Title

stata.com

icd10 — ICD-10 diagnosis codes

Description

The `icd10` commands are a suite of commands for working with ICD-10 codes from the second edition (2003) and later. To see the current version of the ICD-10 diagnosis codes and any changes that have been applied, type `icd10 query`.

`icd10 check`, `icd10 clean`, and `icd10 generate` are data management commands. `icd10 check` verifies that a variable contains defined ICD-10 diagnosis codes and provides a summary of any problems encountered. `icd10 clean` standardizes the format of the codes. `icd10 generate` can create a binary indicator variable for whether the code is in a specified set of codes, a variable containing a corresponding higher-level code, or a variable containing the description of the code.

`icd10 lookup` and `icd10 search` are interactive utilities. `icd10 lookup` displays descriptions of the diagnosis codes specified on the command line. `icd10 search` looks for relevant ICD-10 diagnosis codes from key words given on the command line.

Quick start

Determine if ICD-10 diagnosis codes in `diag1` are invalid and store reasons in `invalid`

```
icd10 check diag1, generate(invalid)
```

Standardize display of codes in `diag2` to add a period and left-align codes

```
icd10 clean diag2, dots
```

Generate `descr3` as descriptions of the diagnosis codes in `diag3`

```
icd10 generate descr3 = diag3, description
```

Create binary indicator for malignant or benign neoplasm, as indicated by an ICD-10 code beginning with C or D in `diag4`

```
icd10 generate cancer = diag4, range(C* D*)
```

Look up current descriptions for ICD-10 diagnosis codes W70 through W79

```
icd10 lookup W70/W79
```

Look up codes where the description contains the words “delivery” or “birth”

```
icd10 search delivery birth, or
```

Menu

Data > ICD codes > ICD-10

Syntax

Verify that variable contains defined codes

```
icd10 check varname [if] [in] [, checkopts]
```

Clean variable and verify formatting

```
icd10 clean varname [if] [in], {generate(newvar)|replace} [cleanopts]
```

Generate new variable from existing variable

```
icd10 generate newvar = varname [if] [in], {category|short} [check]
icd10 generate newvar = varname [if] [in], description [genopts]
icd10 generate newvar = varname [if] [in], range(codelist) [check]
```

Display code descriptions

```
icd10 lookup codelist [, version(#)]
```

Search for codes from descriptions

```
icd10 search ["]text["] [["]text["] ...] [, or version(#)]
```

Display ICD-10 version

```
icd10 query
```

codelist is one of the following or any combination thereof:

icd10code	(the particular code)
icd10code`*`	(all codes starting with)
icd10code`/`*icd10code*	(the code range)

or any combination of the above, such as `A27.0 G40* Y60/Y69.9`.

checkopts	Description
`fmtonly`	check only format of the codes
`summary`	frequency of each invalid or undefined code; may not be combined with `list`
`list`	list observations with invalid or undefined ICD-10 codes
`generate(`*newvar*`)`	create new variable marking invalid codes
`version(#)`	year to check codes against; default is `version(2016)`

cleanopts	Description
`check`	check that variable contains ICD-10 codes before cleaning
`dots`	format codes with a period
`pad`	add space to the right of three-character codes

genopts	Description
`addcode(begin | end)`	add code to the beginning or end of the description
`pad`	add spaces to the right of the code; must specify `addcode(begin)`
`nodots`	format codes without a period; must specify `addcode()`
`check`	check that variable contains ICD-10 codes before generating new variable
`version(#)`	select description from year #; default is `version(2016)`

Options

Options are presented under the following headings:

Warning: The option descriptions are brief and use jargon. Please read *Introduction to ICD coding* in [D] **icd** before using the `icd10` command.

Options for icd10 check

`fmtonly` tells `icd10 check` to verify that the codes fit the format of ICD-10 codes but not to check whether the codes are defined.

`summary` specifies that `icd10 check` should report the frequency of each invalid or undefined code that was found in the data. To see a list of observations with invalid or undefined codes instead of the frequency for each code, use the `list` option. `summary` may not be combined with `list`.

`list` specifies that `icd10 check` list the observation number, the invalid or undefined ICD-10 code, and the reason the code is invalid or if it is undefined code. To see the frequency with which each code occurs rather than a list of observations, use the `summary` option. `list` may not be specified with `summary`.

`generate(`*newvar*`)` specifies that `icd10 check` create a new variable containing, for each observation, 0 if the observation contains a defined code or is missing. Otherwise, it contains a number from 1 to 8 if the code is invalid or 99 if the code is undefined. The positive numbers indicate the kind of problem and correspond to the listing produced by `icd10 check`. The values are labeled with the Stata-defined value label `__icd_10`.

`version(#)` specifies the version of the codes that `icd10 check` should reference when determining whether a code is defined. # may be any value between 2003, which is the second edition of ICD-10 without any updates applied, and 2016, which is the fifth edition of ICD-10. The appropriate value of # should be determined from the data source. The default is `version(2016)`.

Warning: The default value of `version()` will change over time so that the most recent codes are used. Using the default value rather than specifying a specific version may change results after a new version of the codes is introduced.

Options for icd10 clean

`generate(`*newvar*`)` specifies that a new variable should be created that will contain the formatted version of the codes in *varname*. You must specify either `generate()` or `replace`, and the two options may not be specified together.

`replace` specifies that the existing values of *varname* be replaced with the formatted values. You must specify either `generate()` or `replace`, and the two options may not be specified together.

`check` specifies that `icd10 clean` should first check that *varname* contains codes that fit the format of ICD-10 codes. Specifying the `check` option will slow down the `clean` subcommand.

`dots` specifies that the period be included in the final format. If `dots` is not specified, then all periods are removed.

`pad` specifies that spaces be added to the end of the codes to make the (implied) dots align vertically in listings. Specifying `pad` makes the resulting codes look better when used with most other Stata commands. The default is to left-align codes without adding spaces.

Options for icd10 generate

`category`, `short`, `description`, and `range(`*codelist*`)` specify the contents of the new variable that `icd10 generate` is to create. You do not need to `icd10 clean` *varname* before using `icd10 generate`; it will accept any ICD-10 format or combination of formats.

`category` and `short` generate a new variable that also contains ICD-10 codes. The resulting variable may be used with the other `icd10` subcommands.

`category` specifies to extract the three-character category code from the ICD-10 code.

`short` is designed for users who have data with greater specificity than the standard four-character ICD-10 codes. `short` will reduce five- and six-character codes to four characters. Three- and four-character codes are left as they are.

`description` creates *newvar* containing descriptions of the ICD-10 codes.

`range(`*codelist*`)` creates a new indicator variable equal to 1 when the ICD-10 code is in the range specified and equal to 0 otherwise.

`addcode(begin | end)`, `pad`, `nodots`, `check`, and `version(#)` are additional options that are only allowed if `description` is also specified.

`addcode()` specifies that the code should be included with the text describing the code. Specifying `addcode(begin)` will prepend the code to the text. Specifying `addcode(end)` will append the code to the text.

`pad` specifies that the code that is to be added to the description should be padded spaces to the right of the code so that the start of description text is aligned for all codes. `pad` may only be specified with `addcode(begin)`.

`nodots` specifies that the code that is added to the description should be formatted without a period. `nodots` may only be specified if `addcode()` is also specified.

`check` specifies that `icd10 generate` should first check that *varname* contains codes that fit the format of ICD-10 codes. Specifying the `check` option will slow down the `generate` subcommand.

`version(#)` specifies the version of the codes that `icd10 generate` should reference when selecting the description of a code. # may be any value between 2003, which is the second edition of ICD-10 without any updates applied, and 2016, which is the fifth edition of ICD-10. The default is `version(2016)`.

Warning: The default value of `version()` will change over time so that the most recent codes are used. Using the default value rather than specifying a specific version may change results after a new version of the codes is introduced.

Option for icd10 lookup

`version(#)` specifies the version of the codes that `icd10 lookup` should reference when searching for the description of the specified code. # may be any value between 2003, which is the second edition of ICD-10 without any updates applied, and 2016, which is the fifth edition of ICD-10. The default is `version(2016)`.

Warning: The default value of `version()` will change over time so that the most recent codes are used. Using the default value rather than specifying a specific version may change results after a new version of the codes is introduced.

Options for icd10 search

`or` specifies that ICD-10 codes be searched for descriptions that contain any word specified with `icd10 search`. The default is to list only descriptions that contain all the words specified.

`version(#)` specifies the version of the codes that `icd10 search` should reference when searching the text of the specified code or codes. # may be any value between 2003, which is the second edition of ICD-10 without any updates applied, and 2016, which is the fifth edition of ICD-10.

By default, descriptions for all versions are searched, meaning that codes that changed descriptions and that have descriptions in multiple versions that contain the search terms will be duplicated. To ensure a list of unique code values, specify the version number.

Remarks and examples

stata.com

Remarks are presented under the following headings:

If you have not yet read *Introduction to ICD coding* in [D] **icd** before using the `icd10` command.

Using icd10

The general format of an ICD-10 diagnosis code is

$$\{\texttt{A–Z}\}\{\texttt{0–9}\}\{\texttt{0–9}\}[\,\texttt{.}\,][\texttt{0–9}]$$

The code begins with a single letter followed by two digits. It may have an additional third digit after the period.

For example, in the ICD-10 coding system, E11.0 (Type 2 diabetes mellitus: With coma) and C56 (Malignant neoplasm of ovary) are diagnosis codes, although some datasets record (and some people write) E110 rather than E11.0. The `icd10` commands understand both ways of recording codes. The commands are also insensitive to codes recorded with or without leading and trailing blanks and are case-insensitive.

All the following codes are acceptable formats to record codes in Stata.

```
N94.0
  M32
K12
F102
x40
```

The list of defined codes and their associated descriptions is provided under license from the World Health Organization (WHO); see [R] **copyright icd10**. To view the current license and a log of changes that WHO has made to the list of ICD-10 codes since the `icd10` commands were implemented in Stata, type

```
. icd10 query
ICD-10 Version and Change Log
  License agreement
    ICD-10 codes used by permission of the World Health Organization (WHO),
        from: International Statistical Classification of Diseases and
        Related Health Problems, Tenth Revision (ICD-10) 2010 Edition. Vols.
        1-3. Geneva, World Health Organization, 2011.
    See copyright icd10 for the ICD-10 copyright notification.
  Edition 2016
    The ICD-10 data were obtained from WHO on 05feb2015.
    All updates scheduled for implementation through 01jan2016 have been
        applied.  This was verified using the Cumulative Official Updates to
        ICD-10 which may be found at
        http://www.who.int/classifications/icd/icd10updates/en/index.html and
        then clicking the "Official WHO Updates combined 1996-2014 Volume 3"
        link.
    Between 2015 and 2016:
           13 codes added,   4 codes deleted,   0 code descriptions changed.
(output omitted)
```

❑ Technical note

It is possible for codes to have up to two more digits to form five- and six-character codes. Supplemental subdivisions of ICD-10 codes may occur at the fifth and sixth characters. These supplemental subdivisions are primarily used to indicate anatomical site and additional information about the diagnosis, for example, whether a fracture was open or closed (World Health Organization 2011). However, these codes are not part of the standard four-character system codified by WHO for international morbidity and mortality reporting and are not considered valid by `icd10`.

If your data contain these longer codes, you can use `icd10 generate` with option `short` to shorten your codes to the relevant four-character subcategory code. Any existing three- and four-character codes in the data are left as they were originally.

❑

Managing datasets with ICD-10 codes

The `icd10` suite of commands has three data-management commands. `icd10 check` verifies that the ICD-10 codes in *varname* are valid. `icd10 clean` standardizes the format of ICD-10 codes in *varname*. And `icd10 generate` produces a new variable from an existing variable containing ICD-10 codes. It will create a variable containing the associated category code, a description of the code, or a binary indicator for whether the code is in a specified set of codes.

Example 1: Checking the validity of a variable

Although not necessary, a good place to start is with `icd10 check`. The commands in the `icd10` suite will return an error message if the codes in your data are not valid. Running `icd10 check` is a good way to avoid error messages later.

The `australia10` dataset contains total deaths in 2010 for males and females from Australia, taken from WHO's mortality data. Below, we `list` the first 10 observations.

```
. use http://www.stata-press.com/data/r14/australia10
(Australian mortality data, 2010)
. list in 1/10, sepby(cause) noobs
```

cause	sex	deaths
A020	Male	1
A020	Female	4
A021	Male	3
A021	Female	1
A047	Male	16
A047	Female	25
A048	Female	4
A049	Male	1
A049	Female	1
A063	Male	1

We will specify the `generate()` option to create a new variable called `prob` that will indicate that the code in `cause` is valid (`prob` = 0) or will indicate a value of 1 through 8 for the reason the code is not valid. `icd10 check` also creates a value of 99, which indicates that the code is not defined but otherwise conforms to the formatting requirements for ICD-10 codes.

```
. icd10 check cause, generate(prob)
(cause contains no missing values)
cause contains undefined codes:
    1.  Invalid placement of period                    0
    2.  Too many periods                               0
    3.  Code too short                                 0
    4.  Code too long                                  0
    5.  Invalid 1st char (not A-Z)                     0
    6.  Invalid 2nd char (not 0-9)                     0
    7.  Invalid 3rd char (not 0-9)                     0
    8.  Invalid 4th char (not 0-9)                     0
   99.  Code not defined                               6
        Total                                          6
```

`icd10 check` reports that there are three observations with undefined codes. In this case, this is because we failed to specify that the data were reported using the ICD-10 codes from 2010.

```
. drop prob
. icd10 check cause, generate(prob) year(2010)
(cause contains defined codes; no missing values)
```

We see now that there are no errors in our dataset.

◁

Example 2: Standardizing the format of codes

If we plan to do any reporting with these codes later, we may want to make them more readable, so we use `icd10 clean` with the `dots` option.

When we listed our data before, it was sorted by cause of death and showed very few deaths assigned to the first several codes. It might be more interesting to see the most frequent causes of death. So before we list the data this time, we sort them in descending order with `gsort`.

```
. icd10 clean cause, dots replace
. gsort -deaths
. list cause sex deaths in 1/10, sepby(cause)
```

	cause	sex	deaths
1.	I21.9	Male	5,057
2.	I21.9	Female	4,885
3.	C34.9	Male	4,859
4.	I25.9	Male	3,805
5.	I25.9	Female	3,636
6.	F03	Female	3,517
7.	C61	Male	3,236
8.	I64	Female	3,204
9.	C34.9	Female	3,130
10.	C50.9	Female	2,842

Now it is clear that we have a mix of three- and four-character codes. `icd10 clean` will automatically change the display format of the diagnosis variable so that it is left-aligned, as shown above.

Example 3: Looking up a single code

In example 2, we see that the highest number of reported deaths for men and women is for code I21.9. If we were curious about what this code is, we could type

```
. icd10 lookup I21.9
    I21.9 Acute myocardial infarction, unspecified
```

and we would see that these are deaths from acute myocardial infarction, commonly known as heart attacks. Because the `icd10` commands are case-insensitive and do not care whether we use the dot, we could have typed `i21.9`, `I219`, or `i219`, and Stata would have returned the same results.

Creating new variables

We now proceed to create new variables for later use.

▷ Example 4: Creating an indicator variable

Suppose that after watching several high-action nature shows on television, we now believe that death due to shark attack is common in Australia. It did not show up in our top-ten list above, but we would like to see how many we have in our data. We can look up the code using WHO's interactive web utility (http://apps.who.int/classifications/icd10/browse/2010) and then use `icd10 generate` with the `range()` option to create an indicator for whether death occurred by shark bite (`shark`).

```
. icd10 generate shark=cause, range(W56)
. tabulate shark [fweight=deaths]
```

shark	Freq.	Percent	Cum.
0	143,472	100.00	100.00
1	1	0.00	100.00
Total	143,473	100.00	

Reality was not nearly as exciting as television—there was only one death with a code relating to shark bite in Australia in 2010.

If we wanted to study something less sensational, we could expand the *icd10rangelist* to a more complex list of codes. For example, perhaps we want to study the number of deaths from myocardial infarction (MI) and complications that occurred afterward. We might pick codes I21.0 through I21.9, I22.0 through I22.9, and I23.0 through I23.8. We could create the variable `mi` by typing

```
. icd10 generate mi=cause, range(I210/I219 I220/I229 I230/I238)
. tabulate mi [fweight=deaths]
```

mi	Freq.	Percent	Cum.
0	133,522	93.06	93.06
1	9,951	6.94	100.00
Total	143,473	100.00	

We see that 9,951 deaths were from MI or complications thereof, which equates to about 6.9% of all deaths in Australia in 2010. It appears that hearts are far more dangerous than sharks.

◁

❑ Technical note

WHO reserves codes in categories U00 through U49 for the provisional assignment of new diseases and designates codes U50 through U99 for research purposes (World Health Organization 2011).

In general, codes in categories U50 through U99 are treated as undefined. This means that you do not need to take any special steps as long as your codes fit within the accepted four-character format. However, if you wish to exclude U codes from the commands, you can use the `if` qualifier.

With the exception of `icd10 generate` with the `description` option, the `icd10` commands will continue to work as normal with undefined U codes. As a rule, `icd10 generate` with the `description` option will return missing values for codes U50 through U99. Note that some of these codes, however, are defined and considered valid by `icd10` because WHO has distributed descriptions for them. For these codes, `icd10 generate` with option `description` will return results. The affected codes vary by year.

❑

Stored results

`icd10 check` stores the following in `r()`:

Scalars

`r(e#)`	number of errors of type #
`r(esum)`	total number of errors
`r(miss)`	number of missing values

`icd10 clean` stores the following in `r()`:

Scalars

`r(N)`	number of changes

`icd10 lookup` stores the following in `r()`:

Scalars

`r(N)`	number of codes found

Acknowledgments

We thank the World Health Organization for making ICD-10 codes available to Stata users. See [R] **copyright icd10** for allowed usage.

We thank Joe Canner of the Johns Hopkins University School of Medicine, who wrote `mycd10` and `mycd10p`, which provide many utilities for ICD-10 diagnosis and procedure codes. The commands rely on a user-supplied ICD-10 lookup dataset for diagnosis codes and ICD-10-PCS codes from the U.S. Centers for Medicare and Medicaid Services for procedure codes.

References

de Kraker, M. E. A., M. Wolkewitz, P. G. Davey, H. Grundmann, and Burden Study Group. 2011. Clinical impact of antimicrobial resistance in European hospitals: Excess mortality and length of hospital stay related to methicillin-resistant staphylococcus aureus bloodstream infections. *Antimicrobial Agents and Chemotherapy* 55: 1598–1605.

Klevens, R. M., M. A. Morrison, J. Nadle, S. Petit, K. Gershman, S. Ray, L. H. Harrison, R. Lynfield, G. Dumyati, J. M. Townes, A. S. Craig, E. R. Zell, G. E. Fosheim, L. K. McDougal, R. B. Carey, and S. K. Fridkin. 2007. Invasive methicillin-resistant Staphylococcus aureus infections in the United States. *Journal of the American Medical Association* 298: 1763–1771.

World Health Organization. 2011. International Statistical Classification of Diseases and Related Health Problems, Vol. 2: 2016 Edition. Instruction manual. http://apps.who.int/classifications/icd10/browse/Content/statichtml/ICD10Volume2_en_2016.pdf.

World Health Organization Mortality Data Base (Cause of Death Query online; accessed December 11, 2014). http://apps.who.int/healthinfo/statistics/mortality/causeofdeath_query/.

Also see

[D] **icd** — Introduction to ICD commands