10 Listing data and basic command syntax

Command syntax

This chapter gives a basic lesson on Stata's command syntax while showing how to control the appearance of a data list.

As we have seen throughout this manual, you have a choice between using menus and dialogs and using the Command window. Although many find the menus more natural and the Command window baffling at first, some practice makes working with the Command window often much faster than using menus and dialogs. The Command window can become a faster way of working because of the clean and regular syntax of Stata commands. We will cover enough to get you started; help language has more information and examples, and [U] **11 Language syntax** has all the details.

The syntax for the list command can be seen by typing help list:

$$\underline{\text{l}}\text{ist} \ \big[\ \textit{varlist}\ \big]\ \big[\ \textit{if}\ \big]\ \big[\ \textit{in}\ \big]\ \big[\ \text{, \textit{options}}\ \big]$$

Here is how to read this syntax:
- Anything inside square brackets is optional. For the list command,
    a. *varlist* is optional. A *varlist* is a list of variable names.
    b. *if* is optional. The if qualifier restricts the command to run only on those observations for which the qualifier is true. We saw examples of this in [GSW] **6 Using the Data Editor**.
    c. *in* is optional. The in qualifier restricts the command to run on particular observation numbers.
    d. , and *options* are optional. *options* are separated from the rest of the command by a comma.
- Optional pieces do not preclude one another unless explicitly stated. For the list command, it is possible to use a *varlist* with *if* and *in*.
- If a part of a word is underlined, the underlined part is the minimum abbreviation. Any abbreviation at least this long is acceptable.
    a. The l in list is underlined, so l, li, and lis are all equivalent to list.
- Anything not inside square brackets is required. For the list command, only the command itself is required.

Keeping these rules in mind, let's investigate how list behaves when called with different arguments. We will be using the dataset afewcarslab.dta from the end of the previous chapter.

list with a variable list

Variable lists (or *varlist*s) can be specified in a variety of ways, all designed to save typing and encourage good variable names.
- The *varlist* is optional for list. This means that if no variables are specified, it is equivalent to specifying all variables. Another way to think of it is that the default behavior of the command is to run on all variables unless restricted by a *varlist*.
- You can list a subset of variables explicitly, as in list make mpg price.
- There are also many shorthand notations:
    m* means all variables starting with m.
    price-weight means all variables from price through weight in the dataset order.
    ma?e means all variables starting with ma, followed by any character, and ending in e.

● You can list a variable by using an abbreviation unique to that variable, as in `list gear_r~o`. If the abbreviation is not unique, Stata returns an error message.

```
. list

                     make    price    mpg    weight    gear_r~o     foreign

  1.        VW Rabbit     4697     25      1930        3.78      Foreign
  2.          Olds 98     8814     21      4060        2.41     Domestic
  3.      Chev. Monza     3667      .      2750        2.73     Domestic
  4.                      4099     22      2930        3.58     Domestic
  5.        Datsun 510    5079     24      2280        3.54      Foreign

  6.      Buick Regal     5189     20      3280        2.93     Domestic
  7.      Datsun 810      8129      .      2750        3.55      Foreign

. l make mpg price

                     make    mpg    price

  1.        VW Rabbit     25     4697
  2.          Olds 98     21     8814
  3.      Chev. Monza      .     3667
  4.                      22     4099
  5.        Datsun 510    24     5079

  6.      Buick Regal     20     5189
  7.      Datsun 810       .     8129

. list m*

                     make    mpg

  1.        VW Rabbit     25
  2.          Olds 98     21
  3.      Chev. Monza      .
  4.                      22
  5.        Datsun 510    24

  6.      Buick Regal     20
  7.      Datsun 810       .

. li price-weight

              price    mpg    weight

  1.          4697     25      1930
  2.          8814     21      4060
  3.          3667      .      2750
  4.          4099     22      2930
  5.          5079     24      2280

  6.          5189     20      3280
  7.          8129      .      2750
```

```
. list ma?e

                    make

        1.     VW Rabbit
        2.        Olds 98
        3.    Chev. Monza
        4.
        5.     Datsun 510

        6.    Buick Regal
        7.     Datsun 810

. l gear_r~o

              gear_r~o

        1.        3.78
        2.        2.41
        3.        2.73
        4.        3.58
        5.        3.54

        6.        2.93
        7.        3.55
```

list with if

The if qualifier uses a logical expression to determine which observations to use. If the expression is true, the observation is used in the command; otherwise, it is skipped. The operators whose results are either true or false are

<	less than
<=	less than or equal
==	equal
>	greater than
>=	greater than or equal
!=	not equal
&	and
|	or
!	not (logical negation)
()	parentheses are for grouping to specify order of evaluation

In the logical expressions, & is evaluated before | (similar to multiplication before addition in arithmetic). You can use this in your expressions, but it is often better to use parentheses to ensure that the expressions are evaluated in the proper order. See [U] **13.2 Operators** for complete details.

```
. list
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
2.	Olds 98	8814	21	4060	2.41	Domestic
3.	Chev. Monza	3667	.	2750	2.73	Domestic
4.		4099	22	2930	3.58	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
6.	Buick Regal	5189	20	3280	2.93	Domestic
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. list if mpg > 22
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
3.	Chev. Monza	3667	.	2750	2.73	Domestic
5.	Datsun 510	5079	24	2280	3.54	Foreign
7.	Datsun 810	8129	.	2750	3.55	Foreign

```
. list if (mpg > 22) & !missing(mpg)
```

	make	price	mpg	weight	gear_r~o	foreign
1.	VW Rabbit	4697	25	1930	3.78	Foreign
5.	Datsun 510	5079	24	2280	3.54	Foreign

```
. list make mpg price gear if (mpg > 22) | (price > 8000 & gear < 3.5)
```

	make	mpg	price	gear_r~o
1.	VW Rabbit	25	4697	3.78
2.	Olds 98	21	8814	2.41
3.	Chev. Monza	.	3667	2.73
5.	Datsun 510	24	5079	3.54
7.	Datsun 810	.	8129	3.55

```
. list make mpg if mpg <= 22 in 2/4
```

	make	mpg
2.	Olds 98	21
4.		22

In the listings above, we see more examples of Stata treating missing numerical values as large values, as well as the care that should be taken when the if qualifier is applied to a variable with missing values. See [GSW] **6 Using the Data Editor**.

list with if, common mistakes

Here is a series of listings with common errors and their corrections. See if you can find the errors before reading the correct entry.

```
. list

              make    price    mpg    weight    gear_r~o      foreign

  1.      VW Rabbit     4697     25      1930        3.78      Foreign
  2.        Olds 98     8814     21      4060        2.41     Domestic
  3.    Chev. Monza     3667      .      2750        2.73     Domestic
  4.                    4099     22      2930        3.58     Domestic
  5.      Datsun 510    5079     24      2280        3.54      Foreign

  6.    Buick Regal     5189     20      3280        2.93     Domestic
  7.    Datsun 810      8129      .      2750        3.55      Foreign

. list if mpg=21
=exp not allowed
r(101);
```

The error arises because "equal" is expressed by ==, not by =. Corrected, it becomes

```
. list if mpg==21

            make    price    mpg    weight    gear_r~o      foreign

  2.     Olds 98     8814     21      4060        2.41     Domestic
```

Other common errors with logic:

```
. list if mpg==21 if weight > 4000
invalid syntax
r(198);
. list if mpg==21 and weight > 4000
invalid 'and'
r(198);
```

Joint tests are specified with &, not with the word and or multiple ifs. The if qualifier should be if mpg==21 & weight>4000, not if mpg==21 if weight>4000. Here is its correction:

```
. list if mpg==21 & weight > 4000

            make    price    mpg    weight    gear_r~o      foreign

  2.     Olds 98     8814     21      4060        2.41     Domestic
```

A problem with string variables:

```
. list if make==Datsun 510
Datsun not found
r(111);
```

Strings must be in double quotes, as in `make=="Datsun 510"`. Without the quotes, Stata thinks that `Datsun` is a variable that it cannot find. Here is the correction:

```
. list if make=="Datsun 510"

           make    price   mpg   weight   gear_r~o   foreign

  5.   Datsun 510    5079    24     2280       3.54   Foreign
```

Confusing value labels with strings:

```
. list if foreign=="Domestic"
type mismatch
r(109);
```

Value labels look like strings, but the underlying variable is numeric. Variable `foreign` takes on values 0 and 1 but has the value label that attaches 0 to "Domestic" and 1 to "Foreign" (see [GSW] **9 Labeling data**). To see the underlying numeric values of variables with labeled values, use the `label list` command (see [D] **label**), or investigate the variable with `codebook` *varname*. We can correct the error here by looking for observations where `foreign==0`.

There is a second construction that also allows the use of the value label directly.

```
. list if foreign==0

             make    price   mpg   weight   gear_r~o    foreign

  2.       Olds 98    8814    21     4060       2.41   Domestic
  3.   Chev. Monza    3667     .     2750       2.73   Domestic
  4.                  4099    22     2930       3.58   Domestic
  6.   Buick Regal    5189    20     3280       2.93   Domestic

. list if foreign=="Domestic":origin

             make    price   mpg   weight   gear_r~o    foreign

  2.       Olds 98    8814    21     4060       2.41   Domestic
  3.   Chev. Monza    3667     .     2750       2.73   Domestic
  4.                  4099    22     2930       3.58   Domestic
  6.   Buick Regal    5189    20     3280       2.93   Domestic
```

list with in

The `in` qualifier uses a *numlist* to give a range of observations that should be listed. *numlist*s have the form of one number or *first/last*. Positive numbers count from the beginning of the dataset. Negative numbers count from the end of the dataset. Here are some examples:

```
. list

              make    price    mpg    weight    gear_r~o     foreign

  1.      VW Rabbit    4697     25      1930        3.78      Foreign
  2.        Olds 98    8814     21      4060        2.41     Domestic
  3.    Chev. Monza    3667      .      2750        2.73     Domestic
  4.                   4099     22      2930        3.58     Domestic
  5.     Datsun 510    5079     24      2280        3.54      Foreign

  6.    Buick Regal    5189     20      3280        2.93     Domestic
  7.     Datsun 810    8129      .      2750        3.55      Foreign

. list in 1

              make    price    mpg    weight    gear_r~o     foreign

  1.     VW Rabbit     4697     25      1930        3.78     Foreign

. list in -1

              make    price    mpg    weight    gear_r~o     foreign

  7.    Datsun 810     8129      .      2750        3.55     Foreign

. list in 2/4

              make    price    mpg    weight    gear_r~o     foreign

  2.        Olds 98    8814     21      4060        2.41     Domestic
  3.    Chev. Monza    3667      .      2750        2.73     Domestic
  4.                   4099     22      2930        3.58     Domestic

. list in -3/-2

              make    price    mpg    weight    gear_r~o     foreign

  5.    Datsun 510     5079     24      2280        3.54      Foreign
  6.    Buick Regal    5189     20      3280        2.93     Domestic
```

Controlling the list output

The fine control over `list` output is exercised by specifying one or more options. You can use `sepby()` to separate observations by variable. `abbreviate()` specifies the minimum number of characters to abbreviate a variable name in the output. `divider` draws a vertical line between the variables in the list.

```
. sort foreign make
. list ma p g f, sepby(foreign)

                  make    price    gear_r~o     foreign

     1.                    4099       3.58     Domestic
     2.   Buick Regal      5189       2.93     Domestic
     3.   Chev. Monza      3667       2.73     Domestic
     4.        Olds 98     8814       2.41     Domestic

     5.   Datsun 510       5079       3.54      Foreign
     6.   Datsun 810       8129       3.55      Foreign
     7.     VW Rabbit      4697       3.78      Foreign

. list make weight gear, abbreviate(10)

                  make    weight    gear_ratio

     1.                    2930          3.58
     2.   Buick Regal      3280          2.93
     3.   Chev. Monza      2750          2.73
     4.        Olds 98     4060          2.41
     5.   Datsun 510       2280          3.54

     6.   Datsun 810       2750          3.55
     7.     VW Rabbit      1930          3.78

. list, divider

                  make  | price | mpg | weight | gear_r~o |  foreign

     1.                 |  4099 |  22 |  2930  |    3.58  | Domestic
     2.   Buick Regal   |  5189 |  20 |  3280  |    2.93  | Domestic
     3.   Chev. Monza   |  3667 |   . |  2750  |    2.73  | Domestic
     4.        Olds 98  |  8814 |  21 |  4060  |    2.41  | Domestic
     5.   Datsun 510    |  5079 |  24 |  2280  |    3.54  |  Foreign

     6.   Datsun 810    |  8129 |   . |  2750  |    3.55  |  Foreign
     7.     VW Rabbit   |  4697 |  25 |  1930  |    3.78  |  Foreign
```

The separator() option draws a horizontal line at specified intervals. When not specified, it defaults to a value of 5.

```
. list, separator(3)

             make   price   mpg   weight   gear_r~o    foreign

  1.                 4099    22     2930       3.58    Domestic
  2.   Buick Regal   5189    20     3280       2.93    Domestic
  3.   Chev. Monza   3667     .     2750       2.73    Domestic

  4.        Olds 98  8814    21     4060       2.41    Domestic
  5.    Datsun 510   5079    24     2280       3.54     Foreign
  6.    Datsun 810   8129     .     2750       3.55     Foreign

  7.      VW Rabbit  4697    25     1930       3.78     Foreign
```

Break

If you want to interrupt a Stata command, click on the **Break** button, ⊗ .

It is always safe to click on the **Break** button. After you click on **Break**, the state of the system is the same as if you had never issued the original command.

For suggested citations, see the FAQ on citing Stata documentation.