**Creating Standardized Scores in SAS**

In this document I explain how to obtain standardized scores using SAS syntax.

The data for this example are from the study described in the article “Self-efficacy beliefs in college statistics courses[[1]](#footnote-1).” The variable used for this example is the Value subscale from the Survey of Attitudes toward Statistics (named “value” in the dataset). This subscale assesses the degree to which respondents think statistics is valuable. The data are in the SAS dataset “sats.sas7bdat.” There are no missing data.

**Creating *z-* and T*-* Scores**

***Creating standardized scores using Proc Standard***

**Proc standard** can be used to obtain both *z*- and T-scores. However, it cannot create both scores within the same command, so the proc would have to be run twice. Also, the two scores would be put into different datasets. For our example, the syntax would be:

**/\* syntax for z-score\*/**

**proc standard data=**SATS **mean=**0 **std=**1

**out =** new**;**

**run;**

**/\*syntax for t-score\*/**

**proc standard data =** SATS **mean =** 50 **sd =** 10

**out =** new2**;**

**run;**

In the syntax above, the keywords **mean =** and **sd =** specify the mean and standard deviation for the new scores. In the first set of commands, these are 0 and 1, respectively, as these are the mean and standard deviation of *z*-scores.

The second set of commands provides the mean and standard deviation for T*-*scores (50 and 10, respectively).

The working dataset name is “SATS.” The dataset “new” will contain the *z*-scores and dataset “new2” will contain the T-scores.

**Creating Stanine Scores**

Recall that stanines are scores that range from 1 to 9. The percentages of scores within stanines 1- 9 are: 4, 7, 12, 17, 20, 17, 12, 7, 4. This means that stanines of 1 - 9 correspond to the raw scores that have the cumulative percentages of 4, 11, 23, 40, 60, 77, 89, 96. To find the scores that correspond to these cumulative percentages, use the **proc freq** to obtain a frequency distribution table.

**proc freq data =** SATS**;**

**tables** value**;**

**run;**

You will get the frequency distribution table below. With a small number of respondents such as in this example, there will not be an exact match for each of the cumulative percentages needed (4, 11, 23, 40, 60, 77, 89, 96). I have indicated the closest matches with red arrows in the table.

| **value** | **Frequency** | **Percent** | **Cumulative Frequency** | **Cumulative Percent** |
| --- | --- | --- | --- | --- |
| **15** | 1 | 0.97 | 1 | 0.97 |
| **21** | 1 | 0.97 | 2 | 1.94 |
| **23** | 1 | 0.97 | 3 | 2.91 |
| **28** | 2 | 1.94 | 5 | 4.85 |
| **33** | 6 | 5.83 | 11 | 10.68 |
| **35** | 2 | 1.94 | 13 | 12.62 |
| **36** | 4 | 3.88 | 17 | 16.50 |
| **37** | 3 | 2.91 | 20 | 19.42 |
| **38** | 2 | 1.94 | 22 | 21.36 |
| **39** | 5 | 4.85 | 27 | 26.21 |
| **40** | 4 | 3.88 | 31 | 30.10 |
| **41** | 5 | 4.85 | 36 | 34.95 |
| **42** | 8 | 7.77 | 44 | 42.72 |
| **43** | 6 | 5.83 | 50 | 48.54 |
| **44** | 4 | 3.88 | 54 | 52.43 |
| **45** | 9 | 8.74 | 63 | 61.17 |
| **46** | 6 | 5.83 | 69 | 66.99 |
| **47** | 2 | 1.94 | 71 | 68.93 |
| **48** | 4 | 3.88 | 75 | 72.82 |
| **49** | 3 | 2.91 | 78 | 75.73 |
| **50** | 5 | 4.85 | 83 | 80.58 |
| **51** | 2 | 1.94 | 85 | 82.52 |
| **52** | 5 | 4.85 | 90 | 87.38 |
| **53** | 4 | 3.88 | 94 | 91.26 |
| **54** | 2 | 1.94 | 96 | 93.20 |
| **55** | 2 | 1.94 | 98 | 95.15 |
| **59** | 1 | 0.97 | 99 | 96.12 |
| **60** | 1 | 0.97 | 100 | 97.09 |
| **61** | 3 | 2.91 | 103 | 100.00 |

The scores that most closely match the cumulative percentages of 4, 11, 23, 40, 60, 77, 89, 96 are 28, 33, 38, 42, 45, 49, 52, and 59.

The stanines are therefore:

* Lowest score through 28 = 1
* 29 - 33 = 2
* 34 - 38 = 3
* 39 – 42 = 4
* 43 – 45 = 5
* 46 – 49 = 6
* 50 – 52 = 7
* 53 – 59 = 8
* 60 through highest score = 9

Stanines can be obtained in a SAS data step using the **if, else if,** and **then** commands, as shown in the syntax below.

**data** stanine**; set** SATS**;**

**if** value **le** 28 **then** stanine **=** 1**;**

**else if** 29 **le** value **le** 33 **then** stanine **=** 2**;**

**else if** 34 **le** value **le** 38 **then** stanine **=** 3**;**

**else if** 39 **le** value **le** 42 **then** stanine **=** 4**;**

**else if** 43 **le** value **le** 45 **then** stanine **=** 5**;**

**else if** 46 **le** value **le** 49 **then** stanine **=** 6**;**

**else if** 50 **le** value **le** 52 **then** stanine **=** 7**;**

**else if** 53 **le** value **le** 59 **then** stanine **=** 8**;**

**else if** value **ge** 60 **then** stanine **=** 9**;**

**run;**

Stanines corresponding to each score of “value” will be saved into the dataset “stanine,” along with the original variable “value.”

**Creating standardized scores using SAS SQL**

***Creating z- and* T*-scores with SAS SQL***

*Z*- and T-scores can be obtained using **SQL** (Structured Query Language) in SAS. Although perhaps not as familiar to many SAS users as other procs, **Proc SQL** has the advantage of flexibility.

In **SQL**, data are stored in *tables*. The **select** subcommand indicates the variables, or columns, that should be selected from the working dataset. Specifying “\*” means that all columns, or variables, should be selected. The **create table** subcommand in the **proc sql** command below specifies that a new table (dataset) called “new” should be created that contains all the columns (variables) in the working dataset.

The next two lines create the new variables “zscore” and “tscore” from the variable “value.”

The new variable “zscore” equals the variables “value” minus its mean divided by its standard deviation. The new variable “tscore” begins with the formula for “zscore,” multiplies this by 10, and adds 50 to create the *T*-score.

The subcommand *from SATS* specifies the working dataset.

Finally, note that SQL commands use the specification *quit* instead of *run* at the end.

The commands below will result in the dataset “new” that will contain three variables: “value,” “zscore,” and “tscore.”

Finally, note that **proc sql** ends with the command **quit.**

**proc sql;**

**create table** new **as select \*,**

**(**value-**mean**(value))/**std**(value) **as** zscore,

((value-**mean**(value))/**std**(value))\*10 +50 **as** tscore

**from** SATS;

**quit;**

***Creating stanines with Proc SQL***

The **SQL** commands for *z*- and T*­*-scores can be modified to also calculate stanines, as shown below.

The **SQL** subcommand **case** is needed to perform **if-then-else** operations. The subcommand **when** serves the function of **if.**

The subcommand **as** stanine specifies the name for the newly created variable stanine.

**proc sql;**

**create table** new **as select \*,**

**(**value**-mean(**value**))/std(**value**) as** zscore**,**

**((**value**-mean(**value**))/std(**value**))\***10 **+**50 **as** tscore**,**

**case**

**when** value **le** 28 **then** 1

**when** 29 **le** value **le** 33 **then** 2

**when** 34 **le** value **le** 38 **then** 3

**when** 39 **le** value **le** 42 **then** 4

**when** 43 **le** value **le** 45 **then** 5

**when** 46 **le** value **le** 49 **then** 6

**when** 50 **le** value **le** 52 **then** 7

**when 53 le** value **le** 59 **then** 8

**when** value **ge** 60 **then** 9

**else** 0

**end**

**as** stanine

**from** SATS**;**

**quit;**

**Obtaining Percentile Points**

Percentile points that correspond to any percentile rank can be obtained using **proc univariate.** The commands below specify that percentile points corresponding to the ten deciles (percentile ranks of 10, 20, 30, etc.) should be saved into a new dataset named “Pctls.” The subcommand **noprint** suppresses printing of the usual descriptive statistics.

The **pctlpre** and **pctlname** subcommands create names for the new percentile point variables. **Pcetlpre**creates the prefix for the name and**pctlname**creates the suffix. For example, the first percentile point will be named “Valuepct10.”

**proc univariate data=**SATS **noprint;**

**var** value**;**

**output out=**Pctls **pctlpts =** 10 20 30 40 50 60 70 80 90

**pctlpre =** Value

**pctlname =** pct10 pct20 pct30 pct40 pct50 pct60 pct70 pct80 pct90**;**

**run;**

1. Finney, S.J., & Schraw (2003). *Contemporary Educational Psychology*, *28,* 161–186 [↑](#footnote-ref-1)