ds — Compactly list variables with specified properties

Description Quick start Menu Syntax Options
Remarks and examples Stored results Acknowledgments Also see

Description

`ds` lists variable names of the dataset currently in memory in a compact or detailed format, and lets you specify subsets of variables to be listed, either by name or by properties (for example, the variables are numeric). In addition, `ds` leaves behind in `r(varlist)` the names of variables selected so that you can use them in a subsequent command.

`ds`, typed without arguments, lists all variable names of the dataset currently in memory in a compact form.

Quick start

List variables in alphabetical order

```
ds, alpha
```

List all string variables

```
ds, has(type string)
```

List all numeric variables

```
ds, has(type numeric)
```

Same as above, but exclude date-formatted variables

```
ds, not(format %td* type string)
```

List all variables whose label includes the phrase "my text" regardless of case

```
ds, has(varlabel "*my text*") insensitive
```

Menu

Data > Describe data > Compactly list variable names

Syntax

Simple syntax

```
ds [ , alpha ]
```

Advanced syntax

```
ds [varlist] [ , options ]
```

options	Description
Main	
`not`	list variables not specified in *varlist*
`alpha`	list variables in alphabetical order
`detail`	display additional details
`varwidth(#)`	display width for variable names; default is `varwidth(12)`
`skip(#)`	gap between variables; default is `skip(2)`
Advanced	
`has(`*spec*`)`	describe subset that matches *spec*
`not(`*spec*`)`	describe subset that does not match *spec*
`insensitive`	perform case-insensitive pattern matching
`indent(#)`	indent output; seldom used

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`insensitive` and `indent(#)` are not shown in the dialog box.

spec	Description
`type` *typelist*	specified types
`format` *patternlist*	display format matching *patternlist*
`varlabel` [*patternlist*]	variable label or variable label matching *patternlist*
`char` [*patternlist*]	characteristic or characteristic matching *patternlist*
`vallabel` [*patternlist*]	value label or value label matching *patternlist*
`linkname` *namelist*	link name matching *namelist*

typelist used in has(type *typelist*) and not(type *typelist*) is a list of one or more types, each of which may be alias, unknown, numeric, string, str#, strL, byte, int, long, float, or double or may be a *numlist* such as 1/8 to mean "str1 str2 ... str8". Examples include

has(type alias)	was created by fralias add; see [D] **fralias**
has(type unknown)	is type alias, but the link is broken
has(type int)	is of type int
has(type byte int long)	is of integer type
not(type int)	is not of type int
not(type byte int long)	is not of the integer types
has(type numeric)	is a numeric variable
not(type string)	is not a string (str# or strL) variable (same as above)
has(type 1/40)	is str1, str2, ..., str40
has(type str#)	is str1, str2, ..., str2045 but not strL
has(type strL)	is of type strL but not str#
has(type numeric 1/2)	is numeric or str1 or str2

patternlist used in, for instance, has(format *patternlist*), is a list of one or more *patterns*. A pattern is the expected text with the addition of the characters * and ?. * indicates 0 or more characters go here, and ? indicates exactly 1 character goes here. Examples include

has(format *f)	format is %#.#f
has(format %t*)	has time or date format
has(format %-*s)	is a left-justified string
has(varl *weight*)	variable label includes word weight
has(varl *weight* *Weight*)	variable label has weight or Weight

To match a phrase, enclose the phrase in quotes.

has(varl "*some phrase*")	variable label has some phrase

If instead you used has(varl *some phrase*), then only variables having labels ending in some or starting with phrase would be listed.

namelist used in, for instance, has(linkname *namelist*) is a list of one or more names. linkname refers to the linkage variables used to create alias variables; see [D] **fralias**. Abbreviations in *namelist* are not supported.

Options

Main

not specifies that the variables in *varlist* not be listed. For instance, ds pop*, not specifies that all variables not starting with the letters pop be listed. The default is to list all the variables in the dataset or, if *varlist* is specified, the variables specified.

alpha specifies that the variables be listed in alphabetical order. If the variable contains Unicode characters other than plain ASCII, the sort order is determined strictly by the underlying byte order. See **[U] 12.4.2.5 Sorting strings containing Unicode characters**.

detail specifies that detailed output identical to that of describe be produced. If detail is specified, varwidth(), skip(), and indent() are ignored.

varwidth(#) specifies the display width of the variable names; the default is varwidth(12).

skip(#) specifies the number of spaces between variable names, where all variable names are assumed to be the length of the longest variable name; the default is skip(2).

Advanced

has(*spec*) and not(*spec*) select from the dataset (or from *varlist*) the subset of variables that meet or fail the specification *spec*. Selection may be made on the basis of storage type, variable label, value label, display format, or characteristics. Only one not, has(), or not() option may be specified.

has(type string) selects all string variables. Typing ds, has(type string) would list all string variables in the dataset, and typing ds pop*, has(type string) would list all string variables whose names begin with the letters pop.

has(format *patternlist*) specifies variables whose format matches any of the patterns in *patternlist*. has(format *f) would select all variables with formats ending in f, which presumably would be all %#.#f, %0#.#f, and %-#.#f formats. has(format *f *fc) would select all variables with formats ending in f or fc. not(format %t* %-t*) would select all variables except those with date or time-series formats.

has(varlabel) selects variables with defined variable labels. has(varlabel *weight*) selects variables with variable labels including the word "weight". not(varlabel) would select all variables with no variable labels.

has(char) selects all variables with defined characteristics. has(char problem) selects all variables with a characteristic named problem.

has(vallabel) selects variables with defined value labels. has(vallabel yesno) selects variables whose value label is yesno. has(vallabel *no) selects variables whose value label ends in the letters no.

has(linkname) selects variables to create alias variables; see [D] **fralias**.

The following options are available with ds but are not shown in the dialog box:

insensitive specifies that the matching of the *pattern* in has() and not() be case insensitive. Note that the case insensitivity applies only to ASCII characters.

indent(#) specifies the amount the lines are indented.

Remarks and examples

If ds is typed without any operands, then a compact list of the variable names for the data currently in memory is displayed.

▷ Example 1

ds can be especially useful if you have a dataset with over 1,000 variables, but you may find it convenient even if you have considerably fewer variables.

```
. use https://www.stata-press.com/data/r19/educ3
(ccdb46, 52-54)
. ds
fips       popcol    medhhinc  tlf       emp       clfbls    z
crimes     perhspls  medfinc   clf       empmanuf  clfuebls  adjinc
pcrimes    perclpls  state     clffem    emptrade  famnw     perman
crimrate   prcolhs   division  clfue     empserv   fam2w     pertrade
pop25pls   medage    region    empgovt   osigind   famwsamp  perserv
pophspls   perwhite  dc        empself   osigindp  pop18pls  perother
```

Example 2

You might wonder why you would ever specify *varlist* with this command. Remember that *varlist* understands the '`*`' abbreviation character and the '`-`' dash notation; see **[U] 11.4 varname and varlists**.

```
. ds p*
pcrimes   pophspls  perhspls  prcolhs   pop18pls  pertrade  perother
pop25pls  popcol    perclpls  perwhite  perman    perserv
. ds popcol-clfue
popcol    perclpls  medage    medhhinc  state     region    tlf       clffem
perhspls  prcolhs   perwhite  medfinc   division  dc        clf       clfue
```

Example 3

Because the primary use of `ds` is to inspect the names of variables, it is sometimes useful to let `ds` display the variable names in alphabetical order.

```
. ds, alpha
adjinc     crimes    empmanuf  famwsamp  osigindp  perserv   pophspls
clf        crimrate  empself   fips      pcrimes   pertrade  prcolhs
clfbls     dc        empserv   medage    perclpls  perwhite  region
clffem     division  emptrade  medfinc   perhspls  pop18pls  state
clfue      emp       fam2w     medhhinc  perman    pop25pls  tlf
clfuebls   empgovt   famnw     osigind   perother  popcol    z
```

Stored results

`ds` stores the following in `r()`:

Macros

`r(varlist)` the varlist of found variables

Acknowledgments

`ds` was originally written by StataCorp. It was redesigned and rewritten by Nicholas J. Cox of the Department of Geography at Durham University, UK, who is coeditor of the *Stata Journal* and author of *Speaking Stata Graphics*. The purpose was to include the selection options `not`, `has()`, and `not()`; to produce better-formatted output; and to be faster. Cox thanks Richard Goldstein, William Gould, Kenneth Higbee, Jay Kaufman, Jean Marie Linhart, and Fred Wolfe for their helpful suggestions on previous versions.

Also see

[D] **cf** — Compare two datasets

[D] **codebook** — Describe data contents

[D] **compare** — Compare two variables

[D] **compress** — Compress data in memory

[D] **describe** — Describe data in memory or in a file

[D] **format** — Set variables' output format

[D] **fralias** — Alias variables from linked frames

[D] **label** — Manipulate labels

[D] **lookfor** — Search for string in variable names and labels

[D] **notes** — Place notes in data

[D] **order** — Reorder variables in dataset

[D] **rename** — Rename variable