain both, resulting in missing values for the dataset that did not contain the specific variable. Furthermore, the order in which the variables appear in the source datasets is less critical than the variable name itself, as SAS matches variables by name during concatenation.

Additionally, while not strictly required, consistency in variable attributes (such as length and format) is highly recommended. When two datasets define the same variable name but with different lengths (e.g., a character variable defined as length 10 in one dataset and length 20 in another), the variable definition is typically inherited from the first dataset listed in the `SET` statement. This can lead to the truncation of data from subsequent datasets if their values exceed the established length.

Basic Syntax for Data Concatenation

The concatenation process in DATA step is remarkably concise. You define a new target dataset and then list all source datasets sequentially after the `SET` keyword. This syntax is used whether you are combining two datasets or twenty.

To combine `data1` and `data2` into a new dataset called `data3`, you employ the following structure. Note that the order matters: all observations from `data1` will precede all observations from `data2` in the resulting file.

```
/*concatenate two datasets into one*/
data data3;
set data1 data2;
run;
```

The `DATA` statement initiates the creation of the new dataset (`data3`). The `SET` statement specifies the input datasets (`data1 data2`). Finally, the `RUN` statement executes the DATA step, reading every record from the source files and writing them in order to the target dataset. This simple methodology is the foundation of vertical data structure creation in SAS.

Example: Concatenate Datasets in SAS

To illustrate the practical application of Concatenation, let us begin by creating two sample datasets. These datasets represent two separate batches of scoring results, perhaps collected in different weeks, but they share the exact same structure: `firstName`, `lastName`, and `points` scored.

We will define `data1` to hold the results for the first group of participants and `data2` for the second group. It is essential to ensure consistency in variable types: `firstName` and `lastName` are character variables (denoted by the `$`), and `points` is a numeric variable.

```
/*create first dataset*/
data data1;
input firstName $ lastName $ points;
datalines;
Austin Smith 15
Brad Stevens 31
Chad Miller 22
;
run;

/*create second dataset*/
data data2;
input firstName $ lastName $ points;
datalines;
Dave Michaelson 19
Eric Schmidt 29
Frank Wright 20
Greg Gunner 40
Harold Anderson 35
;
run;

/*view datasets*/
proc print data=data1;
proc print data=data2;
```

Upon running the above code, the `PROC PRINT` statements confirm the contents and structure of our initial two source files. We observe that `data1` contains three observations, and `data2` contains five observations, totaling eight observations that we expect to see in our final concatenated file.

Obs	firstName	lastName	points
1	Austin	Smith	15
2	Brad	Stevens	31
3	Chad	Miller	22

Obs	firstName	lastName	points
1	Dave	Michaels	19
2	Eric	Schmidt	29
3	Frank	Wright	20
4	Greg	Gunner	40
5	Harold	Anderson	35

The visual inspection confirms that both source datasets possess identical variables (`firstName`, `lastName`, `points`), setting the stage perfectly for the vertical combination process. If the variables were mismatched, the concatenation would still execute, but the results would be fragmented, with null values filling the gaps where variables did not exist in a specific source file.

Executing the Concatenation Process

Having established our source datasets (`data1` and `data2`), we now apply the simple but powerful concatenation syntax using the SET statement. We define the target dataset as `data3`. This step demonstrates how efficiently multiple data streams can be unified into a single analytical view.

We use the following code block to perform the combination. After the `DATA step` completes the concatenation, we immediately use `PROC PRINT` on the new `data3` file to verify the successful integration of all records from the two source files. This immediate verification is a crucial step in ensuring data integrity.

```
/*concatenate two datasets into one*/
data data3;
set data1 data2;
run;

/*view new dataset*/
proc print data=data3;
```

Executing this code instructs SAS to read every observation from `data1` first, placing them into `data3`, and then, upon reaching the end of `data1`, to immediately transition and begin reading every observation from `data2`, appending them sequentially to `data3`. This sequential reading is the definitive characteristic of concatenation versus other data combination techniques.

Analysis of the Concatenated Output

The resulting dataset, `data3`, provides a unified view of all participants. By inspecting the output generated by `PROC PRINT`, we can confirm that the operation was successful and that the data integrity was maintained across the transition.

Obs	firstName	lastName	points
1	Austin	Smith	15
2	Brad	Stevens	31
3	Chad	Miller	22
4	Dave	Michaels	19
5	Eric	Schmidt	29
6	Frank	Wright	20
7	Greg	Gunner	40
8	Harold	Anderson	35

As evident in the output, the dataset `data3` now contains a total of eight observations. The first three rows correspond exactly to the contents of `data1` (Austin, Brad, Chad), followed immediately by the five observations from `data2` (Dave, Eric, Frank, Greg, Harold). The resulting dataset contains all of the observations from the first two source datasets, stacked vertically and preserving the order in which the source datasets were listed in the `SET` statement.

This result confirms the nature of Concatenation: it increases the number of rows (observations) while keeping the number of columns (variables) constant. Had we attempted a merge operation instead, we would have needed a common key, and the resulting file would likely have eight observations but potentially more columns, depending on the merge type used.

Important Considerations and Scalability

The flexibility of the `SET` statement allows for highly scalable operations. While our example demonstrated combining only two datasets, the underlying syntax allows you to list as many datasets as needed for the Concatenation process. The only structural requirement remains the shared variable names across all listed input files.

When dealing with a large number of datasets that need to be concatenated, especially if they are stored in a specific library and share a common naming convention (e.g., `sales_qtr1`, `sales_qtr2`, etc.), you can utilize abbreviated lists or wildcards within the `SET` statement. For instance, `SET sales_qtr1 -- sales_qtr4;` can be used if the datasets are cataloged sequentially in the SAS library.

A crucial note regarding data processing: If you need to perform conditional processing or introduce new variables that identify the source of the observation (e.g., adding a variable called `Source_File` with a value like 'Data1' or 'Data2'), you must use additional DATA step logic around the `SET` statement. The automatic variable `_N_` increments for every row read, but it doesn't intrinsically track the source file without explicit programming.

Comparison: Concatenation vs. Merging

New SAS users often confuse vertical stacking (concatenation) with horizontal joining (merging). While both combine data, their applications and underlying logic are fundamentally different, dictated by the structure of the resulting dataset and the variables involved. Understanding this distinction is paramount for correct data manipulation.

We can summarize the primary differences using the following structure:

Concatenation (`SET` Statement): This method stacks datasets vertically, increasing the number of observations (rows). It requires matching variable names but ignores observation order unless explicitly controlled. It is used when integrating time-series data or samples collected separately but structured identically.
Merging (`MERGE` Statement): This method joins datasets horizontally, often increasing the number of variables (columns). It requires a common key variable (specified using the `BY` statement) to align records. It is used to enrich a dataset with supplementary information, such as linking employee records to department details.

Choosing the correct method depends entirely on the analytical goal. If the goal is to create a bigger pool of records for a uniform analysis, Concatenation is the appropriate tool. If the goal is to link related information across different data tables based on a shared identifier, merging is necessary.

Further Data Management Resources in SAS

Mastering the `SET` statement for concatenation opens the door to more complex data management tasks. The ability to efficiently pool data is often the first step in a larger analytical workflow. Below are related tutorials explaining how to perform other common and essential tasks in SAS, leveraging the skills learned here:

The following tutorials explain how to perform other common tasks in SAS: