Learning SAS: Using the IN= Option to Identify Input Datasets in the DATA Step

Authored by

Mohammed looti

November 14, 2025

RECOMMENDED CITATION

Mohammed looti (2025). *Learning SAS: Using the IN= Option to Identify Input Datasets in the DATA Step*. PSYCHOLOGICAL STATISTICS. Retrieved from https://statistics.arabpsychology.com/?p=1419

The **IN= option** within the **SAS** programming environment stands as a critical tool for managing data lineage and ensuring robust **data provenance**, particularly during complex data integration tasks. This feature operates exclusively within the **DATA step**, where its fundamental role is to generate a temporary **Boolean variable**. This variable acts as an essential flag, precisely indicating whether the record currently being processed originated from a specific input **dataset** specified by the programmer. This capability is absolutely indispensable when integrating multiple streams of information, as it provides immediate, observation-level source tracking.

The true power of this specialized **option** becomes evident when analysts face the challenge of combining or integrating several distinct **datasets**. Whether the process involves complex merging based on key variables or simple concatenation (stacking records), maintaining the traceability of each record is paramount. Without this immediate source identification, critical functions like data validation, compliance auditing, and subsequent conditional analytical processing become significantly more difficult and susceptible to error. The **IN= option** provides an automated, elegant solution by dynamically creating a binary flag that pinpoints the exact contributing source for every individual **observation** in the resulting output file.

To grasp the strategic utility and practical significance of the **IN= option**, we will walk through a detailed, hands-on example. This demonstration will clearly lay out the necessary syntax and programming logic required to effectively identify and permanently tag the origin of **observations** when unifying multiple input **datasets**. Mastering this technique grants **SAS** programmers essential clarity and superior management capabilities over their consolidated data structures.

Setting Up Our Example Datasets

For this practical illustration, we will use a common data integration scenario involving sports statistics. Our objective is to combine information regarding basketball players from two separate geographic regions within a league. To simulate this, we will establish two distinct **datasets**: the first, named `east_data`, will contain player records exclusively from the Eastern Conference, and the second, `west_data`, will hold comparable data for players from the Western Conference.

Each foundational **dataset** will incorporate simple yet representative variables, specifically the team name and the points scored by a player during a game. Our primary analytical goal is to unify these two separate data sources into one cohesive master file. Crucially, we must then deploy the **IN= option** technique to accurately and explicitly track which conference provided the original record for each player in the final combined file. This arrangement is highly reflective of real-world data merging and auditing requirements.

The following **SAS** code block executes the creation of our two foundational data sources, `east_data` and `west_data`, utilizing the efficient `DATALINES` statement for input. It concludes with a fundamental **PROC PRINT** procedure. This procedure serves to verify the contents of both newly

generated datasets, confirming their structure and population before we proceed with the data combination process.

```
/*create East dataset*/
data east_data;
input team $ points;
datalines;
Celtics 22
Pistons 14
Nets 35
Hornets 19
Magic 22
;
run;

/*create West dataset*/
data west_data;
input team $ points;
datalines;
Mavs 40
Rockets 39
Warriors 23
Lakers 19
Clippers 25
;
run;

/*view datasets*/
proc print data=east_data;
proc print data=west_data;
```

Obs	team	points
1	Celtics	22
2	Pistons	14
3	Nets	35
4	Hornets	19
5	Magic	22

Obs	team	points
1	Mavs	40
2	Rockets	39
3	Warriors	23
4	Lakers	19
5	Clippers	25

Combining Datasets Without Source Identification

Before we introduce the advanced source identification capabilities provided by the **IN= option**, it is necessary to first establish the standard, default process for combining our two source **datasets**, `east_data` and `west_data`. The operation of stacking or chaining datasets, commonly referred to as **appending** or concatenation, is fundamentally accomplished within the **DATA step** using the powerful **SET statement**.

The required mechanism for simple concatenation is highly straightforward: we merely list the names of the source **datasets**, `east_data` and `west_data`, immediately following the **SET statement** in the desired reading sequence. This command instructs the **DATA step** compiler to read all **observations** from the first dataset (East), and upon completion, sequentially read all records from the second dataset (West). The result is a new, consolidated **dataset**, which we have named `all_data`, containing all original records stacked one after the other.

Observe the code below, which executes this basic concatenation and then uses **PROC PRINT** to display the unified output. Notice that while the ten records are present, there is no inherent variable that reveals whether a specific player originated from the Eastern or Western Conference. This ambiguity highlights the core problem the **IN= option** is designed to solve.

/*create new dataset*/

```
data all_data;
set east_data west_data;
run;

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

Obs	team	points
1	Celtics	22
2	Pistons	14
3	Nets	35
4	Hornets	19
5	Magic	22
6	Mavs	40
7	Rockets	39
8	Warriors	23
9	Lakers	19
10	Clippers	25

Leveraging IN= to Identify Data Sources

Following the previous concatenation, a quick review of the `all_data` **dataset** confirms that all ten player records have been successfully unified. However, a critical piece of metadata is absent: there is no explicit variable to indicate whether a given player record originally belonged to the Eastern or Western Conference source file. This lack of source context is a severe limitation for any subsequent **conditional analysis** or auditing process.

The definitive solution to this traceability challenge is the strategic application of the **IN= option** to the input **datasets** listed within the **SET statement**. When utilized, this option instructs **SAS** to automatically generate a temporary, binary **Boolean variable** (which we label `i` in our example) for each source dataset to which the option is attached. This variable is automatically assigned a value of `1` (True) if the current **observation** is being read from that specific dataset, and `0` (False) otherwise.

By integrating this temporary flag into the subsequent logic of the **DATA step**, we can create a permanent, descriptive variable that accurately reports the source. In the code below, we apply `IN=i` specifically to `west_data`. We then employ an `IF-THEN-ELSE` structure: if the variable `i` is

true (meaning the record came from the West), we assign the new variable `conf` the value 'West'; otherwise, we assign it 'East'. This simple construct completely resolves the ambiguity of data origin within our combined **dataset**, providing crucial context for every record.

```
/*create new dataset*/
data all_data;
set east_data west_data(in=i);
if i then conf='West';
else conf='East';
run;

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

Obs	team	points	conf
1	Celtics	22	East
2	Pistons	14	East
3	Nets	35	East
4	Hornets	19	East
5	Magic	22	East
6	Mavs	40	West
7	Rockets	39	West
8	Warriors	23	West
9	Lakers	19	West
10	Clippers	25	West

The resulting output meticulously confirms the efficiency of this technique. By attaching `IN=i` to `west_data`, the temporary variable `i` registers as `1` for all records sourced from the West and `0` for those from `east_data`. The subsequent conditional programming logic successfully translates this **Boolean variable** into the permanent character variable `conf`, which correctly labels 'West' or 'East' for every **observation**. This methodology provides the foundational context required for any accurate, source-aware analysis of the combined data.

Refining Source Identification for Single-Source Flagging

While the previous methodology used a comprehensive `IF-THEN-ELSE` structure to provide

complete dual-source identification, many specialized programming tasks only necessitate flagging or highlighting **observations** originating from one specific input **dataset**. The inherent flexibility of the **IN= option** is perfectly suited to this simpler requirement, allowing for cleaner conditional logic without the complexity of an explicit `ELSE` clause.

In this refined application, the **IN= option** is assigned exclusively to the particular input **dataset** whose records require specific marking. The temporary **Boolean variable** generated by **SAS** is then utilized within a simple `IF` condition. The new flag variable is only assigned a non-missing value when this condition is true--meaning the **observation** originated from the flagged source. For all other records coming from the unflagged source, the new variable will simply retain its default missing value, defined by the variable type **SAS** established during the compilation phase.

In the code example below, we choose to flag only the Eastern Conference players. We apply `IN=i` to `east_data`, while `west_data` remains unflagged. We create a new variable, `east_conf`, using a simple `IF` statement. This method demonstrates an elegant approach to isolating specific subsets of data for further specialized processing or reporting without generating source tags for all records.

```
/*create new dataset*/
data all_data;
set east_data(in=i) west_data;
if i then east_conf='*';
run;

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

Obs	team	points	east_conf
1	Celtics	22	*
2	Pistons	14	*
3	Nets	35	*
4	Hornets	19	*
5	Magic	22	*
6	Mavs	40	
7	Rockets	39	
8	Warriors	23	
9	Lakers	19	
10	Clippers	25	

As shown in the output, the `IN=i` option, applied to `east_data`, results in the temporary variable `i` being `1` exclusively for Eastern Conference players. The following `IF i THEN east_conf='*';` statement successfully creates the variable `east_conf`, assigning it the value `'*'` only for records sourced from `east_data`. Crucially, for all **observations** originating from `west_data`, the condition `i` is false, and `east_conf` remains blank or missing. This technique is highly valuable when the primary goal is to isolate or highlight a specific subset of records derived from one source for targeted processing, ensuring maximum code efficiency and clarity.

Key Benefits and Strategic Applications of the IN= Option

The **IN= option**, despite its deceptively concise syntax, provides profound and necessary advantages in advanced **SAS** programming, particularly when managing large-scale data integration projects. Its principal benefit is the provision of granular control and crystal-clear visibility over the origin of every single **observation** throughout concatenation or merging processes executed during the **DATA step**.

The utility of this feature extends significantly beyond mere source identification; the automatically generated **Boolean variable** enables several critical and advanced data manipulation techniques. Proficient programmers frequently leverage this mechanism to achieve the following operational goals, enhancing the robustness of their data pipelines:

Filter Data Efficiently: It allows for the easy selection or, conversely, the exclusion of specific **observations** based strictly on their source **dataset**, which streamlines the creation of highly specialized subsets for analysis.

Perform Conditional Processing: Specific, complex transformations, necessary aggregations, or specialized calculations can be applied exclusively to records originating from one particular source. This ensures accurate data integrity when working with heterogeneous inputs.

Audit and Validate Merges: The source flag serves as an internal auditing tool, enabling developers to quickly verify the integrity and completeness of merged **datasets** by confirming that all expected records from each source have been correctly incorporated and tagged without loss.

Create Robust Identifiers: New variables can be constructed that intelligently combine existing data elements with the source information, thereby providing a more comprehensive and robust context necessary for subsequent analytical models and reporting systems.

Developing a strong, foundational understanding of how to effectively utilize the **IN= option** is a key indicator of proficient and precise **DATA step** programming. This critical skill empowers users to manage, audit, and analyze intricate data structures with increased confidence, accuracy, and efficiency.

Continuing Your SAS Programming Journey

The **IN= option** represents just one critical facet of the powerful and extensive feature set available in the **SAS** language for comprehensive data manipulation and analysis. To further advance your **SAS** programming expertise, we strongly encourage exploring supplemental tutorials and official documentation that address a wide array of common data tasks. These resources are invaluable for mastering various aspects of the **SAS** language, ranging from essential data handling techniques like merging and subsetting to complex statistical procedures.

The following resources explain how to perform other common and necessary data manipulation tasks in **SAS**: