assertnested — Verify variables nested

Description

assertnested verifies that the values of variables are nested within the values of other variables. If they are nested, the command produces no output. If they are not nested, assertnested informs you that they are not and issues an error return code of 459; see [U] 8 Error messages and return codes.

Quick start

Confirm that the values of psu are nested within stratum
assertnested stratum psu

Confirm that the values of IDs in student are nested within school, which is nested within district
assertnested district school student

For panel data, where panels are individuals with IDs stored in panelid, check that values of age and income are the same for all observations in each panel
assertnested panelid, within(age income)

As above, but treat any missing values the same as nonmissing values
assertnested panelid, within(age income) missing

Syntax

assertnested varlist [if] [in] [ , within(withinvars) missing ]

The variables in varlist are given in the order of biggest grouping to smallest grouping.
by is allowed; see [D] by.

Options

within(withinvars) asserts that the values of varlist are nested within each of the variables in withinvars. That is, assertnested varlist, within(w1 w2 ...) will issue an error if any of assertnested w1 varlist, assertnested w2 varlist, ... issue an error.

missing specifies that missing values in varlist and withinvars are to be treated the same as nonmissing values.
assertnested is a convenience command for checking whether variables are nested. We say that 
v2 is nested within v1 if for all observations that have the same value of v2, the observations also have the same value of v1.

Here are data that are nested.

```
. list v1 v2, sepby(v1)

<table>
<thead>
<tr>
<th></th>
<th>v1</th>
<th>v2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>
```

```
. assertnested v1 v2
```

assertnested succeeds.

Here are data that are not nested.

```
. list v1 v3, sepby(v1)

<table>
<thead>
<tr>
<th></th>
<th>v1</th>
<th>v3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>
```

```
. assertnested v1 v3
v3 not nested within v1
r(459);
```

assertnested fails.

Running
```
assertnested v1 v2 v3
```
is the same as running
```
assertnested v1 v2
assertnested v2 v3
```

Variables must be specified with the biggest nested grouping first, then the second biggest nested grouping, and so on, to the smallest nested grouping.
Example 1: Nested variables

We have a dataset consisting of two school districts in Texas: the district for the city of College Station and the district for the city of Richardson. The dataset contains the actual names of all the public schools in the variable `school` in these districts, given by variable `district`. The dataset contains fictitious student IDs in the variable `student`.

We want to assert that `student` is nested within `school` and that `school` is nested within `district`.

```
use https://www.stata-press.com/data/r16/schools
assertnested district school student
```

Schools are not nested within district! Are some schools in both districts? That is impossible. But it is possible that both districts have one or more schools with the same name. Let’s find them.

We use `egen`’s `tag()` function to tag one observation for each distinct value of `district` for each school. Then we sum up the number of tags in each school. If the schools were nested within district, there would be only one tag per school. We list the districts and schools with more than one tag.

```
egen tag_district = tag(school district)
by school: egen ndistrict = sum(tag_district)
list district school if tag_district == 1 & ndistrict > 1, noobs
```

<table>
<thead>
<tr>
<th>district</th>
<th>school</th>
</tr>
</thead>
<tbody>
<tr>
<td>Richardson</td>
<td>Spring Creek Elementary School</td>
</tr>
<tr>
<td>College Station</td>
<td>Spring Creek Elementary School</td>
</tr>
</tbody>
</table>

Both College Station and Richardson have schools named Spring Creek Elementary School. If we want to check that students are nested within schools, we need to do the check separately by district.

```
by district: assertnested school student
```

Or else Texans need to get more creative about naming their schools.

Example 2: Variables constant within panels

Commands that work with panel data in Stata require the data to be in long form. That is, multiple Stata observations for each panel. Saying a variable is constant within each panel is the same as saying the panels are nested within that variable. `assertnested` allows you to assert that variables are constant within each panel.

We illustrate this with choice model data. Choice model data are stored like panel data in that each individual has multiple observations, one for each possible choice. Characteristics of the individual should be constant across observations for an individual.

We load a dataset with consumer choices for purchasing a new car (see [CM] Intro 2 for a description of these data). Then we check that `gender` and `income` are constant for the observations with the same `consumerid` by using the `within()` option.
. use https://www.stata-press.com/data/r16/carchoice, clear
(Car choice data)
. assertnested consumerid, within(gender income)

The `within()` option is a convenient way to do multiple assertions. The above is the same as running

. assertnested gender consumerid
. assertnested income consumerid

The option `missing` can be specified to treat missing values the same as any other value.

. assertnested consumerid, within(gender income) missing
  consumerid not nested within gender
  r(459);

We see that `gender` is not constant for some consumers when we treat missing values like any other value. Let’s list one person who has missing values for `gender`:

. list consumerid gender if consumerid == 142, abbrev(10)

```
consumerid  gender
------------
   509.    142  .
   510.    142  Male
   511.    142  Male
   512.    142  Male

```

This person has a missing value for `gender` for one observation and nonmissing values for other observations. For the data to pass `assertnested` with the option `missing`, the variable would have to be either all missing or all nonmissing (and the same value) for each individual.

Also see

[D] assert — Verify truth of claim
[CM] Intro 2 — Data layout
[P] capture — Capture return code
[SVY] Survey — Introduction to survey commands
[XT] xt — Introduction to xt commands
[U] 16 Do-files