How to Easily Calculate Percentiles in SAS

Authored by

stats writer

December 1, 2025

Understanding the distribution of a dataset is fundamental in statistical analysis. A key metric used for this purpose is the percentile, which indicates the value below which a given percentage of observations in a group of observations falls. While the concept involves sorting the data and identifying specific data points, the SAS statistical software simplifies this calculation significantly.

To accurately determine percentiles using SAS, analysts primarily rely on the powerful PROC UNIVARIATE procedure. This procedure not only handles the necessary sorting internally but also provides a streamlined mechanism using the OUTPUT statement to save the calculated values into a new dataset for subsequent analysis. Mastering this procedure allows for precise and efficient calculation of critical statistical markers.

Calculating percentiles in SAS can be accomplished using several flexible methods, depending on whether you need a single value, multiple specific values, or values calculated across defined groups within your data. Here, we outline the three most common and effective approaches utilizing SAS programming syntax.

Method 1: Calculating a Single Specific Percentile Value

This method is used when you need to quickly determine the value corresponding to one specific percentile rank (e.g., the 70th percentile) for a given variable. We use the PROC UNIVARIATE procedure, specifying the variable of interest using the VAR statement and defining the target percentile via the PCTLPPTS option within the OUTPUT statement. This technique is essential for focused statistical inquiries.

```
/*Calculate the 70th percentile value for var1 using PROC UNIVARIATE*/
proc univariate data=original_data;
var var1;
output out=percentile_data
pctlpts = 70
pctlpre = P_;
run;
```

Method 2: Calculating Multiple Specific Percentile Values

Often, statistical analysis requires viewing the distribution across several key points, such as the 70th, 80th, and 90th percentiles simultaneously. Instead of running separate procedures, PROC UNIVARIATE allows users to specify multiple percentile points within a single PCTLPPTS option. This dramatically increases efficiency when exploring data quartiles or other distributional splits. The values are simply listed separated by spaces.

```
/*Calculate the 70th, 80th, and 90th percentile values for var1 in one step*/
proc univariate data=original_data;
var var1;
output out=percentile_data
pctlpts = 70 80 90
pctlpre = P_;
run;
```

Method 3: Calculating Percentiles by Group

When analyzing heterogeneous datasets, it is often necessary to calculate percentiles separately for different subgroups defined by a categorical variable (e.g., calculating sales percentiles for each region or test scores for each grade level). In SAS, this requires two steps: first, sorting the input data using PROC SORT based on the grouping variable, and second, applying the BY statement within PROC UNIVARIATE.

The BY statement ensures that the analysis (in this case, the percentile calculation) is performed independently for every unique value of the grouping variable. This is vital for segmenting data analysis and gaining detailed insights into population subsets.

```
/*First, sort original data by the grouping variable (var2)*/
proc sort data=original_data;
by var2;
run;

/*Next, calculate percentiles for var1 grouped by var2 using the BY statement*/
proc univariate data=original_data;
var var1;
by var2;
output out=percentile_data
pctlpts = 70, 80, 90
pctlpre = P_;
run;
```

Key Syntax Parameters Explained

The power of generating percentiles in PROC UNIVARIATE lies in the options used within the OUTPUT statement. Two parameters are absolutely essential for defining and labeling the resulting percentile values.

Note: The **pctlpts** statement specifies which percentile ranks (expressed as percentages from 1 to 100) should be calculated and outputted to the new dataset. The **pctlpre** statement specifies the prefix to use for naming the newly created percentile variables in the output dataset, ensuring clarity and ease of data manipulation. For example, using `P_` results in variables named P_70, P_80, etc.

Setting Up the Sample Data

To demonstrate these three methods effectively, we will use a small sample dataset containing scores (variable: `points`) achieved by two different groups (variable: `team`). This allows us to illustrate both overall and group-specific percentile calculations.

The following SAS code block creates the necessary dataset named `original_data` and then uses PROC PRINT to display the contents, ensuring the data is loaded correctly before proceeding with the analysis.

```
/*Create the sample dataset named original_data*/
data original_data;
input team $ points;
datalines;
A 12
A 15
A 16
A 21
A 22
A 25
A 29
A 31
B 16
B 22
B 25
B 29
B 30
B 31
B 33
B 38
;
run;

/*View the created dataset using PROC PRINT*/
```

proc print data=original_data;

Obs	team	points
1	A	12
2	A	15
3	A	16
4	A	21
5	A	22
6	A	25
7	A	29
8	A	31
9	B	16
10	B	22
11	B	25
12	B	29
13	B	30
14	B	31
15	B	33
16	B	38

Example 1: Calculating a Single Specific Percentile Value

Our first demonstration focuses on finding the value that represents the 70th percentile across the entire `points` variable in the `original_data` dataset, ignoring the team groupings for this step. This calculation identifies the score below which 70% of all observations fall.

We execute PROC UNIVARIATE, specifying `points` as the variable and setting pctlpts = 70. The results are stored in `percentile_data` and then immediately viewed using PROC PRINT to interpret the output.

```
/*Calculate 70th percentile value for points across all observations*/
proc univariate data=original_data;
var points;
output out=percentile_data
pctlpts = 70
pctlpre = P_;
run;
```

/*View the output dataset containing the percentile value*/
proc print data=percentile_data;

Obs	P_70
1	30

The resulting output dataset, `percentile_data`, contains a single observation with a new variable, `P_70`. The calculated value at the 70th percentile is determined to be **30**. This means that 70% of all data points in the combined dataset have a score of 30 or less.

Example 2: Calculating Multiple Specific Percentile Values

Building on the previous example, we now calculate three important descriptive statistics simultaneously: the 70th, 80th, and 90th percentile scores for the `points` variable. This approach is highly useful when trying to understand the upper tail of the data distribution.

By listing the desired percentile values (70, 80, 90) in the PCTLPPTS option, PROC UNIVARIATE generates three new variables in the output dataset: `P_70`, `P_80`, and `P_90` (thanks to the `P_` prefix defined by PCTLPRE).

/*Calculate 70th, 80th, and 90th percentile values for points using a single procedure call*/
proc univariate data=original_data;
var points;
output out=percentile_data
pctlpts = 70 80 90
pctlpre = P_;
run;

Upon execution, the resulting dataset will contain the following calculated values:

Obs	P_70	P_80	P_90
1	30	31	33

Here's how to interpret the multiple percentile outputs generated by this method:

The value at the 70th percentile (`P_70`) is **30**.

The value at the 80th percentile (`P_80`) is **31**.
The value at the 90th percentile (`P_90`) is **33**.

Example 3: Calculating Percentiles by Group

The final and most advanced method demonstrates how to perform separate percentile calculations for different subgroups within the dataset. We want to find the 70th, 80th, 90th, and 95th percentile scores for `points`, calculated individually for `team A` and `team B`.

As noted previously, using the BY statement in PROC UNIVARIATE requires the input data to be sorted first according to the grouping variable (`team`). If the data is not sorted, SAS will return an error or unexpected results. After sorting, the BY team statement ensures the procedure cycles through each unique team value, calculating the requested percentiles for that specific group.

```
/*Step 1: Sort original data by the grouping variable 'team'*/
proc sort data=original_data;
by team;
run;

/*Step 2: Calculate percentiles for points grouped by team*/
proc univariate data=original_data;
var points;
by team;
output out=percentile_data
pctlpts = 70, 80, 90 95
pctlpre = P_;
run;
```

Obs	team	P_70	P_80	P_90	P_95
1	A	25	29	31	31
2	B	31	33	38	38

This output table clearly shows the values for the 70th, 80th, 90th, and 95th percentile for the **points** variable, calculated separately for both Team A and Team B. For instance, the 90th percentile for Team A is 30, whereas the 90th percentile for Team B is 33, indicating a stronger performance in the upper range for Team B.

Summary and Next Steps in SAS Analysis

The PROC UNIVARIATE procedure offers a flexible and robust environment for calculating various types of percentiles in SAS. Whether your goal is a single descriptive statistic or a detailed, group-wise distributional analysis, the combination of VAR, BY, and OUTPUT statements provides the necessary analytical power.

Once you have mastered percentile calculation, you may wish to explore other common statistical tasks in SAS to further enhance your data analysis workflow. These skills are foundational for complex modeling and reporting.

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