How to import a specific range from excel in a SAS program?

Authored by

stats writer

November 19, 2025

The most efficient way to import a specific range from an Excel spreadsheet into a SAS program is by utilizing the powerful PROC IMPORT statement. This procedure is specifically designed to handle external data sources and integrate them seamlessly into the SAS environment. By employing the appropriate options, users gain fine-grained control over the import process, allowing them to precisely define the subset of cells required, rather than importing the entire sheet.

Beyond simply pulling in raw data, the IMPORT procedure enables crucial data management functionalities. Specifically, it allows you to specify the characteristics of the resulting SAS dataset, including column names, data types, and required lengths for each variable. Furthermore, by defining the external data source via the DBMS option, SAS correctly interfaces with the file structure, treating the Excel file as a specialized database resource.

Selecting only the necessary range significantly optimizes performance, particularly when dealing with large Excel files that contain thousands of rows or numerous sheets. By limiting the scope of the data transfer, you reduce processing overhead, minimize resource consumption, and enhance the overall efficiency of your SAS sessions. This targeted approach is essential for maintaining streamlined workflows in data intensive projects.

Understanding the PROC IMPORT Statement

The primary mechanism for external data integration in SAS is the utilization of the **PROC IMPORT** statement, coupled with the crucial **RANGE** option, to precisely define the subset of cells required from an Excel workbook. This combination ensures that only the relevant data block is read and transformed into a SAS dataset, preventing unnecessary data loading and complexity.

When working with Excel files, SAS relies on the Microsoft ACE (Access Database Engine) provider or similar drivers to interpret the file structure. Therefore, correctly specifying the database management system (**DBMS**) type is paramount for successful execution. For modern Excel files (`.xlsx` format), specifying **DBMS=xlsx** informs SAS how to handle the data structure, including sheet names and cell coordinates.

The foundational syntax required to execute a targeted import operation involves defining the output dataset name, providing the exact location of the source file, specifying the file type, handling existing data, managing variable naming conventions, and crucially, defining the boundary of the data using the **RANGE** parameter.

Key Options for Range Specification

You can use the **PROC IMPORT** statement with the **RANGE** option to import a specific range of

cells from an Excel spreadsheet into SAS. This option accepts standard Excel A1 notation, often prefixed by the sheet name, providing precise coordinate control over the data selection.

The following basic syntax demonstrates the necessary components for achieving a successful, range-specific data import:

```
/* Example: Importing targeted data from the Excel file called basketball_data.xlsx */
proc import out=my_data
datafile="/home/u13181/basketball_data.xlsx"
dbms=xlsx
replace;
getnames=YES;
range="Sheet1$C4:E11";
run;
```

This code block initiates the import process, directing SAS to read the specified Excel file and create a new dataset. The **RANGE** parameter is the core differentiator here, ensuring that only data points within C4 and E11 on Sheet1 are considered for the transformation into the **my_data** SAS dataset.

Detailed Explanation of PROC IMPORT Options

To fully leverage the power of the import procedure, it is essential to understand the function of each option used within the syntax. These options control where the data goes, where it comes from, how it is interpreted, and what happens if the destination dataset already exists.

The successful execution of `PROC IMPORT` hinges upon setting these options correctly according to your environment and data needs. Misconfiguration of parameters like **DBMS** or **datafile** location will result in errors, halting the import process immediately. The use of the **RANGE** option transforms the import from a blanket operation to a surgical selection.

Here's a detailed breakdown of the critical options utilized in the PROC IMPORT statement:

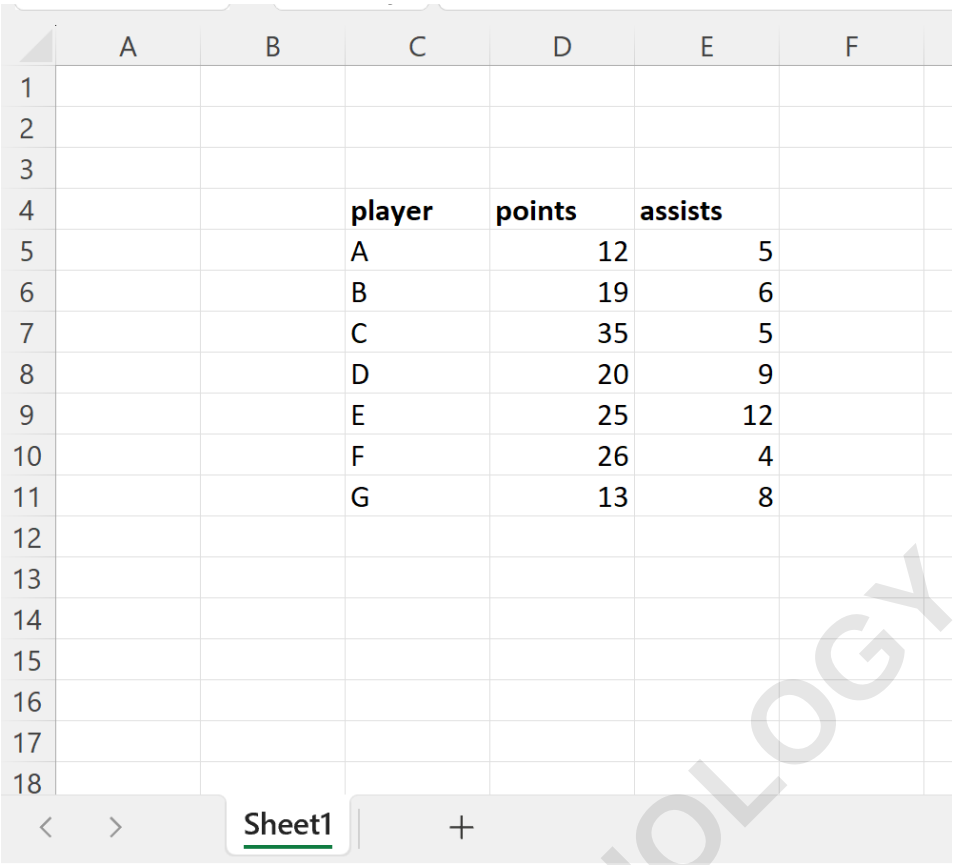
out: This mandatory option defines the name of the output SAS dataset once the data has been successfully imported and processed by SAS.
datafile: This option specifies the exact operating system path and filename of the external Excel file you intend to import.
dbms: This is crucial for defining the format of the external file being imported. For modern Excel files (`.xlsx`), the value must be set to **xlsx**. For older formats (`.xls`), **excel** or other appropriate drivers might be necessary. This tells SAS which driver to use to interpret the external data structure, which is vital for proper handling of range specifications and data types.

replace: This optional statement instructs SAS to overwrite the output dataset if a dataset with the same name already exists in the current library. Omitting this might cause the procedure to fail if the dataset already exists.
getnames: Setting this option to **YES** (the default behavior) ensures that the first row of the imported data range is utilized as the variable names (column headers) in the resulting SAS dataset. If the first row does not contain meaningful header information, this should be set to **NO**, and SAS will assign default column names (e.g., COL1, COL2, etc.).
range: This is the critical parameter for targeted imports. It specifies the specific coordinates of cells--including the sheet name and cell boundaries (e.g., "Sheet1$C4:E11")--from which data should be extracted.

A key aspect of the **RANGE** definition is the inclusion of the sheet name, followed by a dollar sign ($) and the standard Excel A1 range notation. Note that this particular example imports the cells within the rectangular block defined by **C4:E11** on **Sheet1** from the source Excel workbook called **basketball_data.xlsx**. This precise definition ensures that extraneous headers, footers, or summary rows outside this block are excluded.

Visualizing the Source Data

To fully appreciate how the **RANGE** option impacts the data structure, it is helpful to first examine the original structure of the Excel source file. The following image represents the complete layout of the **basketball_data.xlsx** file before the targeted import operation is executed.



	A	B	C	D	E	F
1						
2						
3						
4			**player**	**points**	**assists**	
5			A	12	5	
6			B	19	6	
7			C	35	5	
8			D	20	9	
9			E	25	12	
10			F	26	4	
11			G	13	8	
12						
13						
14						
15						
16						
17						
18						

Sheet1

As illustrated, the Excel sheet contains various elements, including potential titles (A1:E2), blank rows, and metadata. The actual structured dataset begins only around row 3 or 4. If we intend to import only the player statistics (Player Name, Points, Rebounds), we must carefully define the range to exclude any introductory material and ensure that the headers are correctly captured.

In this specific scenario, we observe that the data headers (Name, Points, Rebounds) start at row 4, and the data block continues down. Defining the range incorrectly could result in missing variable names or importing irrelevant metadata rows, thereby contaminating the analytical process.

Example 1: Importing Data Without Specifying a Range

To establish a baseline for comparison, we will first demonstrate the behavior of PROC IMPORT when the **RANGE** option is entirely omitted. When this option is absent, SAS attempts to automatically detect the usable data block within the Excel sheet, starting from the first non-empty cell. This often results in the import of extra rows or columns if the sheet is not perfectly clean.

We can use the following syntax to import the entire relevant portion of the Excel file into a SAS dataset named **my_data** without explicitly specifying a range of cells to import:

```
/* Import data from Excel file called basketball_data.xlsx without range specification */
proc import out=my_data
datafile="/home/u13181/basketball_data.xlsx"
dbms=xlsx
replace;
getnames=YES;
run;

/* View the resulting dataset using PROC PRINT */
proc print data=my_data;
```

Upon execution, SAS reads the Excel file, identifies the boundaries automatically, and creates the temporary dataset **my_data**. The subsequent `PROC PRINT` statement displays the contents of the newly created dataset, allowing us to inspect exactly what data SAS interpreted as the relevant block.

Analysis of Example 1 Results

The outcome of the range-omitted import clearly illustrates the importance of explicit range specification when the source sheet contains ancillary information. Because the Excel file had descriptive titles and possibly blank rows preceding the actual column headers, SAS included these elements in the automatic import boundary.

Obs	A	B	C	D	E
1					
2					
3			player	points	assists
4			A	12	5
5			B	19	6
6			C	35	5
7			D	20	9
8			E	25	12
9			F	26	4
10			G	13	8

Since we didn't use the **RANGE** statement to specify a range of cells to import, SAS initiated the import from the very first cell containing data and continued until it encountered a fully empty row

or column. In this visualization, it appears SAS captured the overall header "Basketball Team Performance" as the first variable, resulting in skewed data interpretation and inappropriate variable names.

This illustrates a significant limitation of relying on automatic detection. If the Excel sheet is not structured such that the data table starts exactly where SAS assumes, the resulting SAS dataset will likely contain unwanted rows that must be manually cleaned, defeating the purpose of efficient data loading.

Example 2: Importing Data from Excel File into SAS and Specify Range

The superior method involves applying the **RANGE** option to isolate the clean, structured data block, ensuring that SAS begins reading exactly where the meaningful data headers start and ends precisely where the data concludes. This provides a robust and repeatable method for handling complex external source files.

We can use the following syntax to import the Excel file into a SAS dataset named **my_data**, this time utilizing the **RANGE** option to import only the specific, clean range of cells: **Sheet1$C4:E11**. This range assumes that the usable data headers are located in row 4 and the statistics continue until row 11, specifically isolating the three columns C, D, and E.

```
/* Import specific cells from Excel file called basketball_data.xlsx using the RANGE option */
proc import out=my_data
datafile="/home/u13181/basketball_data.xlsx"
dbms=xlsx
replace;
getnames=YES;
range="Sheet1$C4:E11";
run;

/* View the resulting, cleaned dataset */
proc print data=my_data;
```

The execution of this procedure is markedly different. By enforcing the range "Sheet1$C4:E11", we instruct SAS to ignore all rows above C4 and all columns outside C, D, and E. Since **GETNAMES=YES** is specified, SAS correctly treats row 4 (within the defined range) as the variable headers for the resulting dataset.

Obs	player	points	assists
1	A	12	5
2	B	19	6
3	C	35	5
4	D	20	9
5	E	25	12
6	F	26	4
7	G	13	8

As demonstrated by the output, the imported SAS dataset is now clean, containing only the desired variables (Name, Points, Rebounds) and the corresponding eight observations. The extraneous header information has been successfully excluded, resulting in a dataset immediately ready for statistical analysis without further cleaning steps.

Leveraging Named Ranges for Flexibility

While using standard A1 notation (e.g., C4:E11) is effective, it can become fragile if the source Excel spreadsheet undergoes structural changes, such as the insertion or deletion of rows and columns. A more robust technique is to define a named range directly within the Excel file and reference that name within the SAS **RANGE** option.

Note that you can also pass a named range defined in the Excel spreadsheet to the **RANGE** statement. Named ranges provide a level of abstraction: even if the data block moves (e.g., from C4:E11 to D5:F12), the name remains valid, and the SAS code does not require modification.

For example, if the cell range **C4:E11** had a named range assigned in Excel called **my_stats_data**, then you would simplify the argument in the **PROC IMPORT** statement to **range="my_stats_data"**. When using a named range, you generally omit the sheet name prefix (the $ operator), as the name itself is globally recognized within the workbook.

Note: You can find the complete, authoritative documentation for the **PROC IMPORT** statement in SAS's official resources.

Conclusion and Further Resources

The use of the **RANGE** option within PROC IMPORT is the definitive method for efficiently and accurately transferring specific blocks of data from Excel into the SAS program environment. This precision ensures data integrity and minimizes the subsequent data cleaning required for analysis.

For users looking to expand their SAS skills, the following tutorials explain how to perform other common data management and analysis tasks: