Data Area Editor

The Natural data area editor is used to create and modify a data area. A data area is a Natural object of the type global data area (GDA), local data area (LDA) or parameter data area (PDA). For information on using a data area, see *Data Areas* in the *Programming Guide*.

A data area contains data element definitions, such as user-defined variables, constants and database fields referenced with a data view in a data definition module (DDM), which can be used by one or more Natural objects. You can also create copycode from a data area. Note that data views from a DDM cannot be defined in PDAs.

Related Topic:

- *Editors in the SPoD Environment* in the *Unicode and Code Page Support* documentation

The *Data Area Editor* documentation covers the following topics:

- Invoking the Data Area Editor
- Edit Mode and Command Mode
- Editing Area
- Line Commands
- Editor Commands and Function Keys
- Storing and Cataloging a Data Area
- Help Information and Selection Options

Invoking the Data Area Editor

You invoke the data area editor with the Natural system command `EDIT` described in the *System Commands* documentation.

To invoke the data area editor for a new data area

- Issue the `EDIT` command specifying the type of data area (`GLOBAL`, `LOCAL` or `PARAMETER`) you want to create.

  For example:

  ```
  EDIT LOCAL
  ```

  An editor screen with an empty editing area appears for an LDA (indicated by `LOCAL` in the top information line of the screen) similar to the example shown in the following instructions.

To invoke the data area editor for an existing data area

- Issue the command EDIT specifying the name of a data area that has been stored as a source object in your current Natural environment.

  For example:

```
EDIT LDA1
```

  An editor screen similar to the example below appears which contains the source of the local data area LDA1:

```
                      Press <ESC> to enter command mode
 Mem: LDA1     Lib: SAGTEST  Type: LOCAL      Size:   1662  Line:    0 of:  36
C T    Comment
  *    *** Top of Data Area ***
  V  1 EMPLOYEES_VIEW                                EMPLOYEES
     2 PERSONNEL-ID                     A          8
  G  2 FULL-NAME
     3 FIRST-NAME                       A         20
     3 MIDDLE-I                         A          1
     3 NAME                             A         20
     2 MIDDLE-NAME                      A         20
     2 MAR-STAT                         A          1
     2 SEX                              A          1
     2 BIRTH                            D
     2 N@BIRTH                          I          2
  G  2 FULL-ADDRESS
  M  3 ADDRESS-LINE                     A         20 (1:191)
     3 CITY                             A         20
     3 ZIP                              A         10
     3 POST-CODE                        A         10
     3 COUNTRY                          A          3
  G  2 TELEPHONE
  F 1 HELP     F 2 CHOICE   F 3 QUIT     F 4 SAVE     F 5 STOW     F 6 CHECK
  F 7 READ     F 8 CLEAR    F 9 MEM TYPE F10 GEN      F11 FLD TYPE F12
```

The editor screen is organized in the following sections (from top to bottom):

Section	Explanation
Command line	Used to issue an editor or a system command or execute a program as described in *Edit Mode and Command Mode*. This line only appears if command mode is set. Otherwise, a message regarding command mode is displayed instead.
Information line	Contains the following information (from left to right) about the data area currently on the editor screen: **Mem:** The name of the data area or `empty` for a new data area that has not yet been saved as a source object with the `SAVE` or `STOW` system command. **Lib:** The library to which you are currently logged on. **Type:** The type of data area: `LOCAL`, `GLOBAL` or `PARAMETER`. The type can be changed by using the editor command `SET TYPE`. **Size:** The size (number of characters) of the current source. **Line:** The number of the current (highlighted) source line. **of:** The total number of lines contained in the source work area.
Editing area	Contains the source of a data area or appears empty for a new data area: see *Using the Editing Area*. You can only scroll in the source or modify the source if edit mode is set: see *Edit Mode and Command Mode*.
Function-key lines	Contains the function keys (F keys) available to execute an editor command: see *Editor Commands and Function Keys*.
Message line	Appears if an error occurs, in which case the message line temporarily overwrites the first function-key line with an appropriate error message.

Edit Mode and Command Mode

The data area editor operates in two different modes: edit mode and command mode.

In edit mode, you can scroll up or down in the source of the current data area, use the line commands required for creating or modifying source lines, and press all F keys available on the screen.

In command mode, you can enter or select an editor or a system command or execute a program. In command mode, you cannot modify the source of the data area.

By default, the data area editor is in edit mode when you invoke it.

To toggle between edit and command mode

- Press ESC.

  If command mode is set, the editor command line appears, which is indicated by **Command:** in the top left corner of the editor screen.

  The current mode is kept for the duration of the Natural session.

To enter a direct command or a program name

- In the command line, enter one of the following:
  - Any Natural system command.

    For example: The system command `CHECK` can be used for checking the syntax of source code and `SAVE` for saving source code (see also *Storing and Cataloging a Data Area*).

    For other system commands related to maintaining and using object sources, see *Editing and Storing Programming Objects* in the *System Commands* documentation.
  - The name of a Natural program to be executed.
  - An editor command. All editor commands available are described in *Editor Commands and Function Keys*.

To select a direct command from a menu

- In the command line, enter an `M` (menu).

  The following command menus appear:

  - **Commands**

    When you select this menu (selected by default), a window with a subset of most frequently used commands appears. You can select and execute one of the following commands: **Check**, **Clear**, **Fld-type**, **Gen**, **Read**, **Save**, **Type** or **Stow**. For explanations of these commands, see the corresponding direct commands described in *Editor Commands and Function Keys*.
  - **Direct Command Line**

    When you select this menu, all command menus are closed and the command line appears.
  - **Quit**

When you select this menu, the data area editor is terminated. Any changes made to the current data area since the last SAVE or STOW command are *not* saved.

Editing Area

The editing area is either empty or contains source code that was last read into the source work area with the command EDIT or READ as shown in the example in *Invoking the Data Area Editor*.

When you read in the source of an existing object, the entire source code is loaded into the source work area and is available for editing. However, depending on the size of the source, the editing area may not show all of the lines that belong to the source. In this case, you have to scroll down in the source to go to the line you want to view or modify as described in *Viewing and Selecting Source Lines*.

This section covers the following topics:

- Columns in the Editing Area
- Viewing and Selecting Source Lines
- Creating and Modifying Source Lines
- Input Fields in the Definition or Redefine Window

Columns in the Editing Area

The editing area of the editor screen is organized in columns where all attributes that belong to a variable or field definition are maintained in one line.

The columns contained in the editing area are described in the following section. The contents of the columns depend on the values entered in the **Definition** or **Redefine** window (see *Input Fields in the Definition or Redefine Window*) when creating or modifying a variable or field. The contents also depend on whether a counter field (C* variable) was created from a field.

Note:
A column heading can change or disappear depending on the type of variable or field contained in the current source line.

Column Heading	Explanation
C	The command column in which you can enter one of the line commands required to create or modify source lines. See also *Line Commands*.

Column Heading	Explanation
T	The type of variable or field. Possible types are: `B` **Block** A data block within a GDA. `C` **Constant** or **Counter Variable** A user-defined constant (not applicable to PDAs) or a counter field (C* variable). A counter field is used for a multiple-value field or a periodic group within a view (DDM). `F` **Data Field**: filler character. The filler bytes that can be denoted within a field or variable being redefined. `G` **Group** A group within a view (DDM). `M` **Multiple Field** A multiple-value field within a view (DDM). `O` **Object Handle** The handle of an object. `P` **Periodic Group** A periodic group within a view (DDM). `R` **Redefined Field** The redefinition of a variable or field. `V` Not applicable to PDAs. **View** A view definition created from a DDM. *blank* **Data Field** A user-defined variable or field, or a group structure (not within a view). `*` **Comment**. A comment field.

Column Heading	Explanation
L	The level number of the variable or field (1 - 99). Variables which are not within a hierarchical structure and view definitions must be assigned level 1. Level numbers cannot be used with data block definitions.
Name of *variable-field-type* or **Comment**	The name of the variable or field. This column heading changes according to the type of variable or field currently selected as indicated in the description of column **T**. For a view, in addition to **Name of *variable-field-type***, the **Name of DDM** column appears. For a block, in addition to **Name of *variable-field-type***, the **Name of Parent Block** column appears.
F	The Natural data format of the variable or field.
Length	The length of the variable or field.
Index/Comment	The array indices and/or comment of the variable or field.
M	The **M** (Miscellaneous) column contains an X if an edit mask, header, and/or initial value is defined for a variable or field. This column does not appear for a view, or a group, periodic group or multiple-value field within a view.

Viewing and Selecting Source Lines

In edit mode, you can scroll up or down in the current data area to view all lines of the source currently contained in the source work area, select a line for modification, or position at the line where you want to insert new lines.

To scroll in a source and select a line

- Scroll up or down one line with UP ARROW and DOWN ARROW respectively.

  Or:
  Press HOME or END to access the corresponding first (topmost) line or the last (bottommost) line contained in the source work area.

  Or:
  Move the current source line up or down one screen page at a time with PAGE UP or PAGE DOWN respectively. Check your file *SAGtermcap* for entries if you have no access to any of these keys.

  The line in which the cursor is positioned is the current line, which is selected (highlighted) and can be modified.

Creating and Modifying Source Lines

In edit mode, you can create or modify the variable or field definition in the current source line by using the appropriate line command described in *Line Commands*.

The following are example instructions for using a line command to add single or multiple lines to a source or modify a line.

To add a line for a new definition other than a view, redefinition or counter field

1. In the **C** column, next to the selected line below which you want to place the new definition, type in the following line command:

   ```
   I
   ```

   If the data area is empty, enter the `I` next to **Top of Data Area**.

   A selection window appears.

2. Select one of the following:

Data Field:	For a user-defined variable or a database field.
Block:	For a data block within a GDA described in the *Programming Guide*.
Constant:	For a user-defined constant described in *User-Defined Constants* in the *Programming Guide*.
Handle:	For an object handle described in *Using Classes and Objects* in the *Programming Guide*.
Structure:	For a hierarchical group structure (not within a view).
Comment:	For a comment field.

   A **Definition** window appears for the selected type of variable or field in which you can enter the required attribute definitions. For explanations of the input fields contained in this window, see *Input Fields in the Definition or Redefine Window*.

To add lines for a view definition

1. In the **C** column, next to the selected line below which you want to place the view, type in the following line command:

   ```
   V
   ```

   If the data area is empty, enter `V` next to **Top of Data Area**.

   A **View Definition** window appears.

2. Enter the name of the view to be created and the DDM from which to create the view. The specified DDM must be contained in the current Natural library in the current system file.

   A **DDM** selection window appears with a list of all fields defined in the specified DDM.

3. Scroll through the list with the arrow (cursor) keys and enter an `X` next to each DDM field you want to select for the view.

Or:
In the **DDM** selection window, enter an A next to any of the DDM fields to select all DDM fields for the view.

You can deselect a field by replacing the X with a blank character.

If no periodic group or multiple-value field is selected, all fields are copied into the data area.

If you selected a periodic group or multiple-value field, an **Occurrences Definition** window appears.

4. In the **From:** field(s), enter the lower bounds of the one- or two-dimensional array to be used in the view, and, in the **To:** field(s), enter the upper bounds.

   The periodic group or multiple-value field is copied into the data area with the specified occurrences as subordinate fields of a hierarchical view definition structure where the view name is at level 1.

To modify the contents of a line

- In the **C** column, next to the selected line, type in the following line command:

  ```
  E
  ```

  The **Definition** window (or **Redefine** window for a redefinition) appears for the variable or field definition contained in the selected line.

Input Fields in the Definition or Redefine Window

You create and modify a variable or field in the **Definition** window or the **Redefine** window for a redefinition. The input fields contained in this window depend on the type of variable or field selected. The table below lists and describes all possible input fields.

The definitions you enter in the **Definition** or **Redefine** window are checked for syntax errors.

For explanations of the variable or field attributes mentioned in this section, see also *User-Defined Variables* and *Defining Fields* in the *Programming Guide* and `DEFINE DATA` in the *Statements* documentation. The values used in the `DEFINE DATA` statement correspond to the values used in a data area.

Input Field	Explanation
Level	The level number (1 - 99) of the variable or field. Variables or fields which are not within a hierarchical structure must be assigned level 1. View definitions must be assigned level 1. Level numbers cannot be used with data block definitions.

Input Field	Explanation
Name	The name of one of the following: • User-defined variable or database field. • User-defined constant. • Structure (group definition, not within a view). • Handle of object. • Block and parent block (GDAs only). • View (DDM). • Multiple-value field (within a view). • Periodic group (within a view). • Counter field (C* variable) for a multiple-value field or a periodic group within a view. For valid names, see *Naming Conventions for User-Defined Variables* in the *Using Natural* documentation. For a user-defined constant, see also `CONSTANT` in the *Statements* documentation.
Format	The Natural data format of the variable or field. For valid formats, see *Format and Length of User-Defined Variables* and *Special Formats* in the *Programming Guide*. For a counter field (C* variable), you can specify the Natural data format/length I2 or I4 (the default setting is N3 for no format/length).
Length	The length of the variable or field. For valid lengths, see *Format and Length of User-Defined Variables* in the *Programming Guide*. No length is permitted for the Natural data formats C, D, T and L. You can define dynamic variables by specifying `DYNAMIC` in the **Length** field. For a counter field (C* variable), you can specify the Natural data format/length I2 or I4 (the default setting is N3 for no format/length).
Arraydefinition	The array definition of the variable or field. You can define the upper and lower bounds of a one- or two-dimensional array as demonstrated in *Examples of Array Definitions*.

Input Field	Explanation
Edit Mask	Not applicable to PDAs. The edit mask of the variable or field to be used when the variable or field is displayed with an I/O statement. Enter a valid value without using apostrophes or parentheses as shown in *Examples of Edit Mask Definitions*. For valid input values, see the corresponding session parameter `EM` described in the *Parameter Reference* documentation.
Header Definition	Not applicable to PDAs. The header to be produced for the variable or field in a `DISPLAY` statement. Enter any alphanumeric character string without using apostrophes or parentheses as shown in *Example of a Header Definition*. For further information on headers, see the corresponding session parameter `HD` described in the *Parameter Reference* documentation.

Input Field	Explanation
Initialization	Not applicable to PDAs. The initial value assigned to a variable or field or the array occurrence(s) defined for a variable or field. You can specify one of the following initialization modes: `F` Free form mode. If you enter `F`, the **Free Form Initialization** window appears, in which you can enter your initial value definitions according to the common Natural syntax definitions in a `DEFINE DATA` statement. You can assign the same initial value to a whole range of field occurrences at a time. See also *Examples of Initial Value Assignments in Free Mode*. For detailed information on defining initial values, see *Initial-Value Definition Initial/Constant Values for an Array* in the *Statements* documentation. `S` Single-value mode. If you enter `S`, the **Single Value Initialization** window appears, in which you can enter a single initial value rather than an initialization clause. Values are entered based on the variable or field type. You only enter the required variable or field value; any further specifications necessary (including apostrophes for alphanumeric variables or fields, or value prefixes such as `H` for hexadecimal, `D` for date and `T` for time) are generated automatically. For example, to specify an initial value of H'F1F2' for a binary variable (B2), enter `F1F2`. The data editor generates: `INIT <H'F1F2'>`. If the variable or field is an array, all elements of the array are listed. A value for each element can be entered (optional). **Caution:** Changing the initialization type deletes all previously entered initialization information. See also the sections *Initial Values (and the RESET Statement)* and *Initial Values for Arrays* in the *Programming Guide*.

Input Field	Explanation
Value Clause	Only applies to PDAs. Determines the way in which the value of a variable or field specified as a parameter in a CALLNAT statement is passed from a program to an invoked object (for example, a subprogram). Valid input values are: *blank* — Indicates the parameter specification call-by-reference (default). V — Indicates the parameter specification call-by-value. R — Indicates the parameter specification call-by-value-result. For detailed information, see the corresponding options BY VALUE and BY VALUE RESULT described for the DEFINE DATA statement in *Parameter Data Definition*, and *operand2* described for the CALLNAT statement in the *Statements* documentation.
Optional Param	Only applies to PDAs. Determines whether the variable or field value is passed as a parameter according to the setting of **Value Clause** (see above): N A parameter must be passed (default). Y A parameter *can* be passed. For detailed information, see the corresponding option OPTIONAL described for the DEFINE DATA statement in *Parameter Data Definition* and *operand2* described for the CALLNAT statement in the *Statements* documentation.
Indexdefinition	The array definition of a variable of the type structure (group). This input field can be used to define the upper and lower bounds for an array, to supply initial values for a variable or to supply an edit mask for a variable. See also *Examples of Array Definitions*.
Comment	Commentary text.

Examples of Array Definitions:

```
(2,2)    /* 2 dimensions, 2 occurrences
(2,2,2)  /* 3 dimensions, 2 occurrences
(1:10,2)
(-1:3,2)
```

Examples of Edit Mask Definitions:

```
999.99
XXX...XX
MM.DD.YY
```

Example of a Header Definition:

```
HEADER TEXT
```

Examples of Initial Value Assignments in Free Mode:

```
INIT<3>
INIT<'ABC'>
INIT<H'F1F2'> /* binary variable (B2)
CONST<12>
INIT ALL<'ABC'>
```

Line Commands

You enter a line command in the **C** column of a source line when edit mode is set. The command entered may not be indicated in the column.

For a selection list of all available line commands, press F2.

All line commands provided by the data area editor are described in the following table. The expression "edit block" used in the table, denotes a marked block of source lines.

Command	Function
C (Copy)	Copies one or more lines. If C is entered within an edit block, all lines of this block are copied. See also *To cut/copy and paste an edit block.*
D (Delete)	Deletes one or more lines. If D is entered within an edit block, all lines within this block are deleted. For additional information, see the line commands C (Copy) and P (Paste). See also *To cut/copy and paste an edit block.*
E (Edit)	Depending on the variable or field type selected, opens the **Definition** or **Redefine** window (see also *Input Fields in the Definition or Redefine Window*), in which you can modify the attributes of the variable or field contained in this line. For a field within a view definition, you can only modify the following: **Level** (except periodic groups and multiple-value fields), **Edit Mask**, **Header Definition**, **Indexdefinition** and **Comment**.

Command	Function
H (Unmark block)	Unmarks the current edit block. This command must be issued from within the edit block.
I (Insert line)	Inserts a line for a new definition as described in *To add a line for a new definition other than a view, redefinition or counter field*.
M (Add comment)	Adds comment mark(s) to the selected line or the marked edit block. These lines are ignored by a compiler check. This may be useful for testing purposes.
N (Remove comment)	Removes comment mark(s) from the selected line or the marked edit block.
P (Paste)	Pastes one or more lines into the data area after the current line. See also *To cut/copy and paste an edit block.*

Command	Function
R (Redefine)	Redefines the variable or field contained in this line as single variable or a group of variables. A window appears in which you can select one of the following redefine options: **Data Field** Redefines the variable or field as a single user-defined variable. **Filler** Redefines the variable or field using the filler option (*n*X). With the filler option, *n* filler bytes can be denoted within the variable or field being redefined, where *n* can be up to 10 digits (1 GB). The definition of trailing filler bytes is optional. **Structure** Redefines the variable or field as a structure (group). **Comment** Redefines the variable or field as a comment. Depending on the redefine option selected, either a **Redefine** or a **Definition** window appears, in which you can enter the definitions required (see also *Input Fields in the Definition or Redefine Window*). The data area editor keeps track of the number of free bytes still available for the redefinition. If there are no free bytes, the redefine function ends. Depending on the redefine option used, the data area editor automatically adds the lines required for the redefinition above the variable or field that is redefined.
S (Show)	Displays the **Definition** or **Redefine** window which contains all definitions of the variable or field contained in this line (see also *Input Fields in the Definition or Redefine Window*). Changes are not possible. If an initialization has been specified, a separate window containing initialization information is also displayed. Information scrolling is possible.
V (Insert view)	Inserts a view definition as described in *To add lines for a view definition*.
X (Mark block start)	Marks the beginning of an edit block.

Command	Function
Y (Mark block end)	Marks the end of an edit block.
Z (Mark group)	Marks an entire group structure as an edit block. The edit block starts at the current line and includes all consecutive lines with levels less than the current line. For example, if Z is entered in a view line, the entire view is marked as an edit block.
* (Generate counter)	Not applicable to PDAs. Generates a counter field (C* variable) from the multiple-value field or periodic group contained in this line. The counter field is placed in the line above which you entered the command. A counter field is used to retrieve the number of occurrences of a multiple-value field or a period group. See also *Referencing the Internal Count for a Database Array (C* Notation)* in the *Programming Guide*.

The following are instructions for using line commands to move or copy a block of lines or a single line within the source.

To cut/copy and paste an edit block

1. Next to the first line of the block of lines (or the single line) to be cut or copied, enter the following line command:

   ```
   X
   ```

   The line is marked.

2. Scroll up or down to the last line of the block of lines and enter the following line command:

   ```
   Y
   ```

   For a single line, you enter the line command Y in the same line, in which you entered the line command X.

   All lines that belong to the block are marked (highlighted) thus representing the edit block.

3. Within the edit block, enter one of the following line commands:

   ```
   C
   ```

to copy the line(s), or

```
D
```

to delete (cut) the line(s).

4. Position the cursor in the line below which you want to paste the line(s) and enter the following line command:

```
P
```

The line(s) are pasted into the source.

5. You can unmark the edit block by entering the following line command within the edit block:

```
H
```

The edit block is unmarked and only the current line is highlighted.

Editor Commands and Function Keys

This section describes the editor commands and editor-specific system commands that can be entered in the command line or selected from the **Commands** menu in command mode.

In addition, this section describes alternative F keys that can be used in edit mode. In command mode, you can only use F1.

For explanations of the syntax symbols used in the editor commands, refer to *System Command Syntax* in the *System Commands* documentation.

Command	F Key	Function
n/a	F1	Provides help information on editor features. See also *Help Information and Selection Options*. **Note:** When you enter `HELP` you will invoke the help function for error messages as described for the system command `HELP` in the *System Commands* documentation.
n/a	F2	Invokes a selection window with valid input values for the input field in which the cursor is positioned.

Command	F Key	Function
CATALOG [*object-name*]	n/a	Executes the system command CATALOG which checks and catalogs the current data area definition. You must supply an object name with the command if you catalog a new data area definition or if you want to copy the current data area. Otherwise, an appropriate message appears. See also *Storing and Cataloging a Data Area*.
CHECK	F6	Executes the system command CHECK which checks the syntax of the current data area definition. A window informs you that a syntax check is in process. If a syntax error is found, the line containing the error becomes the current line, and the error is displayed in the message line. If no errors are found, a corresponding message is displayed.
CLEAR	F8	Executes the system command CLEAR which clears the source work area. Changes to the data area currently contained in the source work area are lost if they were not previously saved.
FLD TYPE	F11	FLD TYPE is available from the **Commands** menu only (you cannot enter it in the command line). Invokes a window in which you can change the type of the current variable or comment. You can select one of the following types: **D** Data field **B** Block **C** Constant **H** Handle **S** Structure ***** Comment The type of a field within a view definition cannot be changed. **Caution:** When changing types, some attribute definitions of a variable or comment can be lost.

Command	F Key	Function
GENERATE *object-name*	F10	Generates a Natural object of the type copycode from the current data area definition thus overwriting the contents of the source work area. Changes made to the data area since the last SAVE or STOW are lost. The copycode object is stored as a source object in the specified Natural library in the current system file under the *object-name* supplied with the command.
QUIT or . (a period)	F3	Terminates the data area editor. Any changes made since the last SAVE or STOW command are lost.
READ *object-name*	F7	Executes the system command READ which reads an existing data area definition into the source work area. The data area must be stored as a source object with the specified *object-name* in the current Natural library in the current system file.
SAVE [*object-name*]	F4	Executes the system command SAVE which saves the current data area definition. You must supply an object name if you save a new data area definition or if you want to copy the current data area. Otherwise, an appropriate message appears or input window appears. If you press F4, a window always prompts you for name of the data area and the library. If the current object and library names are correct, no entry is required. See also *Storing and Cataloging a Data Area*.

Command	F Key	Function
SCAN *scan-value*	n/a	Scans the data area for a character string (*scan-value*) in the **Name of *variable-field-type*** column (default) and/or the **M** column of the editor screen, depending on whether the SET SCAN command was executed earlier. The line in which the *scan-value* is found is highlighted. If the first instance of *scan-value* is highlighted, you can go to the next instance by pressing ENTER. The scan is performed from the first to the last line in the source work area and wraps around to the beginning after the last line is reached. **Note:** The SCAN command performs an exact search for the specified *scan-value*. This should be taken into account when searching for DBCS (Double Byte Character Set) characters.
SET ABS [ON|OFF]	n/a	Determines whether the SCAN command operates in absolute or non-absolute mode. ON The SCAN command operates in absolute mode, which means that the value to be scanned needs not be delimited by blanks or special characters. OFF The SCAN command operates in non-absolute mode, which means that the value to be scanned must be delimited by blanks or special characters. The default is OFF.

Command	F Key	Function
`SET SCAN [COMMENT|NAME|LEVEL]`	n/a	Determines the column(s) in which the `SCAN` command searches for a *`scan-value`*: `COMMENT` The **Index/Comment** column is scanned. `NAME` The **Name of *variable-field-type*** (or **Comment**) column is scanned. `LEVEL` Scans within a hierarchical group structure. The default is `NAME`.
`STOW [object-name]`	F5	Executes the system command `STOW` which saves and catalogs the current data area definition. You must supply an object name if you `STOW` a new data area or if you want to copy the current data area. See also *Storing and Cataloging a Data Area*.
`SET TYPE G|L|A`	F9	Changes the type of the current data area: `G` Global data area `L` Local data area `A` Parameter data area

Storing and Cataloging a Data Area

Before a data area can be used in a Natural program (or another object), the data area must be saved as a source object and/or a cataloged object in the current Natural environment.

The commands used for saving and cataloging the current data area definition are described in the following section.

You must supply an object name if you save or catalog a new data area definition or if you want to copy the current data area. Otherwise, an appropriate message appears or a window prompts you for the name. For the naming conventions that apply when saving or cataloging a data area, refer to *Object Naming*

Conventions in the *Using Natural* documentation.

To save a data area as a source object

- Enter the system command SAVE according to the syntax rules described in the *System Commands* documentation.

  Or:
  Press F4.

  Or:
  From the **Commands** menu, choose **Save**.

  The source of the data area is stored as a source object in the specified Natural library in the current system file. The data area is not checked for syntax errors.

To save a data area as a cataloged object

- Enter the system command CATALOG according to the syntax rules described in the *System Commands* documentation.

  The source of the data area is checked for syntax errors. If no errors are found, it is saved as a cataloged object in the specified Natural library in the current system file.

To save a data area as a source object *and* a cataloged object

- Enter the system command STOW according to the syntax rules described in the *System Commands* documentation.

  Or:
  Press F5.

  Or:
  From the **Commands** menu, choose **Stow**.

  The source of the data area is checked for syntax errors. If no errors are found, it is saved as a source object and a cataloged object in the specified Natural library in the current system file.

Source Format for Data Area Storage

The data area editor uses an internal source format to store the sources of data areas in the FUSER system file. The following new features and definitions that are available from Natural Version 6.3 onwards require that the data area source is stored in the FUSER system file using an extended source format:

Levels 10 to 99
Large arrays
X-arrays
`INIT` clause for handles in `DEFINE DATA`
Dynamic variables
Large variables
`INIT` clause for dynamic variables in `DEFINE DATA`
Arrays for dynamic variables
Comment lines of up to 77 characters

Exception: Large arrays, X-arrays, large variables and dynamic variables can be read in both source formats depending on their size. If they exceed a specific length (for example, variables with more than four digits), the extended source format is used. See also *Format and Length of User-Defined Variables* in the *Programming Guide*.

Data areas that are stored using the extended source format cannot be used or edited with Natural Version 6.2 where a different source format was used. The data area editor of Natural Version 6.3 and above supports the Natural Version 6.2 format *and* the extended source format. The editor can read both formats and converts the Natural Version 6.2 format to the extended source format. As long as no Natural Version 6.3 (and above) features or definitions are used, data areas are stored in the Natural Version 6.2 format by default. This format guarantees compatibility between data area sources stored in a Natural Version 6.2 and a Natural Version 6.3 (and above) environment.

Help Information and Selection Options

You can use the help system to obtain help information on editor functions and input fields. In addition, in edit mode, you can use the help system to choose a valid input value or a line command from a list.

To display help information

1. In edit mode, position the cursor in the **C** column and press F1 for a summary of all editor functions available.

   Or:
   Place the cursor on the field about which you require further information and press F1 for instructions on using this field.

   A window appears with help information.

   You can use the arrow (cursor) keys to scroll up or down in the window.

2. Choose ENTER or ESC to close the help window.

To select a valid value or appropriate line command

- Place the cursor on the field for which you want to select a valid input value or execute a line command and press F2.

  If applicable, a selection window appears from which you can select an input value or a line command.