The Stata Journal

The *Stata Journal*, electronic version (ISSN 1536-8734) is a publication of Stata Press, and Stata is a registered trademark of StataCorp LP.

The Stata Journal (2006)
6, Number 3, pp. 430–432

Stata tip 36: Which observations?

Nicholas J. Cox
Department of Geography
Durham University
Durham City, UK
n.j.cox@durham.ac.uk

A common question is how to identify which observations satisfy some specified condition. The easiest answer is often to use `list`, as in

```
. use http://www.stata-press.com/data/r9/auto, clear
(1978 Automobile Data)
. list rep78 if rep78 == 3
  (output omitted)
```

An equivalent is to use `edit` instead. In either case, the basic ingredients to an answer are

1. At least an `if` condition and possibly an `in` condition, too. Even if we start out interested in all observations, the condition of interest will be specified using `if`.

2. The observation numbers themselves. Evidently some commands will show them (`list` and `edit` being examples), but otherwise we will need to work a little harder and do something like

   ```
   . gen long obsno = _n
   ```

   and work with that new variable. Here I spelled out that the variable type to be used is a `long`. Consulting the help for data types shows that an `int` will work for datasets with up to 32,740 observations. The default for a new variable is `float`: this will often be fine, but it is dangerous for very large datasets because not every large integer less than Stata's maximum dataset size can be held exactly.

What other complications will we need to worry about when specifying conditions?

- Precision problems with noninteger values, prominently documented but nevertheless a frequent source of minor grief (e.g., see Cox [2006] and references therein).
- Ties; i.e., more than one observation may satisfy a specified condition.
- Conditions involving string comparisons as well as numeric comparisons.

`list` or `edit` shows us the observation numbers for a particular condition, but not compactly or retrievably. We do not want to have to type out those numbers if we need them for some other purpose. To get a more compact display, one approach uses `levelsof` after generating an observation number variable.


```
. levelsof obsno if rep78 == 3
1 2 4 6 8 9 10 11 13 14 16 19 25 26 27 28 31 32 34 36 37 39 41 42 44 49 50 54 60 65
```

In an (updated) Stata 8, use `levels`, not `levelsof`. The help for `levelsof` shows that you can put the list of observation numbers into a local macro for further manipulation and that this list is accessible immediately after issuing the command as `r(levels)`.

If you want the `obsno` variable for this kind of purpose, you might want it shortly for something similar, so it might as well be left in memory as long as there is plenty to spare. But `obsno` will remain identical in contents to `_n` only as long as the sort order is not changed.

```
. assert obsno == _n
```

is a good way to check whether that remains true. `assert` gives no output if the assertion made is true for every observation, no news thus being good news in this example. See also Gould (2003).

Asking for the levels of an observation number variable works when ties are present and when string comparisons are specified. You can also add whatever other `if` or `in` conditions apply.

The main problem to worry about in practice is the precision problem. Consider

```
. summarize gear
```

Variable	Obs	Mean	Std. Dev.	Min	Max
gear_ratio	74	3.014865	.4562871	2.19	3.89

What if we want to see which observations are equal to the maximum?

```
. levelsof obsno if gear == 3.89
```

shows nothing and so fails to find the observation(s), whereas

```
. levelsof obsno if gear == float(3.89)
56
```

happens to give the right answer, but you will not always be so lucky. In other circumstances, what you see (3.89) might be more rounded than it should be. The best approach in general is to use the saved results produced by commands such as those, which are documented in the manual entry for each command. Thus after `summarize`,

```
. levelsof obsno if gear == r(max)
56
```

gives the right answer, as it does in this example,

```
. levelsof obsno if gear == `r(max)'
56
```

Nevertheless, I recommend using `r(max)` rather than `` `r(max)' `` because the former gives you access to the maximum precision possible. A similar comment applies to e-class results.

Incidentally, because `levelsof` is r-class it will overwrite the r-class results left behind by `summarize`, so you will need to issue such commands in the right order. Thus if we wanted to see both the maximums and the minimums, we would need to repeat commands. As a variation, we use the `meanonly` option, which despite its name does leave the maximum and minimum in memory.

```
. summarize gear, meanonly
. levelsof obsno if x == r(max)
56
. summarize x, meanonly
. levelsof obsno if x == r(min)
12
```

References

Cox, N. J. 2006. Stata tip 33: Sweet sixteen: Hexadecimal formats and precision problems. *Stata Journal* 6: 282–283.

Gould, W. 2003. Stata tip 3: How to be assertive. *Stata Journal* 3: 448.