

expandcl — Duplicate clustered observations[Description](#)
[Options](#)[Quick start](#)
[Remarks and examples](#)[Menu](#)
[Also see](#)[Syntax](#)

Description

`expandcl` duplicates clusters of observations and generates a new variable that identifies the clusters uniquely.

`expandcl` replaces each cluster in the dataset with n copies of the cluster, where n is equal to the required expression rounded to the nearest integer. The expression is required to be constant within cluster. If the expression is less than 1 or equal to *missing*, it is interpreted as if it were 1, and the cluster is retained but not duplicated.

Quick start

Duplicate each set of observations on clusters identified by `cvar` 3 times, and store new cluster identifier in `newcv`

```
expandcl 3, cluster(cvar) generate(newcv)
```

Duplicate each cluster of observations the number of times stored in `v`

```
expandcl v, cluster(cvar) generate(newcv)
```

Menu

Data > Create or change data > Other variable-transformation commands > Duplicate clustered observations

Syntax

```
expandcl [=] exp [if] [in], cluster(varlist) generate(newvar)
```

Options

`cluster(varlist)` is required and specifies the variables that identify the clusters before expanding the data.

`generate(newvar)` is required and stores unique identifiers for the duplicated clusters in *newvar*. *newvar* will identify the clusters by using consecutive integers starting from 1.

Remarks and examples

[stata.com](https://www.stata.com)

► Example 1

We will show how `expandcl` works by using a small dataset with five clusters. In this dataset, `cl` identifies the clusters, `x` contains a unique value for each observation, and `n` identifies how many copies we want of each cluster.

```
. use https://www.stata-press.com/data/r18/expclxmpl
. list, sepby(cl)
```

	cl	x	n
1.	10	1	-1
2.	10	2	-1
3.	20	3	0
4.	20	4	0
5.	30	5	1
6.	30	6	1
7.	40	7	2.7
8.	40	8	2.7
9.	50	9	3
10.	50	10	3
11.	60	11	.
12.	60	12	.

```
. expandcl n, generate(newcl) cluster(cl)
(2 missing counts ignored; observations not deleted)
(2 noninteger counts rounded to integer)
(2 negative counts ignored; observations not deleted)
(2 zero counts ignored; observations not deleted)
(8 observations created)
. sort newcl cl x
```

```
. list, sepby(newcl)
```

	cl	x	n	newcl
1.	10	1	-1	1
2.	10	2	-1	1
3.	20	3	0	2
4.	20	4	0	2
5.	30	5	1	3
6.	30	6	1	3
7.	40	7	2.7	4
8.	40	8	2.7	4
9.	40	7	2.7	5
10.	40	8	2.7	5
11.	40	7	2.7	6
12.	40	8	2.7	6
13.	50	9	3	7
14.	50	10	3	7
15.	50	9	3	8
16.	50	10	3	8
17.	50	9	3	9
18.	50	10	3	9
19.	60	11	.	10
20.	60	12	.	10

The first three clusters were not replicated because `n` was less than or equal to 1. `n` is 2.7 in the fourth cluster, so `expandcl` created two replications (2.7 was rounded to 3) of this cluster, bringing the total number of clusters of this type to 3. `expandcl` created two replications of cluster 50 because `n` is 3. Finally, `expandcl` did not replicate the last cluster because `n` was missing.

◀

Also see

[D] [expand](#) — Duplicate observations

[R] [bsample](#) — Sampling with replacement

Stata, Stata Press, and Mata are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2023 StataCorp LLC, College Station, TX, USA. All rights reserved.

