The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go “beyond the Stata manual” in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

For more information on the *Stata Journal*, including information for authors, see the web page

http://www.stata-journal.com

The *Stata Journal* is indexed and abstracted in the following:

- Science Citation Index Expanded (also known as SciSearch®)
- CompuMath Citation Index®

**Copyright Statement:** The *Stata Journal* and the contents of the supporting files (programs, datasets, and help files) are copyright © by StataCorp LP. The contents of the supporting files (programs, datasets, and help files) may be copied or reproduced by any means whatsoever, in whole or in part, as long as any copy or reproduction includes attribution to both (1) the author and (2) the *Stata Journal*.

The articles appearing in the *Stata Journal* may be copied or reproduced as printed copies, in whole or in part, as long as any copy or reproduction includes attribution to both (1) the author and (2) the *Stata Journal*.

Written permission must be obtained from StataCorp if you wish to make electronic copies of the insertions. This precludes placing electronic copies of the *Stata Journal*, in whole or in part, on publicly accessible web sites, file servers, or other locations where the copy may be accessed by anyone other than the subscriber.

Users of any of the software, ideas, data, or other materials published in the *Stata Journal* or the supporting files understand that such use is made without warranty of any kind, by either the *Stata Journal*, the author, or StataCorp. In particular, there is no warranty of fitness of purpose or merchantability, nor for special, incidental, or consequential damages such as loss of profits. The purpose of the *Stata Journal* is to promote free communication among Stata users.

The *Stata Journal*, electronic version (ISSN 1536-8734) is a publication of Stata Press. Stata and Mata are registered trademarks of StataCorp LP.
**Stata tip 66: ds—A hidden gem**

Martin Weiss  
University of Tuebingen  
Tuebingen, Germany  
martin.weiss@uni-tuebingen.de  

`ds` is one of a few dozen “undocumented” commands in Stata whose names are available in `help undocumented`. Contrary to the help file assertion that “an undocumented command is a command of very limited interest, usually only to Stata programmers”, `ds` is extremely helpful, both interactively and in programs. The main hurdle to its widespread adoption by Stata users seems to be limited awareness of its existence.

Stata users are generally familiar with the `describe` command. `describe` allows you to gain a rapid overview of the dataset in memory, or with the `using` modifier, a dataset residing on a hard disk or available on the Internet. `ds` also allows that with its `detail` option, omitting only the general information provided in the header of the output of the `describe` command. For example,

```
. sysuse uslifeexp2.dta
(U.S. life expectancy, 1900-1940)
. describe
```

```
Contains data from C:\Program Files\Stata10\ado\base\u\uslifeexp2.dta
obs: 41 U.S. life expectancy, 1900-1940
vars: 2 2 Apr 2007 14:39
size: 574 (99.9% of memory free) (_dta has notes)

storage  display       value          variable name
variable name type format label variable label

year       int %9.0g label Year
le          float %9.0g label life expectancy
```

Sorted by: year

```
. ds, detail
```

```
variable name
storage type display format value label

year       int %9.0g label Year
le          float %9.0g label life expectancy
```

More importantly, `ds` also provides the means of identifying subsets of variables with specified properties. It complements `lookfor` (see [D] `lookfor`), which allows you to search for certain strings in variable names or variable labels. `ds` enhances this functionality extensively, letting you specify

- certain types of variables;
- whether variable labels, value labels, and characteristics have been attached, and if so, whether they match certain patterns; and
- variables with specific formats.
A further systematic feature is that you can specify either the subset of variables satisfying particular properties or the complementary subset that does not satisfy those properties. As a simple example of the latter, when you are using the *auto.dta*, the command *ds make, not* specifies all variables other than *make*.

These capabilities prove particularly handy with large or poorly known datasets. As a simple example, pretend you were not familiar with the *auto* dataset and were looking for string variables.

```stata
.sysuse auto.dta
(1978 Automobile Data)
.* show all variables featuring type string
  . ds, has(type string)
  make
```

While *describe* would list the variable types, leaving the task of finding a certain type to you, *ds* can provide precisely what you were looking for. Despite its “un-documented” status, a dialog box can ease navigation through the intricacies of this command: to try the dialog box, type *db ds*. Beyond the results shown as usual, *ds* also leaves behind a list of variables found in *r(varlist)*, which is available for use by subsequent commands, such as *list* (see [D] list) or *summarize* (see [R] summarize). Many of the properties that you can search for with *ds* can also be extracted with extended macro functions; see [P] macro. To illustrate, consider the *voter* dataset shipped with Stata.

```stata
.sysuse voter.dta
.* show all variables with value labels attached
  . ds, has(vall)
  candidat inc
.* show all variables not of type float
  . ds, not(type float)
  candidat inc pfrac pop
.* show mean of all variables with format %10.0g
  . ds, has(format %10.0g)
  pfrac pop
  . tabstat `r(varlist)`
  stats | pfrac pop
  ------+----------------
  mean | 6.733333 104299.3
```

The help file accessed by typing *help ds* gives several more examples. The help file accessed by typing *help varfind.pat.examp* explains the use of wildcards within the specification of patterns.