MATRIX DEFINITION

PURPOSE

Define a new matrix from a previously created matrix or a group of previously created variables.

DESCRIPTION

The columns of a matrix are identified by appending a column number to the matrix name (e.g., matrix C would have columns C1, C2, and so on). The matrix definition command allows you to take variables (C1, C2, and so on) and put them in a matrix.

SYNTAX 1

LET <mat> = MATRIX DEFINITION <var> <rows> <cols> <SUBSET/EXCEPT/FOR qualification>
where <var> is a is a variable that defines the first column of the matrix;
<rows> is a number or parameter that defines the number of rows to use in the variables (should be less than or equal to number of rows in <var>);
<cols> is a number or parameter that defines the variables to use (e.g., if <var> is A1 and <cols> is 3, then A1 and the two variables stored in memory after A1 should already exist and they will make up the columns of <mat>);
<mat> is a matrix where the resulting matrix is saved;
and where the <SUBSET/EXCEPT/FOR qualification> is optional and rarely used in this context.

This syntax copies the <cols> columns starting with <var> and from row 1 to row <rows>.

SYNTAX 2

LET <mat2> = MATRIX DEFINITION <mat1> <rows> <cols> <SUBSET/EXCEPT/FOR qualification>
where <mat1> is a matrix from which another matrix is created (starts at position (1,1) of <mat1>);
<rows> is a number or parameter that defines the number of rows to use in <mat1> (starts with row 1);
<cols> is a number or parameter that defines the number of columns to use in <mat1> (starts with column 1);
<mat2> is a matrix where the resulting matrix is saved;
and where the <SUBSET/EXCEPT/FOR qualification> is optional rarely used in this context.

This syntax copies the <cols> columns starting with column 1 of <mat1> and from row 1 to row <rows>.

SYNTAX 3

LET <mat> = MATRIX DEFINITION <var> <rows> <cols> <row1> <SUBSET/EXCEPT/FOR qualification>
where <var> is a is a variable that defines the first column of the matrix;
<rows> is a number or parameter that defines the number of rows to use in the variables (should be less than or equal to number of rows in <var>);
<cols> is a number or parameter that defines the variables to use (e.g., if <var> is A1 and <cols> is 3, then A1 and the two variables stored in memory after A1 should already exist and they will make up the columns of <mat>);
<row1> is a number or parameter that defines the first row of <var> to use;
<mat> is a matrix where the resulting matrix is saved;
and where the <SUBSET/EXCEPT/FOR qualification> is optional and rarely used in this context.

This syntax copies the <cols> columns starting with <var> (stored in the order the variables were created) and from row <row1> to row <rows>.

SYNTAX 4

LET <mat2> = MATRIX DEFINITION <mat1> <rows> <cols> <row1> <SUBSET/EXCEPT/FOR qualification>
where <mat1> is a matrix from which another matrix is created (starts at position (1,1) of <mat1>);
<rows> is a number or parameter that defines the last row of <mat1> to use;
<cols> is a number or parameter that defines the number of columns to use in <mat1>;
<row1> is a number or parameter that defines the first row of <mat1> to use;
<mat2> is a matrix where the resulting matrix is saved;
and where the <SUBSET/EXCEPT/FOR qualification> is optional and rarely used in this context.

This syntax copies from column 1 to column <cols> and from row <row1> to row <rows> and is useful for generating a partitioned matrix.

EXAMPLES

LET C = MATRIX DEFINITION FACT1 10 3; . FACT1 is a variable

```
LET C = MATRIX DEFINITION M 10 3; . M is a matrix
```

NOTE

One caution is that DATAPLOT assumes that the columns to be placed in the matrix are stored contiguously (variables are stored in the order that they are created). If this assumption is not valid, you can get unpredictable results. To get around this problem, you can do something like the following:

```
WRITE JUNK.DAT AGE SEX RACE
READ MATRIX JUNK.DAT X
```

These commands store X1 (AGE), X2 (SEX), and X3 (RACE) in contiguous positions in the matrix X. You can delete the original variables if you wish by entering the command DELETE AGE SEX RACE.

A second caution is that new columns are created for the defined matrix. That is, given that X1 through X7 already exist, the following command:

```
LET X = MATRIX DEFINTION X1 10 7
```

generates a new set of X1 to X7 variables that are distinct from the first. This is ambiguous in that DATAPLOT will have stored internally two sets of each of these variables with the same name. The solution is to use a different name on the left. For example,

```
LET XNEW = MATRIX DEFINITION X1 10 7
```

The new variables are given the names XNEW1 to XNEW7 which do not conflict with the previous names.

DEFAULT

None

SYNONYMS

None

RELATED COMMANDS

READ MATRIX	=	Read a matrix from a file or the terminal.
MATRIX SUBMATRIX	=	Define a matrix submatrix.
MATRIX AUGMENT	=	Append columns of one matrix onto another matrix.

APPLICATIONS

Linear Algebra

IMPLEMENTATION DATE

87/10

PROGRAM

```
LET X = DATA -99 1 0 -1
LET Y = DATA -99 0 1 0
. BE SURE VARIABLES ARE IN CONTIGUOUS COLUMNS
LET C1 = X**2 + Y**2; LET C2 = X
LET C3 = Y; LET C4 = 1 FOR I = 1 1 4
LET A = MATRIX DEFINITION C1 4 4
PRINT A
```

The following output is generated.

```
          MATRIX A       --            4 ROWS
                         --            4 COLUMNS

VARIABLES--A1             A2             A3             A4

  0.1960200E+05 -0.9900000E+02 -0.9900000E+02  0.1000000E+01
  0.1000000E+01  0.1000000E+01  0.0000000E+00  0.1000000E+01
  0.1000000E+01  0.0000000E+00  0.1000000E+01  0.1000000E+01
  0.1000000E+01 -0.1000000E+01  0.0000000E+00  0.1000000E+01
```