Title

stata.com

tabdisp — Display tables

Description Syntax Options Remarks and examples Also see

Description

`tabdisp` displays data in a table. `tabdisp` calculates no statistics and is intended for use by programmers.

For the corresponding command that calculates statistics and displays them in a table, see [R] **table**.

Although `tabdisp` is intended for programming applications, it can be used interactively for listing data.

Syntax

```
tabdisp rowvar [colvar [supercolvar]] [if] [in], cellvar(varnames)
    [by(superrowvars) format(%fmt) center left concise missing totals
    dotz cellwidth(#) csepwidth(#) scsepwidth(#) stubwidth(#)]
```

`by` is allowed; see [D] **by**.

rowvar, *colvar*, and *supercolvar* may be numeric or string variables. Rows, columns, supercolumns, and superrows are thus defined as

row 1	.
row 2	.

	supercol 1		supercol 2	
	col 1	col 2	col 1	col 2
row 1	.	.	.	.
row 2	.	.	.	.

	col 1	col 2
row 1	.	.
row 2	.	.

	supercol 1		supercol 2	
	col 1	col 2	col 1	col 2
superrow 1:				
row 1	.	.	.	.
row 2	.	.	.	.
superrow 2:				
row 1	.	.	.	.
row 2	.	.	.	.

Options

`cellvar(`*varnames*`)` is required; it specifies the numeric or string variables containing the values to be displayed in the table's cells. Up to five variable names may be specified.

`by(`*superrowvars*`)` specifies numeric or string variables to be treated as superrows. Up to four variables may be specified.

`format(%`*fmt*`)` specifies the display format for presenting numbers in the table's cells. `format(%9.0g)` is the default; `format(%9.2f)` is a popular alternative. The width of the format you specify does not matter, except that `%`*fmt* must be valid. The width of the cells is chosen by `tabdisp` to be what it thinks looks best. The `cellwidth()` option allows you to override `tabdisp`'s choice.

`center` specifies that results be centered in the table's cells. The default is to right-align results. For centering to work well, you typically need to specify a display format as well. `center format(%9.2f)` is popular.

`left` specifies that column labels be left-aligned. The default is to right-align column labels to distinguish them from supercolumn labels, which are left-aligned. If you specify `left`, both column and supercolumn labels are left-aligned.

`concise` specifies that rows with all missing entries not be displayed.

`missing` specifies that, in cells containing missing values, the missing value (`.`, `.a`, `.b`, ..., or `.z`) be displayed. The default is that cells with missing values are left blank.

`totals` specifies that observations where *rowvar*, *colvar*, *supercolvar*, or *superrowvars* contain the system missing value (`.`) be interpreted as containing the corresponding totals of `cellvar()`, and that the table be labeled accordingly. If the `dotz` option is also specified, observations where the stub variables contain `.z` will be thus interpreted.

`dotz` specifies that the roles of missing values `.` and `.z` be interchanged in labeling the stubs of the table. By default, if any of *rowvar*, *colvar*, *supercolvar*, and *superrowvars* contains missing (`.`, `.a`, `.b`, ..., or `.z`), then "`.`" is placed last in the ordering. `dotz` specifies that `.z` be placed last. Also, if option `totals` is specified, `.z` values rather than "`.`" values will be labeled "Total".

`cellwidth(#)` specifies the width of the cell in units of digit widths; 10 means the space occupied by 10 digits, which is `0123456789`. The default `cellwidth()` is not a fixed number but rather a number chosen by `tabdisp` to spread the table out while presenting a reasonable number of columns across the page.

`csepwidth(#)` specifies the separation between columns in units of digit widths. The default is not a fixed number but rather a number chosen by `tabdisp` according to what it thinks looks best.

`scsepwidth(#)` specifies the separation between supercolumns in units of digit widths. The default is not a fixed number but rather a number chosen by `tabdisp` according to what it thinks looks best.

`stubwidth(#)` specifies the width, in units of digit widths, to be allocated to the left stub of the table. The default is not a fixed number but rather a number chosen by `tabdisp` according to what it thinks looks best.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Limits

Up to four variables may be specified in the by() option, so with the three row, column, and supercolumn variables, seven-way tables may be displayed.

Up to five variables may be displayed in each cell of the table.

The sum of the number of rows, columns, supercolumns, and superrows is called the number of margins. A table may contain up to 3,000 margins. Thus a one-way table may contain 3,000 rows. A two-way table could contain 2,998 rows and 2 columns, 2,997 rows and 3 columns, . . . , 1,500 rows and 1,500 columns, . . . , or 2 rows and 2,998 columns. A three-way table is similarly limited by the sum of the number of rows, columns, and supercolumns. An $r \times c \times d$ table is feasible if $r + c + d \leq 3{,}000$. The limit is set in terms of the sum of the rows, columns, supercolumns, and superrows—not, as you might expect, their product.

Introduction

If you have not read [R] **table**, please do so. tabdisp is what table uses to display the tables.

tabdisp calculates nothing. tabdisp instead displays the data in memory. In this, think of tabdisp as an alternative to list. Consider the following little dataset:

```
. use http://www.stata-press.com/data/r14/tabdxmpl1
. list
```

	a	b	c
1.	0	1	15
2.	0	2	26
3.	0	3	11
4.	1	1	14
5.	1	2	12
6.	1	3	7

We can use tabdisp to list it:

```
. tabdisp a b, cell(c)
```

a	b 1	2	3
0	15	26	11
1	14	12	7

tabdisp is merely an alternative way to list the data. It is when the data in memory are statistics by category that tabdisp becomes really useful. table provides one prepackaging of that idea.

Unlike list, tabdisp is unaffected by the order of the data. Here are the same data in a different order:

```
. use http://www.stata-press.com/data/r14/tabdxmpl2
. list
```

	a	b	c
1.	1	3	7
2.	0	3	11
3.	1	2	12
4.	1	1	14
5.	0	1	15
6.	0	2	26

and yet the output of `tabdisp` is unaffected.

```
. tabdisp a b, cell(c)
```

a	b 1	2	3
0	15	26	11
1	14	12	7

Nor does `tabdisp` care if one of the cells is missing in the data.

```
. drop in 6
(1 observation deleted)
. tabdisp a b, cell(c)
```

a	b 1	2	3
0	15		11
1	14	12	7

On the other hand, `tabdisp` assumes that each value combination of the row, column, superrow, and supercolumn variables occurs only once. If that is not so, `tabdisp` displays the earliest occurring value:

```
. input
            a           b           c
  6. 0 1 99
  7. end
. list
```

	a	b	c
1.	1	3	7
2.	0	3	11
3.	1	2	12
4.	1	1	14
5.	0	1	15
6.	0	1	99

```
. tabdisp a b, cell(c)
```

a	b 1	2	3
0	15		11
1	14	12	7

Thus our previous claim that `tabdisp` was unaffected by sort order has this one exception.

Finally, `tabdisp` uses variable and value labels when they are defined:

```
. label var a "Sex"
. label define sex 0 male 1 female
. label values a sex
. label var b "Treatment Group"
. label def tg 1 "controls" 2 "low dose" 3 "high dose"
. label values b tg
. tabdisp a b, cell(c)
```

Sex	Treatment Group controls	low dose	high dose
male	15		11
female	14	12	7

There are two things you can do with `tabdisp`.

You can use it to list data, but be certain that you have a unique identifier. In the automobile dataset, the variable make is unique:

```
. use http://www.stata-press.com/data/r14/auto2, clear
(1978 Automobile Data)
. list make mpg weight displ rep78
```

	make	mpg	weight	displa~t	rep78
1.	AMC Concord	22	2,930	121	Average
2.	AMC Pacer	17	3,350	258	Average
3.	AMC Spirit	22	2,640	121	.
4.	Buick Century	20	3,250	196	Average
5.	Buick Electra	15	4,080	350	Good
6.	Buick LeSabre	18	3,670	231	Average
7.	Buick Opel	26	2,230	304	.
8.	Buick Regal	20	3,280	196	Average
9.	Buick Riviera	16	3,880	231	Average
10.	Buick Skylark	19	3,400	231	Average
11.	Cad. Deville	14	4,330	425	Average
12.	Cad. Eldorado	14	3,900	350	Fair
13.	Cad. Seville	21	4,290	350	Average
14.	Chev. Chevette	29	2,110	231	Average
15.	Chev. Impala	16	3,690	250	Good
16.	Chev. Malibu	22	3,180	200	Average
17.	Chev. Monte Carlo	22	3,220	200	Fair
18.	Chev. Monza	24	2,750	151	Fair
19.	Chev. Nova	19	3,430	250	Average
20.	Dodge Colt	30	2,120	98	Excellent
21.	Dodge Diplomat	18	3,600	318	Fair
22.	Dodge Magnum	16	3,600	318	Fair
23.	Dodge St. Regis	17	3,740	225	Fair
24.	Ford Fiesta	28	1,800	98	Good
25.	Ford Mustang	21	2,650	140	Average
26.	Linc. Continental	12	4,840	400	Average
27.	Linc. Mark V	12	4,720	400	Average
28.	Linc. Versailles	14	3,830	302	Average
29.	Merc. Bobcat	22	2,580	140	Good
30.	Merc. Cougar	14	4,060	302	Good
31.	Merc. Marquis	15	3,720	302	Average
32.	Merc. Monarch	18	3,370	250	Average
33.	Merc. XR-7	14	4,130	302	Good
34.	Merc. Zephyr	20	2,830	140	Average
35.	Olds 98	21	4,060	350	Good
36.	Olds Cutl Supr	19	3,310	231	Average
37.	Olds Cutlass	19	3,300	231	Average
38.	Olds Delta 88	18	3,690	231	Good
39.	Olds Omega	19	3,370	231	Average
40.	Olds Starfire	24	2,730	151	Poor
41.	Olds Toronado	16	4,030	350	Average
42.	Plym. Arrow	28	3,260	156	Average
43.	Plym. Champ	34	1,800	86	Excellent

44.	Plym. Horizon	25	2,200	105	Average
45.	Plym. Sapporo	26	2,520	119	.
46.	Plym. Volare	18	3,330	225	Fair
47.	Pont. Catalina	18	3,700	231	Good
48.	Pont. Firebird	18	3,470	231	Poor
49.	Pont. Grand Prix	19	3,210	231	Average
50.	Pont. Le Mans	19	3,200	231	Average
51.	Pont. Phoenix	19	3,420	231	.
52.	Pont. Sunbird	24	2,690	151	Fair
53.	Audi 5000	17	2,830	131	Excellent
54.	Audi Fox	23	2,070	97	Average
55.	BMW 320i	25	2,650	121	Good
56.	Datsun 200	23	2,370	119	Good
57.	Datsun 210	35	2,020	85	Excellent
58.	Datsun 510	24	2,280	119	Good
59.	Datsun 810	21	2,750	146	Good
60.	Fiat Strada	21	2,130	105	Average
61.	Honda Accord	25	2,240	107	Excellent
62.	Honda Civic	28	1,760	91	Good
63.	Mazda GLC	30	1,980	86	Good
64.	Peugeot 604	14	3,420	163	.
65.	Renault Le Car	26	1,830	79	Average
66.	Subaru	35	2,050	97	Excellent
67.	Toyota Celica	18	2,410	134	Excellent
68.	Toyota Corolla	31	2,200	97	Excellent
69.	Toyota Corona	18	2,670	134	Excellent
70.	VW Dasher	23	2,160	97	Good
71.	VW Diesel	41	2,040	90	Excellent
72.	VW Rabbit	25	1,930	89	Good
73.	VW Scirocco	25	1,990	97	Good
74.	Volvo 260	17	3,170	163	Excellent

```
. tabdisp make, cell(mpg weight displ rep78)
```

Make and Model	Mileage (mpg)	Weight (lbs.)	displacement	rep78
AMC Concord	22	2,930	121	Average
AMC Pacer	17	3,350	258	Average
AMC Spirit	22	2,640	121	
Audi 5000	17	2,830	131	Excellent
Audi Fox	23	2,070	97	Average
BMW 320i	25	2,650	121	Good
Buick Century	20	3,250	196	Average
Buick Electra	15	4,080	350	Good
Buick LeSabre	18	3,670	231	Average
Buick Opel	26	2,230	304	
Buick Regal	20	3,280	196	Average
Buick Riviera	16	3,880	231	Average
Buick Skylark	19	3,400	231	Average
Cad. Deville	14	4,330	425	Average
Cad. Eldorado	14	3,900	350	Fair
Cad. Seville	21	4,290	350	Average
Chev. Chevette	29	2,110	231	Average
Chev. Impala	16	3,690	250	Good
Chev. Malibu	22	3,180	200	Average
Chev. Monte Carlo	22	3,220	200	Fair

Chev. Monza	24	2,750	151	Fair
Chev. Nova	19	3,430	250	Average
Datsun 200	23	2,370	119	Good
Datsun 210	35	2,020	85	Excellent
Datsun 510	24	2,280	119	Good
Datsun 810	21	2,750	146	Good
Dodge Colt	30	2,120	98	Excellent
Dodge Diplomat	18	3,600	318	Fair
Dodge Magnum	16	3,600	318	Fair
Dodge St. Regis	17	3,740	225	Fair
Fiat Strada	21	2,130	105	Average
Ford Fiesta	28	1,800	98	Good
Ford Mustang	21	2,650	140	Average
Honda Accord	25	2,240	107	Excellent
Honda Civic	28	1,760	91	Good
Linc. Continental	12	4,840	400	Average
Linc. Mark V	12	4,720	400	Average
Linc. Versailles	14	3,830	302	Average
Mazda GLC	30	1,980	86	Good
Merc. Bobcat	22	2,580	140	Good
Merc. Cougar	14	4,060	302	Good
Merc. Marquis	15	3,720	302	Average
Merc. Monarch	18	3,370	250	Average
Merc. XR-7	14	4,130	302	Good
Merc. Zephyr	20	2,830	140	Average
Olds 98	21	4,060	350	Good
Olds Cutl Supr	19	3,310	231	Average
Olds Cutlass	19	3,300	231	Average
Olds Delta 88	18	3,690	231	Good
Olds Omega	19	3,370	231	Average
Olds Starfire	24	2,730	151	Poor
Olds Toronado	16	4,030	350	Average
Peugeot 604	14	3,420	163	
Plym. Arrow	28	3,260	156	Average
Plym. Champ	34	1,800	86	Excellent
Plym. Horizon	25	2,200	105	Average
Plym. Sapporo	26	2,520	119	
Plym. Volare	18	3,330	225	Fair
Pont. Catalina	18	3,700	231	Good
Pont. Firebird	18	3,470	231	Poor
Pont. Grand Prix	19	3,210	231	Average
Pont. Le Mans	19	3,200	231	Average
Pont. Phoenix	19	3,420	231	
Pont. Sunbird	24	2,690	151	Fair
Renault Le Car	26	1,830	79	Average
Subaru	35	2,050	97	Excellent
Toyota Celica	18	2,410	134	Excellent
Toyota Corolla	31	2,200	97	Excellent
Toyota Corona	18	2,670	134	Excellent
VW Dasher	23	2,160	97	Good
VW Diesel	41	2,040	90	Excellent
VW Rabbit	25	1,930	89	Good
VW Scirocco	25	1,990	97	Good
Volvo 260	17	3,170	163	Excellent

Mostly, however, tabdisp is intended for use when you have a dataset of statistics that you want to display:

```
. collapse (mean) mpg, by(foreign rep78)
. list
```

	rep78	foreign	mpg
1.	Poor	Domestic	21
2.	Fair	Domestic	19.125
3.	Average	Domestic	19
4.	Good	Domestic	18.4444
5.	Excellent	Domestic	32
6.	.	Domestic	23.25
7.	Average	Foreign	23.3333
8.	Good	Foreign	24.8889
9.	Excellent	Foreign	26.3333
10.	.	Foreign	14

```
. tabdisp foreign rep78, cell(mpg)
```

	Repair Record 1978					
Car type	Poor	Fair	Average	Good	Excellent	.
Domestic	21	19.125	19	18.4444	32	23.25
Foreign			23.3333	24.8889	26.3333	14

```
. drop if rep78==.
(2 observations deleted)
. tabdisp foreign rep78, cell(mpg) format(%9.2f) center
```

	Repair Record 1978				
Car type	Poor	Fair	Average	Good	Excellent
Domestic	21.00	19.12	19.00	18.44	32.00
Foreign			23.33	24.89	26.33

Treatment of string variables

The variables specifying the rows, columns, supercolumns, and superrows may be numeric or string. Also, the variables specified for inclusion in the table may be numeric or string. In the example below, all variables are strings, including reaction:

```
. use http://www.stata-press.com/data/r14/tabdxmpl3, clear
. tabdisp agecat sex party, c(reaction) center
```

	Party Affiliation and Sex			
	Democrat		Republican	
Age category	Female	Male	Female	Male
Old	Favor	Favor	Indifferent	Strongly Favor
Young	Strongly Favor	Indifferent	Disfavor	Disfavor

Treatment of missing values

The `cellvar()` variables specified for inclusion in the table may contain missing values, and whether the variable contains a missing value or the observation is missing altogether makes no difference:

```
. use http://www.stata-press.com/data/r14/tabdxmpl4
. list
```

	sex	response	pop
1.	0	0	12
2.	0	1	20
3.	0	2	.a
4.	1	0	15
5.	1	1	11

```
. tabdisp sex response, cell(pop)
```

	Response		
Sex	0	1	2
0	12	20	
1	15	11	

In the above output, the $(1, 3)$ cell is empty because the observation for `sex` $= 0$ and `response` $= 2$ has a missing value for `pop`. The $(2, 3)$ cell is empty because there is no observation for `sex` $= 1$ and `response` $= 2$.

If you specify the `missing` option, rather than cells being left blank, the missing value will be displayed:

```
. tabdisp sex response, cell(pop) missing
```

	Response		
Sex	0	1	2
0	12	20	.a
1	15	11	.

Missing values of the row, column, superrow, and supercolumn variables are allowed, and, by default, missing values are given no special meaning. The output below is from a different dataset.

```
. use http://www.stata-press.com/data/r14/tabdxmpl5
. list
```

	sex	response	pop
1.	0	0	15
2.	0	1	11
3.	0	.	26
4.	1	0	20
5.	1	1	24
6.	1	.	44
7.	.	.	70
8.	.	0	35
9.	.	1	35

```
. tabdisp sex response, cell(pop)
```

	response		
sex	0	1	.
0	15	11	26
1	20	24	44
.	35	35	70

If you specify the `total` option, however, the system missing values are labeled as reflecting totals:

```
. tabdisp sex response, cell(pop) total
```

	response		
sex	0	1	Total
0	15	11	26
1	20	24	44
Total	35	35	70

`tabdisp` did not calculate the totals; it merely labeled the results as being totals. The number 70 appears in the lower right because there happens to be an observation in the dataset where both `sex` and `response` contain a system missing value and `pop` = 70.

Here the row and column variables were numeric. If they had been strings, the `total` option would have given the special interpretation to `sex` = `""` and `response` = `""`.

Also see

[R] **table** — Flexible table of summary statistics

[R] **tabstat** — Compact table of summary statistics

[R] **tabulate oneway** — One-way table of frequencies

[R] **tabulate, summarize()** — One- and two-way tables of summary statistics

[R] **tabulate twoway** — Two-way table of frequencies

[D] **collapse** — Make dataset of summary statistics