Title

stata.com

destring — Convert string variables to numeric variables and vice versa

Description
Quick start
Menu
Syntax
Options for destring
Options for tostring
Remarks and examples
Acknowledgment
References
Also see

Description

`destring` converts variables in *varlist* from string to numeric. If *varlist* is not specified, `destring` will attempt to convert all variables in the dataset from string to numeric. Characters listed in `ignore()` are removed. Variables in *varlist* that are already numeric will not be changed. `destring` treats both empty strings "" and "`.`" as indicating sysmiss (`.`) and interprets the strings "`.a`", "`.b`", ..., "`.z`" as the extended missing values `.a`, `.b`, ..., `.z`; see [U] **12.2.1 Missing values**. `destring` also ignores any leading or trailing spaces so that, for example, " " is equivalent to "" and " . " is equivalent to "`.`".

`tostring` converts variables in *varlist* from numeric to string. The most compact string format possible is used. Variables in *varlist* that are already string will not be converted.

Quick start

Convert `strg1` from string to numeric, and place result in `num1`

```
destring strg1, generate(num1)
```

As above, but ignore the % character in `strg1`

```
destring strg1, generate(num1) ignore(%)
```

As above, but return `.` for observations with nonnumeric characters

```
destring strg1, generate(num1) force
```

Convert `num2` from numeric to string, and place result in `strg2`

```
tostring num2, generate(strg2)
```

As above, but format with a leading zero and 3 digits after the decimal

```
tostring num2, generate(strg2) format(%09.3f)
```

Menu

destring

Data > Create or change data > Other variable-transformation commands > Convert variables from string to numeric

tostring

Data > Create or change data > Other variable-transformation commands > Convert variables from numeric to string

Syntax

Convert string variables to numeric variables

```
destring [varlist], {generate(newvarlist)|replace} [destring_options]
```

Convert numeric variables to string variables

```
tostring varlist, {generate(newvarlist)|replace} [tostring_options]
```

destring_options	Description
* `generate(`*newvarlist*`)`	generate $newvar_1$, ..., $newvar_k$ for each variable in *varlist*
* `replace`	replace string variables in *varlist* with numeric variables
`ignore("`*chars*`" [, `*ignoreopts*`])`	remove specified nonnumeric characters, as characters or as bytes, and illegal Unicode characters
`force`	convert nonnumeric strings to missing values
`float`	generate numeric variables as type `float`
`percent`	convert percent variables to fractional form
`dpcomma`	convert variables with commas as decimals to period-decimal format

* Either `generate(`*newvarlist*`)` or `replace` is required.

tostring_options	Description
* `generate(`*newvarlist*`)`	generate $newvar_1$, ..., $newvar_k$ for each variable in *varlist*
* `replace`	replace numeric variables in *varlist* with string variables
`force`	force conversion ignoring information loss
`format(`*format*`)`	convert using specified format
`usedisplayformat`	convert using display format

* Either `generate(`*newvarlist*`)` or `replace` is required.

Options for destring

Either `generate()` or `replace` must be specified. With either option, if any string variable contains nonnumeric characters not specified with `ignore()`, then no corresponding variable will be generated, nor will that variable be replaced (unless `force` is specified).

`generate(`*newvarlist*`)` specifies that a new variable be created for each variable in *varlist*. *newvarlist* must contain the same number of new variable names as there are variables in *varlist*. If *varlist* is not specified, `destring` attempts to generate a numeric variable for each variable in the dataset; *newvarlist* must then contain the same number of new variable names as there are variables in the dataset. Any variable labels or characteristics will be copied to the new variables created.

`replace` specifies that the variables in *varlist* be converted to numeric variables. If *varlist* is not specified, `destring` attempts to convert all variables from string to numeric. Any variable labels or characteristics will be retained.

`ignore("`*chars*`" [ , `*ignoreopts*` ])` specifies nonnumeric characters be removed. *ignoreopts* may be `aschars`, `asbytes`, or `illegal`. The default behavior is to remove characters as characters, which is the same as specifying `aschars`. `asbytes` specifies removal of all bytes included in all characters in the ignore string, regardless of whether these bytes form complete Unicode characters. `illegal` specifies removal of all illegal Unicode characters, which is useful for removing high-ASCII characters. `illegal` may not be specified with `asbytes`. If any string variable still contains any nonnumeric or illegal Unicode characters after the ignore string has been removed, no action will take place for that variable unless `force` is also specified. Note that to Stata the comma is a nonnumeric character; see also the `dpcomma` option below.

`force` specifies that any string values containing nonnumeric characters, in addition to any specified with `ignore()`, be treated as indicating missing numeric values.

`float` specifies that any new numeric variables be created initially as type `float`. The default is type `double`; see [D] **Data types**. `destring` attempts automatically to compress each new numeric variable after creation.

`percent` removes any percent signs found in the values of a variable, and all values of that variable are divided by 100 to convert the values to fractional form. `percent` by itself implies that the percent sign, "`%`", is an argument to `ignore()`, but the converse is not true.

`dpcomma` specifies that variables with commas as decimal values should be converted to have periods as decimal values.

Options for tostring

Either `generate()` or `replace` must be specified. If converting any numeric variable to string would result in loss of information, no variable will be produced unless `force` is specified. For more details, see `force` below.

`generate(`*newvarlist*`)` specifies that a new variable be created for each variable in *varlist*. *newvarlist* must contain the same number of new variable names as there are variables in *varlist*. Any variable labels or characteristics will be copied to the new variables created.

`replace` specifies that the variables in *varlist* be converted to string variables. Any variable labels or characteristics will be retained.

`force` specifies that conversions be forced even if they entail loss of information. Loss of information means one of two circumstances: 1) The result of `real(string(`*varname*`, "`*format*`"))` is not equal to *varname*; that is, the conversion is not reversible without loss of information; 2) `replace` was specified, but a variable has associated value labels. In circumstance 1, it is usually best to specify `usedisplayformat` or `format()`. In circumstance 2, value labels will be ignored in a forced conversion. `decode` (see [D] **encode**) is the standard way to generate a string variable based on value labels.

`format(`*format*`)` specifies that a numeric format be used as an argument to the `strofreal()` function, which controls the conversion of the numeric variable to string. For example, a format of `%7.2f` specifies that numbers are to be rounded to two decimal places before conversion to string. See *Remarks and examples* below and [FN] **String functions** and [D] **format**. `format()` cannot be specified with `usedisplayformat`.

`usedisplayformat` specifies that the current display format be used for each variable. For example, this option could be useful when using U.S. Social Security numbers or daily or other dates with some `%d` or `%t` format assigned. `usedisplayformat` cannot be specified with `format()`.

Remarks and examples

stata.com

Remarks are presented under the following headings:

destring
tostring
Saved characteristics
Video example

destring

▷ Example 1

We read in a dataset, but somehow all the variables were created as strings. The variables contain no nonnumeric characters, and we want to convert them all from string to numeric data types.

```
. use https://www.stata-press.com/data/r17/destring1
. describe
Contains data from https://www.stata-press.com/data/r17/destring1.dta
 Observations:            10
    Variables:             5                  3 Mar 2020 10:15
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
id                str3    %9s
num               str3    %9s
code              str4    %9s
total             str5    %9s
income            str5    %9s
-------------------------------------------------------------------------------
Sorted by:
. list
```

	id	num	code	total	income
1.	111	243	1234	543	23423
2.	111	123	2345	67854	12654
3.	111	234	3456	345	43658
4.	222	345	4567	57	23546
5.	333	456	5678	23	21432
6.	333	567	6789	23465	12987
7.	333	678	7890	65	9823
8.	444	789	8976	23	32980
9.	444	901	7654	23	18565
10.	555	890	6543	423	19234

```
. destring, replace
id: all characters numeric; replaced as int
num: all characters numeric; replaced as int
code: all characters numeric; replaced as int
total: all characters numeric; replaced as long
income: all characters numeric; replaced as long
```

```
. describe
Contains data from https://www.stata-press.com/data/r17/destring1.dta
 Observations:            10
    Variables:             5                  3 Mar 2020 10:15
```

Variable name	Storage type	Display format	Value label	Variable label
id	int	%10.0g		
num	int	%10.0g		
code	int	%10.0g		
total	long	%10.0g		
income	long	%10.0g		

```
Sorted by:
     Note: Dataset has changed since last saved.
. list
```

	id	num	code	total	income
1.	111	243	1234	543	23423
2.	111	123	2345	67854	12654
3.	111	234	3456	345	43658
4.	222	345	4567	57	23546
5.	333	456	5678	23	21432
6.	333	567	6789	23465	12987
7.	333	678	7890	65	9823
8.	444	789	8976	23	32980
9.	444	901	7654	23	18565
10.	555	890	6543	423	19234

◁

▷ Example 2

Our dataset contains the variable `date`, which was accidentally recorded as a string because of spaces after the year and month. We want to remove the spaces. `destring` will convert it to numeric and remove the spaces.

```
. use https://www.stata-press.com/data/r17/destring2, clear
. describe date
```

Variable name	Storage type	Display format	Value label	Variable label
date	str14	%10s		

```
. list date

      +----------+
      |     date |
      |----------|
  1.  | 1999 12 10 |
  2.  | 2000 07 08 |
  3.  | 1997 03 02 |
  4.  | 1999 09 00 |
  5.  | 1998 10 04 |
      |----------|
  6.  | 2000 03 28 |
  7.  | 2000 08 08 |
  8.  | 1997 10 20 |
  9.  | 1998 01 16 |
 10.  | 1999 11 12 |
      +----------+

. destring date, replace ignore(" ")
date: character space removed; replaced as long
. describe date

Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
date            long    %10.0g
. list date

      +----------+
      |     date |
      |----------|
  1.  | 19991210 |
  2.  | 20000708 |
  3.  | 19970302 |
  4.  | 19990900 |
  5.  | 19981004 |
      |----------|
  6.  | 20000328 |
  7.  | 20000808 |
  8.  | 19971020 |
  9.  | 19980116 |
 10.  | 19991112 |
      +----------+
```

◁

▷ Example 3

Our dataset contains the variables `date`, `price`, and `percent`. These variables were accidentally read into Stata as string variables because they contain spaces, dollar signs, commas, and percent signs. We want to remove all of these characters and create new variables for `date`, `price`, and `percent` containing numeric values. After removing the percent sign, we want to convert the `percent` variable to decimal form.

```
. use https://www.stata-press.com/data/r17/destring2, clear
. describe
Contains data from https://www.stata-press.com/data/r17/destring2.dta
 Observations:            10
    Variables:             3                  3 Mar 2020 22:50
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
date            str14   %10s
price           str11   %11s
percent         str3    %9s
-------------------------------------------------------------------------------
Sorted by:
. list
```

	date	price	percent
1.	1999 12 10	$2,343.68	34%
2.	2000 07 08	$7,233.44	86%
3.	1997 03 02	$12,442.89	12%
4.	1999 09 00	$233,325.31	6%
5.	1998 10 04	$1,549.23	76%
6.	2000 03 28	$23,517.03	35%
7.	2000 08 08	$2.43	69%
8.	1997 10 20	$9,382.47	32%
9.	1998 01 16	$289,209.32	45%
10.	1999 11 12	$8,282.49	1%

```
. destring date price percent, generate(date2 price2 percent2) ignore("$ ,%")
> percent
date: character space removed; date2 generated as long
price: characters $ , removed; price2 generated as double
percent: character % removed; percent2 generated as double
. describe
Contains data from https://www.stata-press.com/data/r17/destring2.dta
 Observations:            10
    Variables:             6                  3 Mar 2020 22:50
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
date            str14   %10s
date2           long    %10.0g
price           str11   %11s
price2          double  %10.0g
percent         str3    %9s
percent2        double  %10.0g
-------------------------------------------------------------------------------
Sorted by:
     Note: Dataset has changed since last saved.
```

```
. list
```

	date	date2	price	price2	percent	percent2
1.	1999 12 10	19991210	$2,343.68	2343.68	34%	.34
2.	2000 07 08	20000708	$7,233.44	7233.44	86%	.86
3.	1997 03 02	19970302	$12,442.89	12442.89	12%	.12
4.	1999 09 00	19990900	$233,325.31	233325.31	6%	.06
5.	1998 10 04	19981004	$1,549.23	1549.23	76%	.76
6.	2000 03 28	20000328	$23,517.03	23517.03	35%	.35
7.	2000 08 08	20000808	$2.43	2.43	69%	.69
8.	1997 10 20	19971020	$9,382.47	9382.47	32%	.32
9.	1998 01 16	19980116	$289,209.32	289209.32	45%	.45
10.	1999 11 12	19991112	$8,282.49	8282.49	1%	.01

◁

tostring

Conversion of numeric data to string equivalents can be problematic. Stata, like most software, holds numeric data to finite precision and in binary form. See the discussion in [U] **13.12 Precision and problems therein**. If no `format()` is specified, `tostring` uses the format `%12.0g`. This format is, in particular, sufficient to convert integers held as bytes, ints, or longs to string equivalent without loss of precision.

However, users will often need to specify a format themselves, especially when the numeric data have fractional parts and for some reason a conversion to string is required.

▷ Example 4

Our dataset contains a string month variable and numeric year and day variables. We want to convert the three variables to a `%td` date.

```
. use https://www.stata-press.com/data/r17/tostring, clear
. list
```

	id	month	day	year
1.	123456789	jan	10	2001
2.	123456710	mar	20	2001
3.	123456711	may	30	2001
4.	123456712	jun	9	2001
5.	123456713	oct	17	2001
6.	123456714	nov	15	2001
7.	123456715	dec	28	2001
8.	123456716	apr	29	2001
9.	123456717	mar	11	2001
10.	123456718	jul	3	2001

```
. tostring year day, replace
year was float now str4
day was float now str2
. generate date = month + "/" + day + "/" + year
. generate edate = date(date, "MDY")
. format edate %td
```

```
. list
```

	id	month	day	year	date	edate
1.	123456789	jan	10	2001	jan/10/2001	10jan2001
2.	123456710	mar	20	2001	mar/20/2001	20mar2001
3.	123456711	may	30	2001	may/30/2001	30may2001
4.	123456712	jun	9	2001	jun/9/2001	09jun2001
5.	123456713	oct	17	2001	oct/17/2001	17oct2001
6.	123456714	nov	15	2001	nov/15/2001	15nov2001
7.	123456715	dec	28	2001	dec/28/2001	28dec2001
8.	123456716	apr	29	2001	apr/29/2001	29apr2001
9.	123456717	mar	11	2001	mar/11/2001	11mar2001
10.	123456718	jul	3	2001	jul/3/2001	03jul2001

◁

Saved characteristics

Each time the `destring` or `tostring` commands are issued, an entry is made in the characteristics list of each converted variable. You can type `char list` to view these characteristics.

After example 3, we could use `char list` to find out what characters were removed by the `destring` command.

```
. char list
  date2[destring]:            Character removed was: space
  date2[destring_cmd]:        destring date price percent, generate(date2 pri..
  price2[destring]:           Characters removed were: $ ,
  price2[destring_cmd]:       destring date price percent, generate(date2 pri..
  percent2[destring]:         Character removed was: %
  percent2[destring_cmd]:     destring date price percent, generate(date2 pri..
```

Video example

How to convert a string variable to a numeric variable

Acknowledgment

`destring` and `tostring` were originally written 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*.

References

Cox, N. J. 1999a. dm45.1: Changing string variables to numeric: Update. *Stata Technical Bulletin* 49: 2. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, p. 14. College Station, TX: Stata Press.

——. 1999b. dm45.2: Changing string variables to numeric: Correction. *Stata Technical Bulletin* 52: 2. Reprinted in *Stata Technical Bulletin Reprints*, vol. 9, p. 14. College Station, TX: Stata Press.

——. 2011. Speaking Stata: MMXI and all that: Handling Roman numerals within Stata. *Stata Journal* 11: 126–142.

Cox, N. J., and W. W. Gould. 1997. dm45: Changing string variables to numeric. *Stata Technical Bulletin* 37: 4–6. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 34–37. College Station, TX: Stata Press.

Cox, N. J., and C. B. Schechter. 2018. Speaking Stata: Seven steps for vexatious string variables. *Stata Journal* 18: 981–994.

Cox, N. J., and J. B. Wernow. 2000a. dm80: Changing numeric variables to string. *Stata Technical Bulletin* 56: 8–12. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 24–28. College Station, TX: Stata Press.

——. 2000b. dm80.1: Update to changing numeric variables to string. *Stata Technical Bulletin* 57: 2. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 28–29. College Station, TX: Stata Press.

Jeanty, P. W. 2013. Dealing with identifier variables in data management and analysis. *Stata Journal* 13: 699–718.

Also see

[D] **egen** — Extensions to generate

[D] **encode** — Encode string into numeric and vice versa

[D] **generate** — Create or change contents of variable

[D] **split** — Split string variables into parts

[FN] **String functions**