ds — List variables matching name patterns or other characteristics

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

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
List variables matching name patterns or other characteristics

typelist used in \texttt{has(type typelist)} and \texttt{not(type typelist)} is a list of one or more types, each of which may be 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

\begin{itemize}
\item \texttt{has(type int)} is of type int
\item \texttt{has(type byte int long)} is of integer type
\item \texttt{not(type int)} is not of type int
\item \texttt{not(type byte int long)} is not of the integer types
\item \texttt{has(type numeric)} is a numeric variable
\item \texttt{not(type string)} is not a string (str# or strL) variable (same as above)
\item \texttt{has(type 1/40)} is str1, str2, ..., str40
\item \texttt{has(type str#)} is str1, str2, ..., str2045 but not strL
\item \texttt{has(type strL)} is of type strL but not str#
\item \texttt{has(type numeric 1/2)} is numeric or str1 or str2
\end{itemize}

patternlist used in, for instance, \texttt{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

\begin{itemize}
\item \texttt{has(format *f)} format is \texttt{%.#f}
\item \texttt{has(format %t*)} has time or date format
\item \texttt{has(format %-*s)} is a left-justified string
\item \texttt{has(varl *weight*)} variable label includes word weight
\item \texttt{has(varl *weight* *Weight*)} variable label has weight or Weight
\end{itemize}

To match a phrase, enclose the phrase in quotes.

\texttt{has(varl "*some phrase*")} variable label has some phrase

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

Menu

\begin{itemize}
\item Data \> Describe data \> Compactly list variable names
\end{itemize}

Description

\texttt{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, \texttt{ds} leaves behind in \texttt{r(varlist)} the names of variables selected so that you can use them in a subsequent command.

\texttt{ds}, typed without arguments, lists all variable names of the dataset currently in memory in a compact form.
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.

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)'.

`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('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('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('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 ending in f or fc. `not('format %t* %-t*)' would select all variables except those with date or time-series formats.

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

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.

`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 http://www.stata-press.com/data/r13/educ3
   (ccdb46, 52-54)
. ds
fips    popcol  medhhinc  tlf    emp    clfbls  z
   crimes  perhspls  medfinc  clf  empprunf  clfuebls  adjinc
   pcrimes  perclpls  state  clfem  emptrade  famnw  perman
   crimrate  proclhs  division  clfue  empserv  fam2w  pertrade
   pop25pls  medage  region  empgovt  osigind  famwsamp  perserv
   pophspls  perwhite  dc  empsel  osigindp  pop18pls  perother
```

Example 2

You might wonder why you would ever specify a `varlist` with this command. Remember that a `varlist` understands the `*` abbreviation character and the `-` dash notation; see [U] 11.4 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  clfem
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  empprunf  famwsamp  osigindp  perserv  pophspls
clf  crimrate  empsel  fips  pcrimes  pertrade  prcolhs
clfbls  dc  empserv  medage  perclpls  perwhite  region
clfem  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, and coeditor of the Stata Journal. 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.

References


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 file
[D] format — Set variables’ output format
[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