How to Easily Sample Data with PROC SURVEYSELECT in SAS

Authored by

stats writer

November 19, 2025

Understanding PROC SURVEYSELECT in SAS

The PROC SURVEYSELECT procedure is a fundamental tool within the SAS system, specifically designed for drawing samples from complex populations. Its primary purpose is to facilitate rigorous survey research and statistical analysis by ensuring that the selected sample accurately represents the target population based on predefined criteria. Unlike simple data subsetting, this procedure incorporates statistical rigor, enabling users to implement various probability sampling techniques, which are crucial for valid inference. It provides precise control over the selection process, guaranteeing that every observation has a known, non-zero probability of being included in the sample, which is the cornerstone of unbiased statistical sampling.

This powerful utility is essential when dealing with large datasets where analyzing the entire population is infeasible or unnecessary. It supports a wide array of sampling methods, including but not limited to simple random sampling, stratified sampling, cluster sampling, and systematic sampling. By employing PROC SURVEYSELECT, researchers can control sample size, allocation across strata, and cluster definition, ensuring the resultant dataset is optimally tailored for subsequent analysis procedures like PROC SURVEYMEANS or PROC SURVEYREG.

Effective implementation of this procedure often requires specifying several key options: the input dataset (`DATA=`), the output dataset (`OUT=`), the sampling approach (`METHOD=`), and specific parameters such as the desired sample size (`N=`) or the seed value (`SEED=`) for reproducibility. The ability to use the `SEED` option is particularly valuable, as it ensures that the exact same random sample can be generated repeatedly, a critical requirement for maintaining transparency and verifiability in research findings.

Key Sampling Techniques Supported by PROC SURVEYSELECT

PROC SURVEYSELECT is highly versatile, accommodating diverse research needs by supporting multiple sampling techniques. Choosing the correct method depends entirely on the structure of the population data and the specific research objectives concerning efficiency and representation. The procedure is designed to handle both unistage and multistage designs with equal probability or probability proportional to size (PPS) selection.

The three most frequently utilized methods, which we will detail with practical SAS code examples, include:

Simple Random Sampling (SRS): This is the most fundamental method, where every possible sample of a given size has an equal chance of being selected. This is achieved by specifying the option `METHOD=SRS`. It is straightforward to implement but may not always be the most efficient approach if the population has known sub-groups that need specific representation.

Stratified Random Sampling: This technique involves dividing the population into non-overlapping subgroups (strata) that are homogeneous with respect to certain characteristics (e.g., gender, geographical region, or, in our example, team). A simple random sample is then drawn independently within each stratum. This method often yields more precise estimates than SRS, provided the strata are relevant to the analysis. The `STRATA` statement is used to define these groups.

Clustered Random Sampling: Used when the population is naturally grouped into clusters (e.g., schools or teams). Instead of sampling individuals, entire clusters are sampled randomly. All observations within the selected clusters are typically included in the final sample. This is implemented using the `CLUSTER` statement and is often more cost-effective when the population is geographically dispersed.

It is crucial to understand that while these methods rely on random selection, the specific parameters used (like the number of observations or clusters chosen) directly influence the statistical properties of the resulting sample. Proper planning and method selection are vital before executing the sampling procedure.

Illustrating Sampling Options with Initial Code Snippets

Before diving into the full examples, here is a conceptual overview of the syntax used for the three primary sampling methods within SAS. These preliminary snippets outline the core commands necessary for generating samples.

The core command structure uses `DATA=` for the source dataset, `OUT=` for the resulting sample dataset, and `METHOD=` to define the approach.

Code Snippet for Simple Random Sample (SRS):

```
proc surveyselect data=my_data
out=my_sample
method=srs /*use simple random sampling*/
n=5 /*select a total of 5 observations*/
seed=1; /*set seed to make this example reproducible*/
run;
```

This command selects five random observations from the entire population defined in `my_data`.

Code Snippet for Stratified Random Sample:

```
proc surveyselect data=my_data
```

```
out=my_sample
method=srs /*use simple random sampling*/
n=2 /*select 2 observations from each strata*/
seed=1; /*set seed to make this example reproducible*/
strata grouping_var; /*specify variable to use for stratification*/
run;
```

Here, the `STRATA` statement forces the procedure to sample independently within subgroups, selecting two observations from each unique category found in `grouping_var`.

Code Snippet for Clustered Random Sample:

```
proc surveyselect data=my_data
out=my_sample
n=2 /*select 2 clusters*/
seed=1; /*set seed to make this example reproducible*/
cluster grouping_var; /*specify variable to use for stratification*/
run;
```

The `CLUSTER` statement identifies the sampling units as groups rather than individuals. This code selects two random clusters, and all associated observations within those clusters are included in the sample.

Setting Up the Demonstration Dataset

To effectively demonstrate the functionality of SAS sampling procedures, we will first construct a simple, relatable dataset. This dataset models fictional basketball player statistics, where players belong to distinct teams (serving as potential strata or clusters) and have an associated points score. This structure allows us to illustrate how sampling works both across the entire population and within specific subgroups.

The following SAS data step creates a temporary dataset named `my_data` containing player team assignments and their recorded points. We then use `PROC PRINT` to inspect the structure and content of our population dataset, confirming the availability of 14 observations spread across three teams (A, B, and C).

```
/*create dataset*/
data my_data;
input team $ points;
datalines;
```

```
A 12
A 14
A 22
A 35
A 40
B 12
B 10
B 29
B 33
C 40
C 25
C 11
C 10
C 15
;
run;

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

The output image below confirms the structure of the dataset, showing the 14 observations across the three distinct teams. This dataset serves as our complete population from which we will draw various statistical samples using PROC SURVEYSELECT.

Obs	team	points
1	A	12
2	A	14
3	A	22
4	A	35
5	A	40
6	B	12
7	B	10
8	B	29
9	B	33
10	C	40
11	C	25
12	C	11
13	C	10
14	C	15

Example 1: Implementing Simple Random Sampling (SRS)

Simple random sampling (SRS) is often the baseline method used when there is no prior knowledge suggesting that subgroups within the population behave differently, or when the goal is a basic representation of the whole. In this scenario, we aim to select a fixed number of observations--specifically five players--from the entire pool of 14 players without consideration of their team affiliation. This ensures that every individual player has an equal probability of being included in the final sample.

To perform SRS using PROC SURVEYSELECT, we must specify `METHOD=SRS`, which instructs SAS to treat the population as a single, uniform entity for selection purposes. We set `N=5` to dictate the required sample size. Importantly, setting the `SEED=1` option guarantees that if this procedure is run again, the exact same set of random observations will be generated, which is critical for ensuring reproducibility in survey research.

The complete SAS syntax below executes the SRS selection and then immediately outputs the resultant sample dataset, `my_sample`.

```
proc surveyselect data=my_data
out=my_sample
method=srs /*use simple random sampling*/
n=5 /*select a total of 5 observations*/
```

seed=1; /*set seed to make this example reproducible*/
run;

/*view sample*/
proc print data=my_sample;

The resulting sample, visualized below, shows five observations selected randomly from the population. In this specific seeded run, the sample happened to include players primarily from Teams A and C, demonstrating that SRS does not inherently guarantee proportional representation of underlying subgroups.

Obs	team	points
1	B	10
2	B	33
3	C	40
4	C	25
5	C	11

Example 2: Executing Stratified Random Sampling

When researchers need to guarantee representation from specific subgroups within the population, stratified random sampling is the preferred method. This technique requires dividing the population into non-overlapping strata (in our case, the teams A, B, and C) and then conducting independent simple random sampling within each stratum. This approach ensures that the sample reflects the distribution of the strata, leading to more precise estimates for population parameters.

To implement stratification, the key is the `STRATA` statement in PROC SURVEYSELECT, followed by the variable defining the strata. We set `N=2`, instructing SAS to randomly select exactly two observations from *each* stratum defined by the grouping variable. Since there are three teams, the total sample size will be 2 multiplied by 3, resulting in 6 total observations, guaranteeing equal representation from all teams.

The code below demonstrates this implementation. Note the use of `METHOD=SRS` within the strata, meaning the selection process within each team is still a random selection. This combination is essential for controlled representation across all necessary sub-populations for accurate survey research.

proc surveyselect data=my_data

```
out=my_sample
method=srs /*use simple random sampling within strata*/
n=2 /*select 2 observations from each strata*/
seed=1; /*set seed to make this example reproducible*/
strata grouping_var; /*specify variable to use for stratification*/
run;

/*view sample*/
proc print data=my_sample;
```

As evidenced by the output image below, the resulting sample successfully contains two randomly chosen observations from each team (A, B, and C), totaling six observations, thereby ensuring the desired proportional allocation based on our stratification criteria.

Obs	team	points	SelectionProb	SamplingWeight
1	A	35	0.4	2.5
2	A	40	0.4	2.5
3	B	29	0.5	2.0
4	B	33	0.5	2.0
5	C	25	0.4	2.5
6	C	10	0.4	2.5

Example 3: Implementing Clustered Random Sampling

Clustered random sampling is an effective and efficient approach when the population naturally exists in readily identifiable groups or clusters, and collecting data from all members within a selected group is easier or cheaper than sampling individuals across all groups. In this example, we treat the basketball teams (A, B, and C) as our clusters. Instead of selecting individual players, we randomly select entire teams to form the sample.

The crucial element for cluster sampling is the `CLUSTER` statement. By providing the grouping variable (here designated as `grouping_var`, representing the team), SAS treats each unique value of this variable as a single sampling unit. We set `N=2`, instructing the procedure to randomly select two of the three available clusters. Clustered sampling differs from stratification because, typically, all observations belonging to the selected clusters are automatically included in the final sample.

The syntax below executes the cluster selection. The output will reflect that only the observations

from the chosen clusters are included in `my_sample`.

```
proc surveyselect data=my_data
out=my_sample
n=2 /*select a total of 2 clusters*/
seed=1; /*set seed to make this example reproducible*/
cluster grouping_var; /*specify variable to use for clustering*/
run;

/*view sample*/
proc print data=my_sample;
```

After execution with the fixed seed, Teams A and B were selected as the two clusters. The final sample dataset, shown below, contains every single observation originally associated with those two teams. This method resulted in a total sample size of nine observations (five from Team A and four from Team B).

Obs	team	points
1	A	12
2	A	14
3	A	22
4	A	35
5	A	40
6	B	12
7	B	10
8	B	29
9	B	33

Conclusion and Further Resources

Mastering PROC SURVEYSELECT is essential for anyone conducting rigorous statistical analysis in SAS. The procedure offers unparalleled flexibility, enabling researchers to move beyond basic simple random sampling to employ complex, multi-stage designs necessary for large-scale studies. Whether the goal is maximizing statistical efficiency through stratified sampling or reducing logistical costs through clustered random sampling, PROC SURVEYSELECT provides the precise tools needed to generate representative samples for statistical inference.

A critical takeaway is the importance of the `SEED` statement for reproducibility. Furthermore, the output dataset generated by PROC SURVEYSELECT includes crucial variables that document the sampling design, such as sampling probabilities and weights. These variables are not merely metadata; they are required inputs for subsequent specialized survey analysis procedures in SAS (e.g., PROC SURVEYMEANS, PROC SURVEYFREQ) to ensure that variance estimates and confidence intervals are calculated correctly, accounting for the complexity introduced by the sampling method.

For advanced applications, such as unequal probability sampling or systematic sampling, the official documentation provides detailed guidance on utilizing additional statements like `SIZE`, `SAMPSIZE`, and `RATE`.

Note: You can find the complete documentation for **PROC SURVEYSELECT** .

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