

# Data import/export

## Importing data files

Limited amounts of data can be entered directly in RESAMPLING STATS using the **DATA** or **COPY** commands (they are the same). However, when you are working with more than 35-40 observations, you should use the **READ** command to import your data from an external ASCII file.

That file could be set up with each column representing a variable and entries separated by spaces or tabs, as follows:

**Table 11.**

---

21	2324	.123	.	.
24	4321	.321	.	.
31	5123	.213	.	.
19	2998	.132	.	.
.	.	.	.	.
.	.	.	.	.

---

It could also be set up with entries separated by commas, in the following fashion:

**Table 12.**

---

21,2324,.123, ..
24,4321,.321, ..
31,5123,.213, ..
19,2998,.132, ..
. . . . .
. . . . .

---

Each column represents a variable, each row (record) the observations for each of the variables. There should be no commas in the numbers themselves (e.g. 2,324 is not acceptable), since a comma signals to RESAMPLING STATS that the number is ended and it is time to proceed to the next number.

## Using the READ command

To bring the above data into your RESAMPLING STATS program, proceed as follows:

- 1 If necessary (see program operation guide), use the **MAXSIZE** command to make room for your data:

**MAXSIZE a 8000 b 8000 c 8000 (for example)**

This command raises the space allotted to vectors **a**, **b** and **c** from 1000 elements to 8000 elements. You can make more efficient use of memory by setting vector limits individually to the level needed; see the **MAXSIZE** command for details.

- 2 Read in the data:

**READ file "mydata" a b c**

This command will take the first column of your data, located in a file called "mydata," and place it in a vector called **a**. Similarly, it will place the second column in a vector called **b**, and the third column in a vector called **c**.

- 3 The vectors **a**, **b** and **c** can now be used in your RESAMPLING STATS program like any other vectors.

Any string of 8 alphanumeric characters (letters or digits) can be used to name variables (the same rule governs the naming of any vector in RESAMPLING STATS). Variable and vector names must begin with a non-digit. The underscore (**\_**) and dollar sign (**\$**) are also legal — you may wish to use the "**\$**" as a suffix for "pseudo-variables" (resampling proxies for observed or actual variables). Keywords for command options (e.g. percent in the **HISTOGRAM** command) may not be used to name variables. (**READ file "mydata" sam eloise gertrude; READ file "mydata" price1 price2 demand1**)

The program assumes that, when it comes to a space, tab or a comma, the next number in the row that it reads will be the value for the next variable. "Delimiting" one value from the next by having (1) one or more spaces (2) a "tab" entry or (3) a comma preceded and followed by zero or more spaces is the default setting; other approaches may be used (see below under "data format").

There is no fixed limit on the number of variables that can be accommodated, except that their names must be accommodated within one program line (160 characters).

## Data format

The “space/tab/comma-delimited” format described above is the default setting for the **READ** command. You can also use a “fixed” format option. Using as an example the previous data set, suppose it is on your file as follows:

**Table 13.**

---

212324.123	..
244321.321	..
315123.213	..
192998.132	..
. . .	
. . .	

---

The first two columns of characters belong to the first variable, the next 4 to the second variable, and the last 4 to the third variable. You can instruct RESAMPLING STATS to read it in correctly with the following commands:

**READ file “mydata” a %2 b %4 c %4**

The “%2” means RESAMPLING STATS will assume variable **a** is recorded with a field width of 2, variable **b** with a field width of 4, and variable **c** with a field width of 4. Entries must be right-justified; leading blanks are permitted.

This option is useful where you have stored the data in a file without intervening spaces, either to minimize storage space or to accommodate as many variables in a row as possible.

## Skipping columns

To skip columns of recorded data that you don’t want to use, assign them to a dummy vector as follows:

**READ file “mydata” a %5 dummy %2 c %5**

This reads the first 5 columns into the vector **a**, skips the next two columns of information (actually, it reads them into a vector, **dummy**, which you then ignore), then picks up again with the eighth column.

## Exporting data to a file

You can export vectors to an ASCII file with the command **WRITE**. For example:

**WRITE file "results" z**

This takes the vector **z** and writes it to a file "results" as a column of data. If there is more than one vector to be written, **WRITE** turns each vector into a column of data.

**WRITE file "results" z d e**

This takes the vector **z**, **d** and **e** and writes them to the file "results" as columns:

**Table 14.**

---

z1	d1	e1
z2	d2	e2
z3	d3	e3
.	.	.
.	.	.

---

The option **append** causes the data to be tacked on to the end of whatever is already in the file. If there are data in the file and the option **append** is not used, the existing data are erased.

**WRITE file "results" append z d e**

### Field width

Data being written to a file can be formatted in the same way that RESAMPLING STATS can read data being imported:

**WRITE file "results" a %5 b %2 c %5**

This writes the vectors **a**, **b** and **c** to the file "results," allotting a minimum of 5 columns to accommodate vector **a**, 2 columns for vector **b** and 5 columns for vector **c**. If the values being stored require more space, additional space will be allotted to those values, as necessary. A setting of **%0** is equivalent to a "wide as needed" format. The default field width is 12 columns.

## Additional options in data export

### Precision

In addition, you can specify the number of significant digits (or decimal places, depending on the style selected) to which the data will be carried and stored, and select the style in which it is stored.

**WRITE file “results” a %5.2 b %2.4 c %5.3**

This would cause vector **a** to be stored to 2 significant digits, vector **b** to 4 significant digits, and vector **c** to 3. The default setting is .6 — six significant digits or decimal places, depending on the style (see below). The maximum is .9. There is no space between the field width (**%5**) and the precision option (**.2**).

### Style

You can also specify the style in which data are to be stored:

**WRITE file “results” a %5.2f**

This would cause the vector **a** to be written to the file “results” in fixed notation to two decimal places. An explanation of this and other style options follows:

### Different styles of storing data

**Table 15.**

code	explanation
<b>F</b>	Fixed point notation, for which the precision option (see above) indicates the number of digits to the right of the decimal.
<b>E</b>	Scientific notation, for which “precision” indicates the number of significant digits. An exponent of the form E+99 is included. (Example: 5,280 would be recorded as 5.28E+03, meaning 5.28 times 10 <sup>3</sup> )
<b>G</b>	General (default) option. Uses F where possible, switching E where necessary to accommodate long numbers. G eliminates non-significant trailing zeroes (i.e., 2.00 would be written as 2).
<b>B</b>	Blank delimited: same as G style, but appends a blank to each field. (The blank does <i>not</i> count as part of the field width.)

- C** Comma: Same as B with a comma instead of a blank. You can ensure that the last number will not be followed by a comma by setting the last field width as “wide as necessary” (`%0`).
- 

The default setting is G.

---

## Summary

The general form of the **WRITE** command is:

**WRITE file “filename” a %5.2f b %12.6f c %10.3g**

This would write vectors **a**, **b** and **c** to the identified file. Vector **a** would be 5 columns wide, in fixed format with a precision of 2 digits to the right of the decimal point. Vector **b** would be 12 columns wide, also in fixed format and with precision of 6. Vector **c** would be 10 columns wide in general format with precision of 3. Don’t forget that vectors get written to columns:

**Vector a: (a1 a2 a3 a4)**

**Vector b: (b1 b2 b3 b4)**

Writing these two vectors to a file produces:

**a1 b1**

**a2 b2**

**a3 b3**

**a4 b4**

You can also use the option **missing** to specify how to **WRITE** records missing data; see below.

---

## Varying field widths

RESAMPLING STATS does not write asterisks or other characters to substitute for a number that will not fit within the width specification. Instead, it uses more columns for the large number and moves other fields to the right as needed for that particular value.