Natural

System Commands

Version 9.3.3

October 2025

ADABAS & NATURAL

This document applies to Natural Version 9.3.3 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1992-2025 Software GmbH, Darmstadt, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software GmbH product names are either trademarks or registered trademarks of Software GmbH and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software GmbH and/or its subsidiaries is located at https://softwareag.com/licenses.

Use of this software is subject to adherence to Software GmbH's licensing conditions and terms. These terms are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software GmbH Products / Copyright and Trademark Notices of Software GmbH Products". These documents are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software GmbH.

Document ID: NATUX-NNATSYSCOM-933-20251030

Table of Contents

Preface vii
1 About this Documentation 1
Document Conventions 2
Online Information and Support 2
Data Protection 3
2 Issuing System Commands 5
Command Input 6
Command Line 6
NEXT Prompt 6
MORE Prompt 7
3 System Command Syntax 9
Syntax Elements 10
Example of Command Syntax 11
4 System Commands Grouped by Category 13
Navigating in Natural 14
Natural Development Environment 14
Managing Applications with Natural Objects 15
Monitoring, Debugging and Tracing 16
Natural with Other Products 16
5 CATALL 19
CATALL in Interactive Mode 20
CATALL in Batch Mode 21
6 CATALOG 23
7 CHECK 25
8 CLEAR 27
9 COMPOPT 29
COMPOPT Syntax Explanation 30
Compiler Options 30
Specifying Compiler Parameters 30
10 DEBUG 33
11 EDIT 35
EDIT Syntax 1 36
EDIT Syntax 2 38
EDIT Syntax 3 38
12 EXECUTE 39
EXECUTE Syntax Explanation 40
Examples of EXECUTE Command 41
13 FIN 43
14 GLOBALS 45
GLOBALS Syntax Explanation 46
GLOBALS Parameters 46
Interaction with SET GLOBALS and Other Statements 47
15 HELP 49

16 INPL 51
17 KEY 53
Assigning Commands 54
Activating/Deactivating All Keys - KEY ON/OFF 55
Activating/Deactivating Individual Keys - KEY key=ON/OFF 55
18 LAST 57
19 LASTMSG 59
20 LIST 61
LIST Syntax Overview 62
Displaying an Individual Source 63
Displaying a List of Objects 64
Displaying Numbers and Sizes of Objects 65
Displaying Directory Information 66
Displaying Views 66
Displaying File Information of Resource Objects 67
Displaying File Information of Error Message Containers 67
21 LIST XREF 69
22 LOGOFF 71
23 LOGON 73
24 MAIL 75
25 PROFILE 77
26 PROFILER 79
27 PURGE 81
28 READ 83
29 RENAME 85
30 RENUMBER 87
31 RETURN 89
32 RPCERR 91
33 RUN 93
34 SAVE 95
35 SCAN 97
SCAN Menu Options 98
SCAN Subcommands 99
SCAN under Natural Security 100
36 SCRATCH 101
37 SETUP 103
SETUP Syntax Explanation 104
SETUP/RETURN Example 105
38 STOW 107
39 STRUCT 109
Indentation of Source Code Lines 110
40 SYSDDM 113
41 SYSERR 115
42 SYSEXT 117
43 SYSEXV 119

44 SYSFILE 121
45 SYSLVERS 123
46 SYSMAIN 125
47 SYSNCP 127
48 SYSOBJH 129
49 SYSPROD 131
About SYSPROD 132
50 SYSPROF 133
51 SYSRPC 135
52 SYSWIZDB 137
53 SYSWIZDW 139
54 TECH 141
55 UNCATALOG 143
56 UPDATE 145
57 WHICH 147
Direct Command Syntax 148
WHICH Parameter Description 148
Default Output of WHICH Command 148
Full List of WHICH Command 149
Batch Processing 150
WHICH Command under Natural Security 150
58 XREF 151

Preface

This documentation describes the Natural system commands. It is organized under the following headings:

Issuing System Commands	Describes the general rules that apply when you enter a Natural system command.
System Command Syntax	Explains the symbols that are used within the syntax descriptions of Natural system commands.
System Commands Grouped by Category	Provides an overview of the Natural system commands grouped by category.
System Commands in Alphabetical Order	Descriptions of the system commands in alphabetical order.

1 About this Documentation

- Document Conventions 2
- Online Information and Support 2
- Data Protection 3

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

Product Training

You can find helpful product training material on our Learning Portal at **https://learn.softwareag.com**.

Tech Community

You can collaborate with Software GmbH experts on our Tech Community website at **https://techcommunity.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software GmbH news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://hub.docker.com/publishers/softwareag** and discover additional Software GmbH resources.

Product Support

Support for Software GmbH products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Issuing System Commands

- Command Input 6
- Command Line 6
- NEXT Prompt 6
- MORE Prompt 7

Natural system commands perform functions you need to create, maintain or execute Natural objects. In addition, Natural system commands are used to monitor and administer your Natural environment.

Notation *vrs* or *vr*

When used in this documentation, the notation *vrs* or *vr* represents the relevant product version (see also *Version* in the *Glossary*).

Command Input

You can issue a system command by entering it in one of the following ways:

- In the **command line**;
- At the Natural NEXT or MORE prompt.

The following rules apply:

- Command input is not case-sensitive.
- Commands are context-sensitive.
- Some Natural commands affect objects other than the currently active object.

For an explanation of the symbols that are used within the syntax descriptions, see ***System Command Syntax***.

Command Line

If you choose the **Direct** menu from the main menu, the **Direct Command** window is displayed in which you enter the system command.

NEXT Prompt

The NEXT prompt appears in a Natural application or program when no more output is pending.

MORE Prompt

The MORE prompt is displayed at the bottom of an output screen to signal that more output is pending. When a system command is entered in response to a MORE prompt, program execution is interrupted and the system command is executed.

3 System Command Syntax

- Syntax Elements 10
- Example of Command Syntax 11

Syntax Elements

The following symbols are used within the syntax descriptions of system commands:

Element	Explanation
ABCDEF	Upper-case non-italic letters indicate that the term is either a Natural keyword or a Natural reserved word that must be entered exactly as specified.
ABCDEF	If an optional term in upper-case letters is completely underlined (not a hyperlink!), this indicates that the term is the default value. If you omit the term, the underlined value applies.
ABCDEF	If a term in upper-case letters is partially underlined (not a hyperlink!), this indicates that the underlined portion is an acceptable abbreviation of the term.
abcdef	Letters in italics are used to represent variable information. You must supply a valid value when specifying this term.
[ ]	Elements contained within square brackets are optional. If the square brackets contain several lines stacked one above the other, each line is an optional alternative. You may choose at most one of the alternatives.
{ }	If the braces contain several lines stacked one above the other, each line is an alternative. You must choose exactly one of the alternatives.
|	The vertical bar separates alternatives.
...	A term preceding an ellipsis may optionally be repeated. A number after the ellipsis indicates how many times the term may be repeated. If the term preceding the ellipsis is an expression enclosed in square brackets or braces, the ellipsis applies to the entire bracketed expression.
,...	A term preceding a comma-ellipsis may optionally be repeated; if it is repeated, the repetitions must be separated by commas. A number after the comma-ellipsis indicates how many times the term may be repeated. If the term preceding the comma-ellipsis is an expression enclosed in square brackets or braces, the comma-ellipsis applies to the entire bracketed expression.
:...	A term preceding a colon-ellipsis may optionally be repeated; if it is repeated, the repetitions must be separated by colons. A number after the colon-ellipsis indicates how many times the term may be repeated. If the term preceding the colon-ellipsis is an expression enclosed in square brackets or braces, the colon-ellipsis applies to the entire bracketed expression.
Other symbols (except [ ] { } | ... ,... :...)	All other symbols except those defined in this table must be entered exactly as specified. **Exception:** The SQL scalar concatenation operator is represented by two vertical bars that must be entered literally as they appear in the syntax definition.

Example of Command Syntax

```
CATALOG [object-name [library-id]]
```

- `CATALOG` is a Natural keyword which you must enter as specified. The underlining indicates that you may also enter it in abbreviated form as `CAT`.
- *`object-name`* and *`library-id`* are user-supplied operands for which you specify the name of the program you wish to deal with and the ID of the library in which that program is contained.
- The square brackets indicate that *`object-name`* and *`library-id`* are optional elements which you can, but need not, specify. The grouping of the brackets indicate that you can specify `CATALOG` alone, or `CATALOG` followed either by a program name only or by a program name and a library ID; however, you cannot specify a library ID if you do not also specify a program name.

4 System Commands Grouped by Category

- Navigating in Natural 14
- Natural Development Environment 14
- Managing Applications with Natural Objects 15
- Monitoring, Debugging and Tracing 16
- Natural with Other Products 16

This chapter is a summary of Natural system commands grouped by category.

Navigating in Natural

Command	Brief Description
FIN	Terminates a Natural session.
LAST	Displays the system commands that were last executed, and allows you to execute them again.
LOGOFF	Causes the library ID to be set to SYSTEM and the Adabas password to be set to blanks. The contents of the source program work area are not affected by this command.
LOGON	Establishes a library ID for the user. In the specified library, all source or object programs saved during the session will be stored (unless you explicitly specify another library ID in a SAVE, CATALOG or STOW command).
RETURN	Returns to a return point set by a SETUP command.
SETUP	Establishes a return point to which control can be returned using a RETURN command. This allows you to easily transfer from one application to another during a Natural session.

Natural Development Environment

Command	Brief Description
COMPOPT	Sets compilation options that affect the way in which Natural objects are compiled.
GLOBALS	Changes the settings of Natural session parameters.
HELP	Invokes the Natural help system.
KEY	Assigns functions to keys to be used in your Natural session.
LAST	Displays the system commands that were last executed, and allows you to execute them again.
LIST XREF	Displays all active cross-reference data for the current library. (Only available if Predict is installed.)
SYSEXT	Invokes the SYSEXT utility with Natural application programming interfaces.
SYSEXV	Invokes the SYSEXV utility with examples of the new features of the current Natural versions and debugging hints.
SYSFILE	Displays work and print files information.
SYSPROD	Displays a list of the products installed at your site, and information on these products.
SYSPROF	Displays the current definitions of the Natural system files.
UPDATE	Prevents database updating being carried out by a program.

Command	Brief Description
XREF	Controls the use of the Predict function "active cross-references". This function automatically creates documentation in Predict about the objects which a program/data area references. (Only available if Predict is installed.)

Managing Applications with Natural Objects

Command	Brief Description
CATALL	Catalogs (compiles) *all* objects or selected objects in the current library.
CATALOG	Catalogs (compiles) the source code currently in the editor work area, and if the syntax has been found to be correct, stores the resulting cataloged object in the Natural system file.
CHECK	Checks that the source code currently in the editor work area does not contain any syntax errors. Syntax checking is also performed as part of the system commands RUN, CATALL, CATALOG and STOW.
CLEAR	Clears the contents of the editor work area.
EDIT	Opens an editor to create or modify source code.
EXECUTE	Executes a cataloged (compiled) program that has been stored as a cataloged object in the Natural system file.
INPL	Invokes the INPL utility. It is *only* used for the loading of installation data sets into the system files.
LIST	Lists one or more objects contained in the current library or the contents of the editor work area.
PURGE	Deletes one or more source objects from the current library.
READ	Transfers a source object from the Natural system file to the editor work area.
RENAME	Changes the name of an object or the name and the type of an object.
RENUMBER	Renumbers the source code currently in the editor work area.
RUN	Compiles and executes the source program currently in the work area of the editor.
SAVE	Stores the source code currently in the editor work area as a source object in the Natural system file.
SCAN	Searches for a character string within a source with the option to replace the string.
SCRATCH	Deletes a source object and/or the corresponding cataloged object from the current library.
STOW	Catalogs (compiles) and stores source code as both a source object and a cataloged object in the current Natural system file.
STRUCT	Performs structural indentation of a program source and helps detecting structural inconsistencies.
SYSDDM	Creates and maintains Natural data definition modules (DDMs).

Command	Brief Description
SYSERR	Creates and maintains the messages you wish your Natural applications to display to the users.
SYSLVERS	List objects which have been cataloged within a selected Natural version range.
SYSMAIN	Transfers Natural objects within the Natural system from one library to another.
SYSNCP	Creates and maintains the command processors to be used in your Natural applications.
SYSOBJH	Processes Natural and non-Natural objects for distribution in Natural environments.
SYSRPC	Invokes the SYSRPC utility to create and maintain remote procedure calls, that is, provides the settings necessary to execute a subprogram located on a remote server.
SYSWIZDB	Invokes the Natural Data Browser, a development tool wizard within Natural Studio. It enables you to display and print or store file structures.
SYSWIZDW	Invokes the Natural Dialog Wizard, a tool for creating dialogs for specific purposes. The defined dialogs can have several layouts that adapt to desired requirements.
UNCATALOG	Deletes one or more cataloged objects in the current library.
WHICH	Lists objects in the libraries used at run time in the Library Search Order.

Monitoring, Debugging and Tracing

Command	Brief Description
DEBUG	This command is used to invoke the Natural debugger.
HELP	Invokes the Natural help system.
LASTMSG	Displays additional information on the error situation which occurred last.
RPCERR	Displays the last Natural error number and message if related to Natural RPC (Remote Procedure Call), and the last EntireX Broker reason code and associated message.
TECH	Displays technical and other information on your Natural session.

Natural with Other Products

The following system commands are only available in connection with other products installed at your site:

Product	Command	Brief Description
Natural Security	MAIL	Invokes a mailbox to modify its contents and/or expiration date. A mailbox is used as a notice board to broadcast messages to Natural users.
	PROFILE	Displays the security profile currently in effect. This profile informs you of the conditions of use in effect for you in your current Natural environment.
Predict	LIST XREF	Displays all active cross-reference data for the current library.
	XREF	Controls the use of the Predict function "active cross-references". This function automatically creates documentation in Predict about the objects which a program/data area references.

5 CATALL

- CATALL in Interactive Mode 20
- CATALL in Batch Mode 21

This command is used to catalog, check, save or stow all objects or selected objects in the current library.

CATALL in Interactive Mode

```
CATALL
```

When you issue this command, the **Catalog Objects in Library** window appears. In this window, you specify which types of objects are to be processed. Objects are processed in the order in which the object types are listed in the window (see also the information for TYPES in the section ***CATALL in Batch Mode***). Additionally, you can choose which action is to be performed and which objects are to be processed.

You can make the following specifications in the window:

Starting from	Enter an asterisk (*) if you want to process all objects of the selected types in the current library. If you want to restrict the number of objects to a certain range, you can use asterisk notation for the name.
Apply action only to existing modules	If you mark this option, only those objects for which cataloged objects exist in the current library will be cataloged again; source objects for which no cataloged objects exist will not be processed.
Apply action to all sources	If you mark this option, *all* selected objects will be processed.
Action	After pressing PF2, you can select one of the following actions to be applied to the selected objects: ■ **Catalog** ■ **Check** ■ **Save** ■ **Stow** These actions correspond to the system commands of the same names. **Note:** Under Natural Security, certain actions may be disallowed.
Renumber source lines	By default, the source-code lines of sources that were saved or stowed are not renumbered. If you wish automatic renumbering of lines, write an X in this field.
Object types	By default, CATALL applies to objects of all types in the current library (all object types are marked with an X). If you wish objects of a certain type *not* to be affected by CATALL, overwrite the corresponding X with a blank.

Generate new map source	Maps created with previous Natural versions are not necessarily compatible with Natural Version 3.1 and above. Mark this option to ensure that maps are converted during the catalog operation. This option converts and stores maps as source objects *and* as cataloged objects.

During CATALL processing, a statistics window appears and the objects being cataloged are listed.

Caution: If you press any key while CATALL processing is in progress, CATALL will be stopped.

Upon successful completion of processing, an information message is displayed.

If an object was not cataloged successfully, a window showing the object name, error number and error line is displayed.

If the CATALL command is called from within a Natural program with stacked parameters, Natural assumes batch mode and does not display the error dialog in case of compilation errors. Instead of the dialog a file will be created in Natural's temporary directory (TMP_PATH) with name *library.CTL*. This file is empty if no errors are found, otherwise it contains the failed objects with line number and error code.

CATALL in Batch Mode

```
CATALL object-name [ RECAT ] [TYPES types] [ CATALOG ] [options ...]
                   [ ALL   ]               [ CHECK   ]
                                           [ SAVE    ]
                                           [ STOW    ]
```

For the various specifications you can make in the **Catalog Objects in Library** window, there are also corresponding options which you can specify directly with the system command CATALL:

object-name	The name of the object to be cataloged. Enter an asterisk (*) if you want to catalog all objects of the specified types in the current library. If you want to restrict the number of objects to a certain range, you can use asterisk notation for the name.
RECAT / ALL	Corresponds to the options **Apply action only to existing modules**, or **Apply action to all sources** of the **Catalog Objects in Library** screen. RECAT is the default.
TYPES *types*	Corresponds to the object types marked in the **Catalog Objects in Library** screen. Possible *types* are (objects are processed in the order in which the object types are listed below): D DDMs

	G Global data areas L Local data areas A Parameter data areas C Copycodes T Texts 7 Functions N Subprograms S Subroutines H Helproutines M Maps 8 Adapters P Programs 4 Classes 3 Dialogs * All types (this is the default) The *types* have to be specified as *one* character string, for example, GLA for global, local and parameter data areas. By default, CATALL applies to objects of all types in the current library.	
CATALOG / CHECK / SAVE / STOW	Corresponds to the actions of the same names on the **Catalog Objects in Library** screen. CATALOG is the default.	
options	NOREN	No automatic renumbering of source-code lines of source objects.

Note: The individual command components must be separated from one another either by a blank or by the input delimiter character (as defined with the session parameter ID).

6 CATALOG

```
CATALOG [object-name [library-id]]
```

Related commands: SAVE | STOW.

This command is used to catalog (compile) the source code currently in the work area of a Natural editor and (if the syntax has been found to be correct) store the resulting cataloged object in the current Natural system file.

Important: The CATALOG command cannot be used if the profile parameter RECAT has been set to ON; in this case, use the STOW command to compile and store the object.

CATALOG	If you do not specify an *object-name*, the object is cataloged in the current library under the name of the object last read into the source work area (for example, with EDIT or READ). An existing object code will be replaced.
CATALOG *object-name*	A new object is created. As *object-name*, you specify the name under which the new object is to be cataloged. It is stored in the current library. If the object exists, the command is rejected.
CATALOG *object-name library-id*	If you want the new object to be cataloged into another library, you must specify the *library-id* of that library. If the object exists, the command is rejected.

Note: If an FDIC system file is specified in the parameter file which is not valid, Natural will display an appropriate error message when the CATALOG command is issued.

7 CHECK

CHECK

This command is used to check if the syntax of the source code currently in the editor work area contains any errors.

If a syntax error is detected, syntax checking is suspended and the line containing the error is displayed. You can then either correct the line (whereupon verification continues) or press ENTER without modifying the line displayed. This stops the verification procedure and opens the editor.

Note: Syntax checking is also performed as part of the RUN, STOW, CATALOG and CATALL commands.

8 CLEAR

CLEAR

This command is used to clear the source work area of the editor. It can be used if a new program is to be created and there is another object in the source work area.

9 COMPOPT

- COMPOPT Syntax Explanation 30
- Compiler Options 30
- Specifying Compiler Parameters 30

```
COMPOPT [option=value ...]
```

This system command is used to set various compilation options. The options are evaluated when a Natural object is compiled.

If you enter the COMPOPT command without any options, a screen is displayed where you can enable or disable the options described below.

The default settings of the individual options are set with the corresponding profile parameters in the Natural parameter file.

COMPOPT Syntax Explanation

COMPOPT	If you issue the COMPOPT system command without options, the **Compilations Options** screen appears. The keywords available there are described below.
COMPOPT *option=value*	Instead of changing an option on the screen, you can also specify it directly with the COMPOPT command. Example: `COMPOPT DBSHORT=ON`

Compiler Options

The following compiler options are available. For details on the purpose of these options and the possible settings, see the description of the corresponding Natural profile parameter:

DBSHORT | ECHECK | GFID | KCHECK | MASKCME | MAXPREC | PCHECK | PSIGNF | THSEP | TQMARK

Specifying Compiler Parameters

You can specify compiler parameters on different levels:

1. **As Default Settings**

   The default settings of the individual compiler parameters are specified using the **Compiler Options** category of the Configuration Utility and are stored in the Natural parameter file NATPARM.

2. **At Session Start**

   At session start, you can override the compiler option settings by specifying the corresponding profile parameters.

3. **During an Active Natural Session**

   During an active Natural session, there are two ways to change the compiler parameter values with the `COMPOPT` system command: either directly using command assignment (`COMPOPT` *`option=value`*) or by issuing the `COMPOPT` command without options which displays the **Compilation Options** screen. The settings assigned to a compiler option are in effect until you issue the next `LOGON` command to another library. At `LOGON` to a different libary, the default settings (see item 1 above) will be resumed. Example:

```
OPTIONS KCHECK=ON
DEFINE DATA LOCAL
1 #A (A25) INIT <'Hello World'>
END-DEFINE
WRITE #A
END
```

4. **In a Natural Object**

   In a Natural object (for example: program, subprogram), you can set compiler parameters with the `OPTIONS` statement. Example:

```
OPTIONS KCHECK=ON
WRITE 'Hello World'
END
```

   The compiler options defined in an `OPTIONS` statement will only affect the compilation of this object, but do not update settings set with the command `COMPOPT`.

10 DEBUG

```
DEBUG object-name
```

This command is used to invoke the Natural debugger. With the command, you specify the name of the object to be debugged.

You can use the debugger only in a remote development environment (SPoD), using Natural Studio.

Note: This command is not executable in batch mode.

11 EDIT

- EDIT Syntax 1 36
- EDIT Syntax 2 38
- EDIT Syntax 3 38

This command is used to invoke a Natural editor for the purpose of creating and editing a Natural source.

Note: This command is not executable in batch mode. This command is not available if the Natural program, data area, and map editor are disabled in your environment. Use the source editor of NaturalONE to edit a source object.

Three different forms of command syntax exist. These are documented in the following sections.

Related command: READ.

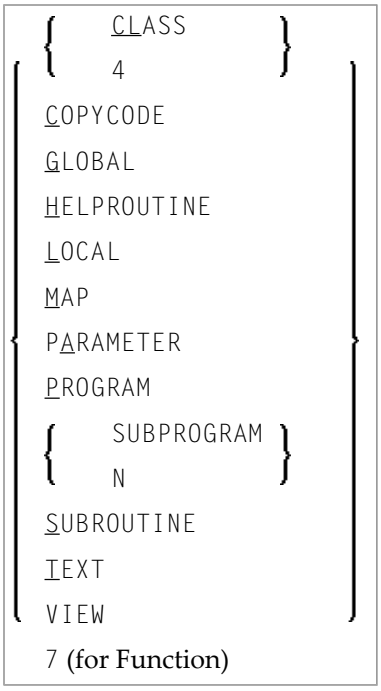
See also *Object Naming Conventions* in the *Using Natural* documentation.

EDIT Syntax 1

```
EDIT [object-type] [object-name [library-id]]
```

object-type

The following object types can be edited:



Which editor is invoked depends on the type of object to be edited:

- Local data areas, global data areas or parameter data areas are edited with the data area editor.
- Maps are edited with the map editor.

- Classes are edited with the program editor.
- `EDIT VIEW` only works in the current library and when an *object-name* is specified. If the object to be viewed is a DDM, the DDM Services are invoked.
- All other types of objects - program, subprogram, subroutine, `7` (for function), helproutine, copycode, text, description - are edited with the program editor.

The object types are described in *Objects for Natural Application Management* in the *Programming Guide*. The editors are described in the *Editors* documentation.

Note: If you are in the program editor and enter `EDIT` *object-name* (whereas the Natural object is of type copycode, class, 7 (for function), helproutine, program, subprogram,subroutine, text), a parallel edit session will be opened. See also the description of the program editor command `NEXT`.

If you specify the name of the object you wish to edit, you need not specify its object type.

object-name

With the `EDIT` command, you specify the name of the object you wish to edit. The maximum length of the object name is 8 characters.

Note: For DDMs, the maximum length is 32 characters.

Natural will then load the object into the edit work area of the appropriate editor and set the object name for a subsequent `SAVE`, `CATALOG`, `STOW` command.

If you do not specify an *object-name* and there is no object in the source work area, the empty program editor screen will be invoked where you can create a program. If the source work area is not empty, the object will be loaded in the appropriate editor.

library-id

If the object you wish to edit is not contained in the library you are currently logged on to, you must specify the *library-id* of the library in which the object to be edited is contained.

If Natural Security is active, a *library-id* must not be specified, which means that you can only edit objects which are in your current library.

EDIT Syntax 2

```
EDIT [ *           ] { *           }
     [ object-type ] { object-name }
```

If you do not remember the name of the object you wish to edit, you can use this form of the `EDIT` command to display a list of objects, and then select from the list the desired object.

`EDIT *`	Displays a list of all objects in your current library.
`EDIT object-type *`	Displays a list of all objects of that type in your current library.

To select an object from a certain range of objects, you can use asterisk notation and wildcard notation for the *object-name* in the same manner as described for the system command .

EDIT Syntax 3

```
EDIT FUNCTION subroutine-name
```

The `EDIT FUNCTION` command may be used to edit a subroutine using the subroutine name (not the object name) with maximally 32 characters.

Note: Please note that the keyword `FUNCTION` used in this syntax is not identical with the Natural **object type** 7 (for function) listed above. See the description of object type Function in the *Programming Guide*.

Example:

```
DEFINE SUBROUTINE CHECK-PARAMETERS
  ...
END-SUBROUTINE
END
```

Assuming that the above subroutine has been saved under the object name `CHCKSUB`, you may edit subroutine `CHECK-PARAMETERS` either by issuing the command:

```
EDIT S CHKSUB
```

or by

```
EDIT F CHECK-PARAMETERS
```

12 EXECUTE

- EXECUTE Syntax Explanation 40
- Examples of EXECUTE Command 41

```
{ EXECUTE  [REPEAT]     program-name      [library-id] }
{ program-name [parameter ...]                         }
```

This command is used to execute a Natural object module of type program. The object module must have been cataloged (that is, stored in object form) in the Natural system file or linked to the Natural nucleus. The execution of an object module does not affect the source code currently in the editor work area.

EXECUTE Syntax Explanation

EXECUTE	The keyword EXECUTE is optional; it is sufficient to specify *program-name*, i.e. the name of the program to be executed.
REPEAT	If the program being executed produces multiple screen output and you wish the screens to be output one after another without intervening prompts, you specify the keyword REPEAT together with the keyword EXECUTE.
program-name	The name of the program to be executed. If you do not specify a *library-id*, Natural can only execute the specified program if it is stored either in your current library or in the current steplib library (the default steplib is SYSTEM).
library-id	If the program is stored in another library, specify the *library-id* of that library. In this case, the program can only be executed if it is actually stored in the specified library. A *library-id* that begins with SYS must not be specified (except SYSTEM).
parameter	When you execute a program by specifying the program name without the keyword EXECUTE, you may pass parameters to the program. These parameters will be read by the first INPUT statement in the executed program. You can specify the parameters as positional parameters or as keyword parameters, with the individual specifications separated from one another by blanks or the input delimiter character (as specified with the session parameter ID). **Note:** If one of the parameters passed contains blanks or is a string which contains blanks, the transfer will only be executed if directly after the program name an input delimiter is set.

Examples of EXECUTE Command

```
EXECUTE PROG1
```

```
EXECUTE PROG1 ULIB1
```

```
PROG1
```

```
PROG1 VALUE1 VALUE2 VALUE3
```

```
PROG1 VALUE1, VALUE2, VALUE3
```

```
PROG1 PARM1=VALUE1, PARM2=VALUE2, PARM3=VALUE3
```

```
PROG1 PARM3=VALUE3 PARM1=VALUE1 VALUE2
```

```
PROG1,ab cd ef,gh,de fg,ab
```

13 FIN

FIN

This command is used to terminate a Natural session. It applies to online sessions as well as batch mode sessions.

A batch mode session is also terminated when an end-of-file condition is detected in the command input data set.

14 GLOBALS

- GLOBALS Syntax Explanation 46
- GLOBALS Parameters 46
- Interaction with SET GLOBALS and Other Statements 47

```
GLOBALS [parameter=value ...]
```

This command is used to set Natural session parameters.

Note: In batch mode, this command is only executable, if parameters are specified. For example, `GLOBALS SM=ON` can be executed in batch mode.

GLOBALS Syntax Explanation

`GLOBALS`	If the `GLOBALS` command is entered without parameters, a screen appears where you can modify the parameter settings.
parameter	Parameter settings can be supplied in any order and must be separated by a blank. If more parameters are specified than will fit on one command line, several `GLOBALS` commands must be issued. Example: `GLOBALS DC=, ID=.` **Note:** Certain session parameters can be modified only using the above mentioned screen, but not via the command line.

GLOBALS Parameters

The following table contains a list of session parameters that can be specified with the system command `GLOBALS`.

Parameters	Function
`CC`	Error Processing in Batch Mode
`CF`	Character for Terminal Commands
`CO`	Compiler Output
`CPCVERR`	Code Page Conversion Error
`DBSHORT`	Interpretation of Database Short Names
`DC`	Character for Decimal Point Notation
`DO`	Display Order of Output Data
`DU`	Dump Generation
`EJ`	Page Eject
`ENDIAN`	Endian Mode for Compiled Objects

Parameters	Function
FS	Default Format/Length Setting for User-Defined Variables
GFID	Global Format IDs
IA	Input Assign Character
ID	Input Delimiter Character
IM	Input Mode
LE	Reaction when Limit for Processing Loop Exceeded
LS	Line Size
LT	Limit for Processing Loops
ML	Position of Message Line
PD	Size of Page Data Set
PM	Print Mode
PS	Page Size for Natural Reports
RQTOUT	REQUEST DOCUMENT Timeout
SA	Sound Terminal Alarm
SF	Spacing Factor
SM	Programming in Structured Mode
THSEPCH	Thousands Separator Character
ZD	Zero-Division Check
ZP	Zero Printing

Interaction with SET GLOBALS and Other Statements

Statement SET GLOBALS

The system command GLOBALS and the statement SET GLOBALS can both be used in the same Natural session. Parameter values specified with a GLOBALS command remain in effect until they are overridden by a new GLOBALS command or SET GLOBALS statement, the session is terminated, or you log on to another library.

Other Statements Influencing the Session Parameter Settings

Some parameter values may be overridden during program execution using the LIMIT, EJECT, and FORMAT statements and by format entries specified in INPUT, DISPLAY, PRINT, and WRITE statements.

15 HELP

```
{ HELP } [ [NAT]nnnn ]
{ ?    } [ USER[nnnn] ]
```

This command is used to invoke online help for error messages.

Note: This command is not executable in batch mode.

`HELP`	Displays the help menu.
`HELP [NAT]`*nnnn*	Entering `HELP` and a number (up to 4 digits, optionally prefixed by "NAT") displays the detailed message text for the Natural error condition associated with that number, that is, the long text of the Natural system error message NAT*nnnn*.
`HELP USER`*nnnn*	Displays the long text of the library-specific error message number *nnnn* in the current library.

16 INPL

`INPL [R]`

This command is used to invoke the Natural `INPL` utility. This utility is *only* used for the loading of installation data sets into the system files as described in the `INPL` online help and in the platform-specific installation documentation.

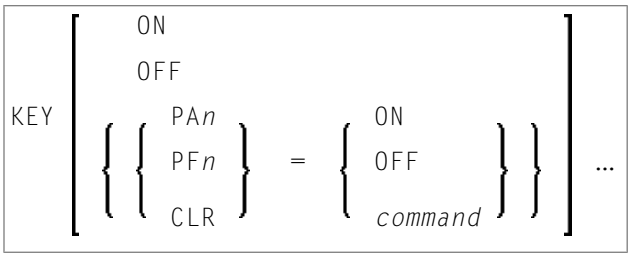
Apart from that, you use the Object Handler to load objects into the system files.

`INPL`	If you enter the `INPL` command without any parameters, the `INPL` utility will be invoked.
`INPL R`	Invokes the `INPL` utility function Natural Security Recover which is only available if Natural Security is installed. It can be used to reset the access to the Natural Security library `SYSSEC`: the user DBA, the library `SYSSEC`, and the link between the two will be redefined as after the initial installation, while all other links to `SYSSEC` will be cancelled. See also *Inaccessible Security Profiles* in the section *Countersignatures* of the *Natural Security* documentation.

For further information, see *INPL Utility* in the *Tools and Utilities* documentation.

17 KEY

- Assigning Commands 54
- Activating/Deactivating All Keys - KEY ON/OFF 55
- Activating/Deactivating Individual Keys - KEY key=ON/OFF 55



This command is used to assign functions to keys on the keyboard of video terminals. Moreover, you can change, activate and deactivate the assigned functions.

This is possible for the following keys:

- PA1 to PA3,
- PF1 to PF24
- CLEAR

To each of these keys, you can assign one of the following functions:

- a Natural system command,
- a Natural terminal command,
- a user-defined command.

Natural will execute the assigned command whenever you press the corresponding key in command mode (**Direct Command** window).

Notes:

1. Assignments made with the system command KEY are totally independent of assignments made with a SET KEY statement in a program.
2. Function-key assignments can also be made by the Natural administrator via the profile parameter KEY.
3. This command is not executable in batch mode.

Assigning Commands

If you enter only the command KEY (without parameters), the **Function-Key Assignments** screen will be displayed. On this screen, you can assign commands to the individual keys by entering the command names in the input fields.

To assign a different command to a key, you overwrite the existing entry in the input field.

To delete a command assignment, you delete the entry in the input field or overwrite it with blanks.

You can also assign commands to individual keys by specifying them directly with the KEY command. For example:

```
KEY PF1=CLEAR
```

If the assigned command contains blanks, it has to be enclosed in apostrophes. For example:

```
PF13='UPDATE OFF'
```

Activating/Deactivating All Keys - KEY ON/OFF

With the command KEY OFF/ON, you deactivate/re-activate all function-key assignments:

KEY OFF	If you the press a function key, Natural will return an appropriate message indicating that the key is not active.
KEY ON	Re-activates all function-key assignments that have previously been deactivated with KEY OFF.

You can also activate/deactivate the keys by overwriting the entry ON/OFF in the field **Activate Keys** at the top right-hand corner of the **Function-Key Assignments** screen.

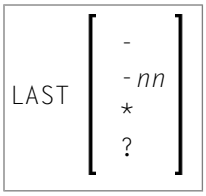
Activating/Deactivating Individual Keys - KEY key=ON/OFF

With the command KEY *key*=OFF/ON, you deactivate/re-activate the command assigned to a specific *key*.

KEY *key*=OFF	Deactivates the command assigned to a specific *key*. For example: `KEY PF24=OFF`
KEY *key*=ON	Re-activates a previously deactivated command assignment. For example: `KEY PF24=ON`

Note: When you deactivated an individual key (for example, PF24=OFF), then deactived all keys (KEY=OFF) and, after that, activate all keys again (KEY=ON), the individually deactivated key is activated, too.

18 LAST



This command is used to display the system command(s) that was/were last executed. Moreover, you can have the displayed command(s) executed again. You can also overwrite them before they are executed.

Only system commands that you actually entered can be displayed via the LAST command; commands issued internally by Natural as a result of a command you entered are not available via LAST.

LAST	The command that was issued last is placed in the **Direct Command** window and can be executed.
LAST -	The command that was issued last is placed in the **Direct Command** window and can be executed. If you enter LAST - again, the last but one command is placed in the **Direct Command** window. By repeatedly entering LAST -, you can thus "page" backwards command by command. **Note:** Instead of repeatedly entering it by hand, you can assign LAST - to a PF key via the system command KEY.
LAST -*nn*	Natural "remembers" up to the last 20 commands that were issued; *nn* must therefore not be greater than 20. The last command but *nn* is placed in the **Direct Command** window and can be executed.
LAST *	A window is displayed showing the last 20 commands that were issued. Use PF8 and PF7 to scroll forward and backward if more than 10 commands are displayed:

	■ To execute a *single* command again, either mark the command with the cursor and press F5, or mark the command with any character and press ENTER. ■ To execute *several* commands again, mark them with numbers in the order in which you wish them to be executed and press ENTER, the commands will then be executed in ascending order of numbers.
`LAST ?`	The Help function for the `LAST` command is called.

19 LASTMSG

LASTMSG

This command is used to display additional information about the error situation which has occurred last.

When Natural displays an error message, it may in some cases be that this error is not the actual error, but an error caused by another error (which in turn may have been caused by yet another error, etc.). In such cases, the LASTMSG command allows you to trace the issued error back to the error which has originally caused the error situation.

When you enter the command LASTMSG, you will get - for the error situation that has occurred last - the error message that has been displayed, as well as all preceding (not displayed) error messages that have led to this error, and the date and time behind the error message itself.

To display information on the corresponding error

- Mark one of these messages with the cursor and press ENTER or PF5.

  The following is displayed:

  - error number;
  - number of the line in which the error occurred;
  - name, type and level of the object that caused the error;
  - name, database ID and file number of the library containing the object;
  - error class (system = error issued by Natural; user = error issued by user application);
  - error type (runtime, syntax, command execution, session termination, program termination, remote procedure call);
  - date and time of the error.

Function Keys

The output screen of the LASTMSG command provides the following function keys:

PF Key	Name	Function
PF1	Help	Invoke help. Provides information about the use of PF2 and PF5.
PF2	Clear	Clears all the error messages from the error message collector.
PF5	Info	Provides detailed information about an error, when the cursor is placed on the error message and you press PF5. The same thing happens when the cursor is placed on an error message and you press ENTER.

Note: The library SYSEXT contains the application programming interface USR2006 which enables you to display in your Natural application the error information supplied by LASTMSG.

Natural RPC (Remote Procedure Call):

If an error occurs on the server, the following error information is not displayed: database ID, file number, date and time.

20 LIST

- LIST Syntax Overview 62
- Displaying an Individual Source 63
- Displaying a List of Objects 64
- Displaying Numbers and Sizes of Objects 65
- Displaying Directory Information 66
- Displaying Views 66
- Displaying File Information of Resource Objects 67
- Displaying File Information of Error Message Containers 67

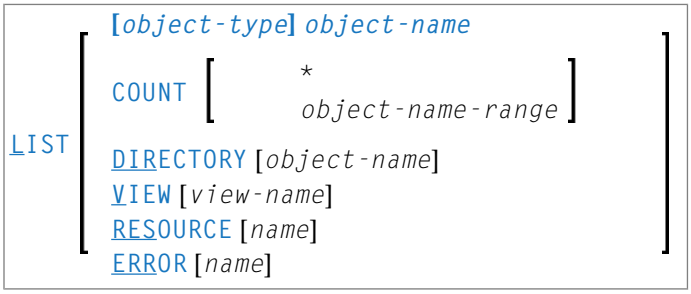
This system command is used to display the source code of a single object or to list one or more objects which are contained in the current library.

Note: This command is not executable in batch mode.

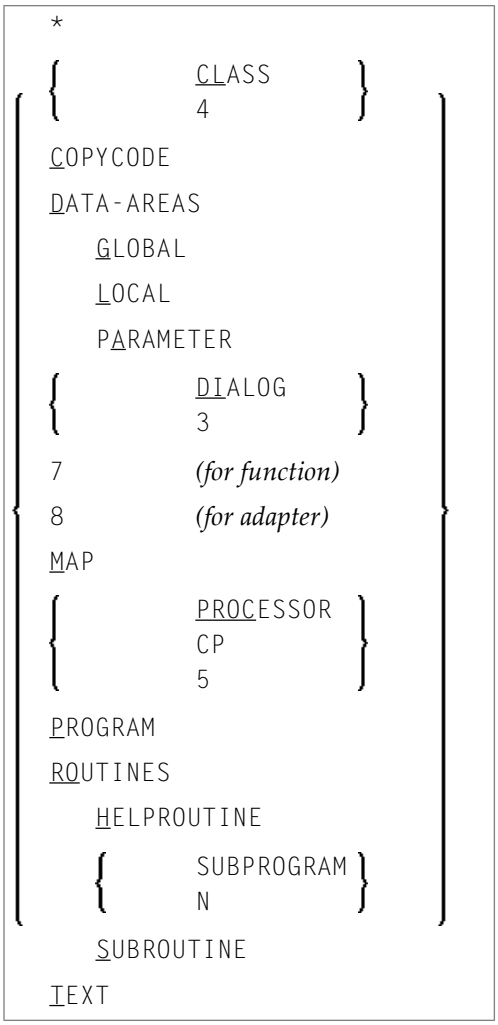
This chapter covers the following topics:

See also the description of the command `LIST XREF`.

LIST Syntax Overview



object-type



object-name

In place of *object-name*, you may specify the name of an object (8 characters long at maximum). You may also specify asterisk notation (*), see the **examples** below.

Displaying an Individual Source

LIST	If you enter only the LIST command itself, without any parameters, the contents of the source of the object currently selected will be listed.
LIST *object-name*	If you enter a single object name with the LIST command, you need not specify the *object-type*. If you specify an *object-type*, you must also specify an *object-name*. In both cases, the object's source code will be listed.
LIST *object-type object-name*	

Displaying a List of Objects

`LIST object-name` `LIST object-type object-name`	To have all objects in the current library listed, except DDMs, you specify an asterisk (*) for the *object-name*, but no *object-type*. To have all objects of a certain type listed, you specify a certain *object-type* and an asterisk (*) for the *object-name*. If you wish a certain range of objects to be listed, you can use asterisk notation (*) for the *object-name* and/or wildcard notation (?).

Examples

- List all objects in the current library, except views, resources, errors - regardless of their types:

```
LIST *
```

- List all subroutines in the current library:

```
LIST S *
```

- List all objects (of any type) whose names begin with `SYS`:

```
LIST SYS*
```

- List all maps whose names begin with `SYS`:

```
LIST M SYS*
```

- List directory information of object `PRG01` in current library:

```
LIST DIR PRG01
```

- List all objects such as `NATAL`, `NATURAL`, `NAT`*vr*`AL` (where *vr* represents the relevant product version):

```
LIST N?T*AL
```

Performing a Function on an Object

To perform a function on an object from the selection list, you simply mark the object with the appropriate function code in the left-hand column.

The function codes are:

Code	Function
C	Check the object's source code.
D	Read the object's source code.
E	Edit the object's source (equivalent to the system command EDIT).
L	List the object's source code.
I	List directory information of the object's source code (equivalent to LIST DIRECTORY *object-name*).
H	Print hardcopy of the object's source.
R	Run (that is, compile and execute) the object's source (equivalent to the system command RUN).
X	Execute the object (equivalent to the system command EXECUTE).
S	Stow the object in source and object form (equivalent to the system command STOW).
U	Delete the object's source and object form.
.	End.

Enter a question mark (?) or use F2 to display the list of the available function codes for the selected object.

Displaying Numbers and Sizes of Objects

LIST COUNT	Displays a summary report that contains the numbers and sizes (in bytes or KB if greater than 1 MB) of all objects stored in the current library. The numbers and sizes listed refer to all objects that have been saved as source (**Saved**) objects only or as cataloged (**Cataloged**) objects only, and all objects for which both saved and cataloged objects (**Stowed**) exist.	
LIST COUNT *	Displays a summary report of all objects where the numbers and sizes of saved/cataloged objects are listed per object type(s) found.	
LIST COUNT *object-name-range*	Displays a summary report (same as above) for the specified range of objects. Valid range notations are:	
	value>	All objects with names greater than or equal to *value*
	value<	All objects with names less than or equal to *value*
	*value**	All objects with names that start with *value*

Displaying Directory Information

`LIST DIRECTORY`	Displays the directory information about the last active object currently in the source work area: ■ **Source code:** "Saved-on" date and time, library name, user ID, programming mode (reporting or structured), Natural version, code page information (if available), operating system, size, encoding. ■ **Object code:** "Cataloged-on" date and time, library name, user ID, programming mode, Natural version, code page information (if available), operating system/version, size, Endian mode. Directory information on the saved source code cannot be always exact, because the source code can be edited with non-Natural editors which are not under the control of Natural.
`LIST DIRECTORY` *`object-name`*	Displays the directory information about the specified object. If you use asterisk notation (*) for *`object-name`*, the directory information of the existing objects is displayed sequentially.

Note: The code page information displayed shows the first 32 characters of the code page only.

Displaying Views

`LIST VIEW`	Displays a list of all views (DDMs).
`LIST VIEW` *`view-name`*	If you specify a single view name, the specified view will be displayed. For the *`view-name`*, you can use asterisk notation to display a list of all views (*) or a certain range of views (for example: A*).

Displaying File Information of Resource Objects

`LIST RESOURCE name`	Displays the file information about the specified resource object. For *name*, you may only use asterisk notation (*).

Example - Display the file information of all resource objects whose name starts with a W:

```
LIST RESOURCE W*
```

Displaying File Information of Error Message Containers

`LIST ERROR name`	Displays the file information about the specified error message container `NnnAPMSL.MSG`, where *nn* is the language code. For *name*, you may only use asterisk notation (*).

21 LIST XREF

```
LIST XREF
```

This command is only available if Predict has been installed.

It is used to display all active cross-reference data for the current library.

For further information, see *List XREF For Natural* in the Predict documentation.

22 LOGOFF

LOGOFF

Related command: LOGON.

This command is used to cause the library ID to be set to SYSTEM and the Adabas password to be set to blanks. The contents of the source program work area are not affected by this command.

LOGOFF has no effect on Natural global parameter settings.

For information on LOGOFF processing under Natural Security, see *How to End a Natural Session* in section *Logging On* of the *Natural Security* documentation.

Note: LOGOFF does *not* cause the Natural session to be terminated.

To terminate the session

- Use the system command FIN, or execute a program that contains a TERMINATE statement.

23 LOGON

```
LOGON library-id [password]
```

Related command: LOGOFF.

This command is used to log on to a library in your environment or create a new library. In the specified library, all newly created source or object programs saved during the session will be stored (unless you explicitly specify another library ID in a SAVE, CATALOG or STOW command).

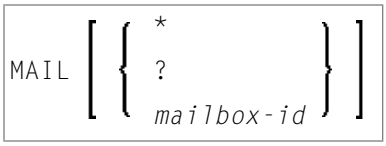
The LOGON command has no direct effect on the source program in the currently active window.

LOGON causes all Natural global data areas and application independent variables (AIVs), all assignments made using the SET KEY statement and retained ISN lists to be released. Data definition modules (DDMs) contained in the DDM buffer area are also released.

LOGON *library-id*	The library ID can be 1 to 8 characters long and must not contain blanks. It can consist of the following characters:	
	A - Z	upper-case alphabetical characters
	0 - 9	numeric characters
	-	hyphen
	_	underscore
	The first character of a library ID must be an upper-case alphabetical character.	
LOGON *library-id password*	The Adabas password; see *Session Parameters* in section *Library Maintenance* of the *Natural Security* documentation.	

For information on LOGON processing under Natural Security, see *Logging On* in the *Natural Security* documentation.

24 MAIL



This command is used to invoke a mailbox which is a kind of “notice board” used to broadcast messages under Natural Security. The contents and/or expiration date of the mailbox can be modified.

`MAIL`	If you enter the `MAIL` command without any parameters, a window is displayed prompting you to enter a mailbox ID.
`MAIL *`	A list of all mailboxes you may use is displayed, and you may then select a mailbox from the list.
`MAIL ?`	
`MAIL mailbox-id`	If you specify a *`mailbox-id`* (maximum 8 characters), the corresponding mailbox is invoked directly. The *`mailbox-id`* must have been defined in Natural Security.

For further information, see *Mailboxes* in the *Natural Security* documentation.

25 PROFILE

This command is available only if Natural Security is installed.

```
PROFILE
```

This command is used to display the security profile currently in effect. This profile informs you of the conditions of use in effect for you in your current Natural environment.

For further information, see *PROFILE Command* in the *Natural Security* documentation.

26 PROFILER

PROFILER

This command is used to invoke the Profiler utility. The Profiler utility monitors the internal process flow of a Natural application and analyzes the performance of the application.

The Profiler utility can be executed in batch mode only.

For further information, see *Profiler Utility* in the *Tools and Utilities* documentation.

27 PURGE

```
PURGE [object-name ...]
```

This command is used to delete one or more source objects.

Note: If the Natural profile parameter RECAT is set to ON, the PURGE command will be rejected for a source for which a corresponding cataloged object exists.

If DDMs reside in a library of the FUSER system file, they can be deleted with the PURGE command.

PURGE	If you enter the PURGE command without an *object-name*, a list of all objects in the current library will be displayed; on the list, you can then mark the objects to be deleted.
PURGE *object-name*	As *object-name*, you specify the name(s) of the object(s) to be deleted. You can only delete objects that are stored in the library to which you are currently logged on. If you wish to delete all objects whose names begin with a specific string of characters, use asterisk notation (*) for the *object-name*.

28 READ

```
READ object-name [library-id]
```

Related command: EDIT.

This command is used to transfer an object that is stored in source form into the source work area. Any object currently in the source work area will be overwritten by the object read.

See also *Object Naming Conventions* in the *Using Natural* documentation.

READ *object-name*	The name of the object to be read. If *object-name* is specified without a library ID, the object will be read only if it is stored in the library to which you are currently logged on.
READ *object-name library-id*	The library in which the object to be read is contained. If both *object-name* and *library-id* are specified, Natural will only read the object if it is stored under the specified library ID.

29 RENAME

```
RENAME [old-name [new-name [new-type]]]
```

This command is used to give a Natural object another name. In addition, you can change the object type.

You can only rename one object at a time. The object to be renamed must be stored in the library to which you are currently logged on. To ensure consistency, Natural will rename source code or object module or both.

See also *Object Naming Conventions* in the *Using Natural* documentation.

RENAME	If you issue the command without parameters, a **Rename Object** window appears where you can specify the same parameters as in the command line.	
old-name	As *old-name* you specify the existing name of the object to be renamed.	
new-name	As *new-name* you specify the name under which the object is to be stored from now on.	
new-type	When you rename an object in source form, you can also change its object type by specifying the corresponding character for *new-type*. The possible values you can specify for *new-type* are:	
	3	Dialog
	4	Class
	5	Processor
	7	Function
	8	Adapter
	9	Resource
	A	Parameter data area
	C	Copycode
	G	Global data area

	H	Helproutine
	L	Local data area
	M	Map
	N	Subprogram
	O	Macro
	P	Program
	S	Subroutine
	T	Text
	Y	Rule
	Z	Recording

30 RENUMBER

```
RENUMBER [(n)]
```

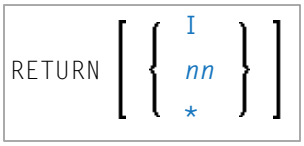
This command is used to renumber the lines in the source code currently in the work area of a Natural editor.

Note: If you want to renumber alphanumeric or Unicode constants, make sure that the `RNCONST` profile parameter is set to `ON`.

`RENUMBER`	If you enter the command without parameter, depending on the total number of source lines in the work area, the following default increment values are used for renumbering:	
	0001 to 0999 lines:	increments of 10
	1000 to 1999 lines:	increments of 5
	2000 to 4999 lines:	increments of 2
	5000 lines and more:	increments of 1
	Starting with line 9999, the line number assigned is always `9999`. Be aware that you cannot compile sources that contain more than 9999 lines.	
`RENUMBER (n)`	*n* can be used to specify a value between 1 and 9999 as the increment for renumbering. If the given increment value would cause the renumbering to exceed the 9999 line number limit, the default increment value is used instead.	

See also *Renumbering of Source-Code Line Number References* in the *Programming Guide*.

31 RETURN



This command is used to return to a previous (or initial) Natural application.

RETURN	If RETURN is specified without any parameters, control will be returned to the previous application (as defined with the system command SETUP). All information about this previous application will be deleted. If no previous application exists, control is returned to the initial application. If RETURN is issued and no return point is set, the RETURN command will be ignored. **Under Natural Security:** A LOGOFF command will be executed if RETURN is issued and no return point has been set.
RETURN I	This command causes control to be returned directly to the initial application. This option also causes Natural to delete all definitions of previous applications (except that of the initial application).
RETURN *nn*	This command causes control to be returned to the *nn*th previous application. When this option is used, all information for applications subsequent to the *nn*th application is deleted.
RETURN *	This command will display a list of all return points which are currently set up. On the list you may then select the return point to which you wish to return.

See the SETUP command for further information and examples.

32 RPCERR

RPCERR

This command is used to display the last Natural error number and message if it was RPC related, and it also displays the last EntireX Broker reason code and associated message. Additionally, the node and server name from the last EntireX Broker call can be retrieved.

For further information, see *Monitoring the Status of an RPC Session* in the *Operating a Natural RPC Environment* section of the *Natural RPC (Remote Procedure Call)* documentation.

33 RUN

RUN [REPEAT] [*program-name* [*library-id*]]

This command is used to compile and execute a source program. The program may be in the source work area or in the Natural system file.

RUN	If *program name* is not specified, Natural will compile and execute the program currently residing in the work area.
REPEAT	REPEAT defines that if the program being executed produces multiple screen output, the screens are to be output one after another without intervening prompting messages. When the program terminates, Natural will enter command mode.
program-name	The name of the program to be run. If *program-name* is specified without a library ID, Natural will read the source program into the source work area, compile, and execute the specified program only if it is stored under the current library ID. If it is not stored under the current library ID, an error message will be issued.
library-id	The library in which the program to be run is contained. If both *program-name* and *library-id* are specified, Natural will retrieve, compile, and execute the specified program only if it is stored under the library ID specified. If it is not stored under the current library ID, an error message will be issued. The setting for *library-id* must not begin with SYS (except SYSTEM).

34 SAVE

```
SAVE [object-name [library-id]]
```

Related commands: STOW | CATALOG.

This command is used to save the source code currently contained in the work area of a Natural editor and store it as a source object in the current Natural system file.

Caution: The SAVE command cannot be used if the profile parameter RECAT has been set to ON; in this case, use the STOW command to compile and store the object.

SAVE	If you use the command without *object-name*, the current source object in the source work area will be saved in the current library. An existing source code will be replaced.
SAVE *object-name*	A new source object is created. As *object-name*, you specify the name under which the source object is to be saved. The new source object is stored in the current library. If the source object exists, the command is rejected.
SAVE *object-name library-id*	When you save a source object under a different name or save a newly created object, the source object will, by default, be stored in the current library. If you wish to store it in another library, you have to specify the desired *library-id* after the *object-name*. A new source object is created, if the source object exists, the command is rejected.

35 SCAN

- SCAN Menu Options 98
- SCAN Subcommands 99
- SCAN under Natural Security 100

SCAN

This command is used to search for a string of characters within an object, with the option to replace the string with another string.

The object may be a single object, all objects beginning with a specified setting, or all objects within a library. The scan may also be restricted to a specific object type.

Important: The source work area is used by the SCAN command. Therefore, a SAVE or STOW command should be issued before using the SCAN command.

Note: This command is not executable in batch mode.

This chapter covers the following topics:

SCAN Menu Options

Any objects except maps, DDMs and data areas can be modified using the scan utility. You can also use the full screen editor to modify other lines than scanned lines. When the object is modified, save it and close the editor. You can then continue scan processing.

When you issue the SCAN command, a window is displayed where you can specify the following:

Field	Explanation	
Scan value	The value to be searched for.	
Replace value	The value which is to replace the scan value.	
Delete value	Will cause the scan value to be deleted (Y/N).	
Object name	The object(s) to be scanned:	
	blank	All objects.
	*	
	*name**	All objects whose names begin with *name*.
Object type	The type(s) of object to be scanned:	
	P	Programs
	C	Copycodes
	N	Subprograms
	S	Subroutines
	H	Helproutines
	M	Maps
	G	Global data areas
	L	Local data areas

Field	Explanation	
	A	Parameter data areas
	T	Texts
	4	Classes
	*	All types
Absolute scan	Y	Will result in an absolute scan.
	N	If absolute scan is set to "N", the scan value must be separated by a blank or specific character.
Library	The ID of the library to be scanned. Default is the current library.	
Case sensitive	N	The search will be for the scan value regardless whether it occurs in upper case, lower case, or a mixture of both.
	Y	The search will be for the scan value exactly as you specify it.

Make the desired specifications and press ENTER. A window appears displaying the lines containing the searched for scan value.

```
+-------------------- Search value MOVE in Program READVAR----------------------+
¦ Commands:                                                                     ¦
¦ 0440     MOVE EMPLOYEES-VIEW.NAME       TO #NAME (J)                        ¦
¦ 0450     MOVE EMPLOYEES-VIEW.FIRST-NAME TO #FIRST (J)                       ¦
¦ 0960            MOVE #NAME(6) TO #START-NAME                                 ¦
¦ 0980            MOVE 'D' TO #DIR           /* new direction DESCENDING       ¦
¦ 1020            MOVE #NAME(1) TO #START-NAME                                 ¦
¦ 1050          MOVE 'A' TO #DIR             /* new direction ASCENDING        ¦
¦ 1090          MOVE #NAME(6) TO #START-NAME                                   ¦
¦ 1180          MOVE #NAME(6) TO #START-NAME                                   ¦
+-------------------------------------------------------------------------------+
```

SCAN Subcommands

Any desired subcommand can be entered in the above window. To enter the command mode, place the cursor in the **Commands** line and press ENTER.

Then you can select one of the following commands:

Command	Function
Edit	Edit object. **Note:** This command is not available if the Natural program, data area and map editor are disabled in your environment. For more information, see *NaturalONE as the Default Development Environment* in the *Editors* documentation.
List	List object as it currently appears in the source work area.
Ignore	Ignore the object currently being scanned, do not save any modifications, and continue with next object.
Quit	Terminate scan processing.

A subcommand can also be invoked by entering its first character.

To deactivate command mode, press ESC.

SCAN Editing Rules

- If the Replace option is used and/or an object is updated in the scan utility, the object will always be saved unless **Ignore** or **Quit** is specified before the next object is scanned.
- Lines containing `PASSWORD=`, `PASSW=`, `CIPHER=`, or `CIPH=` will be ignored by the `SCAN` command.
- The line length of the source object in the scan utility is limited to 72.
- If the replace value causes a line to exceed 80 characters, the line will be split automatically.

SCAN under Natural Security

In order to use `SCAN` in a Natural Security environment, the system commands `LIST`, `EDIT`, and `READ` must be allowed in the current library's security profile. If the Replace option is to be used, the system command `SAVE` must also be allowed.

Under Natural Security, the use of the `SCAN` command may be disallowed in some libraries.

36 SCRATCH

```
SCRATCH [ { *                } ]
          { object-name... }
```

This command is used to delete one or more objects - in both source and object form. The contents of the source work area is not affected.

SCRATCH SCRATCH *	If you enter the SCRATCH command without an *object-name* or without an *object-name* but with an asterisk (*), a list of all objects or all selected objects in the current library will be displayed. On the list you may then mark the objects to be deleted.
SCRATCH *object-name*	As *object-name*, you specify the name(s) of the object(s) to be deleted. You can only delete objects which are stored in your current library. If you wish to delete all objects whose names begin with a specific string of characters, use asterisk notation (*) for the *object-name*.

Note: If an FDIC system file is specified in the parameter file which is not valid, Natural will display an appropriate error message when the SCRATCH command is issued.

37 SETUP

- SETUP Syntax Explanation 104
- SETUP/RETURN Example 105

```
SETUP [application-name] [command-name] [I]
```

This command is used to define applications to which control is to be returned using the RETURN command. This allows you to easily transfer from one application to another during a Natural session.

This chapter covers the following topics:

SETUP Syntax Explanation

The command syntax and the parameters that can be issued with the SETUP system command are explained below. If a parameter is to be omitted, you may use the input delimiter character to mark the beginning of the following parameter(s).

SETUP	If SETUP is issued without parameters, a menu will be displayed for the purpose of entering the command information.
application-name	The name of the application to which control is to be returned. A maximum of 8 characters may be used (A8). If *application-name* is blank, a LOGON command will not be issued. This permits multiple return points within the same application. If *application-name* is "*", the current setting of the system variable *LIBRARY-ID (that is, at the time SETUP is issued) is used to create the LOGON command when RETURN is issued.
command-name	The name of the command which is to be executed when control is returned to the application. A maximum of 60 characters may be used (A60). If *command-name* is blank, no command will be issued after the LOGON. This is useful for applications under Natural Security for which a startup program has already been defined. If *command-name* is "*", the current setting of the system variable *STARTUP (that is, at the time SETUP is issued) is used as the startup command when RETURN is issued.
I	If the I option is specified, all return points defined with previous SETUP commands will be deleted and the application specified with SETUP I will be defined as the new initial application. In a non-Security environment, if you log on from library SYSTEM to another library and no return point has been set, this other library will automatically be set as initial return point.

SETUP/RETURN Example

1. User starts Natural session (default application is `APPL1`).

   Return point `APPL1` is defined on Level 1.
2. User issues command `LOGON APPL2`.
3. User executes a program which stacks two commands (establish return point and go to another application):

   ```
   SETUP *,MENU
   LOGON APPL3
   ```

   Return point `APPL2, STARTUP MENU` is defined on Level 2.
4. User issues command `LOGON APPL4` (user selects another application).
5. User presses a PF key which has the setting `RETURN`. Natural will issue for the user:

   ```
   LOGON APPL2
   MENU
   ```

   Return to `APPL2`, delete Level 2.
6. User executes a program which stacks:

   ```
   SETUP *,MENU
   LOGON APPL5
   ```

   Return point `APPL2, STARTUP MENU` is defined on Level 2.
7. User executes a program which stacks:

   ```
   SETUP *,MENU
   LOGON APPL6
   ```

   Return point `APPL5, STARTUP MENU` is defined on Level 3.
8. User executes a program which stacks:

   ```
   SETUP *,MENU
   LOGON APPL7
   ```

   Return point `APPL6, STARTUP MENU` is defined on Level 4.
9. User executes a program which stacks:

```
SETUP *,MENU
LOGON APPL8
```

Return point `APPL7, STARTUP MENU` is defined on Level 5.

10. User executes a program which stacks:

```
SETUP *,MENU
LOGON APPL9
```

Return point `APPL8, STARTUP MENU` is defined on Level 6.

11. User issues command `RETURN 2` (return two levels back).

Natural will return user to `APPL7`, since that was the second previous session (all information for `APPL8` is now lost). Level 6 (`APPL8`) is deleted, Level 5 (`APPL7`) is activated and level deleted.

12. User issues command `RETURN`.

Level 4 (`APPL6`) is activated, level deleted. Natural will return user to `APPL6`, since that was the session previous to `APPL7`.

13. User issues command `RETURN`.

Level 3 (`APPL5`) is activated, level deleted. Natural will return user to `APPL5`, since that was the session previous to `APPL6`.

14. User issues command `RETURN I`.

Level 2 (`APPL2`) is deleted, Level 1 (`APPL1`) is activated.

38 STOW

```
STOW [object-name [library-id]]
```

Related commands: SAVE | CATALOG.

This command is used to catalog (compile) and store a Natural object (in both source and object form) in the current Natural system file. You can regard this command as a CATALOG followed by a SAVE.

STOW	If you use the command without *object-name*, the source code held in the source area as well as the generated code will be stored under the same name in the current library. Existing source and object code will be replaced.
STOW *object-name*	Use this command syntax to store a new object (source and generated code) named *object-name* in the current library. If the object exists in either source or cataloged form, the command is rejected.
STOW *object-name library-id*	If both *object-name* and *library-id* are specified, a new object will be created and stored under that name in the specified library ID. If the object exists in either source or cataloged form, the command is rejected.

Note: If an FDIC system file is specified in the parameter file which is not valid, Natural will display an appropriate error message when the STOW command is issued.

39 STRUCT

- Indentation of Source Code Lines 110

```
STRUCT [(n)]
```

This command is used to perform structural indentation of the source code of the Natural object currently in the work area of the editor.

`STRUCT`	By default (that is, if `(n)` is not specified), indentation is by 2 positions.
`STRUCT (n)`	The parameter `(n)` may be supplied to specify the number of spaces used for indentation. Possible values: 1 - 9. Example: `STRUCT (5)`

The following types of statements are affected by the `STRUCT` command:

- processing loops (`READ`, `FIND`, `FOR`, etc.),
- conditional statement blocks (`AT BREAK`, `IF`, `DECIDE FOR`, etc.),
- `DO/DOEND` statement blocks,
- `DEFINE DATA` blocks,
- inline subroutines.

This chapter covers the following topics:

Indentation of Source Code Lines

You can have a source program indented so that the indentation of source-code lines reflects the structure of the program.

Note: Indentation is performed differently for a reporting-mode program than for a structured-mode program.

Partial Indentation

You can exclude sections of your program source from structural indentation by using the special statements `/*STRUCT OFF` and `/*STRUCT ON`. These must be entered at the beginning of a source-code line. The source-code lines between these two statements will remain as they are when you issue the `STRUCT` command.

Example of Structural Indentation

Program before being structurally indented:

```
DEFINE DATA LOCAL
1 EMPL VIEW OF EMPLOYEES
2 PERSONNEL-ID
2 FULL-NAME
3 FIRST-NAME
3 NAME
1 VEHI VIEW OF VEHICLES
2 PERSONNEL-ID
2 MAKE
END-DEFINE
FIND EMPL WITH NAME = 'ADKINSON'
IF NO RECORDS FOUND
WRITE 'NO RECORD FOUND'
END-NOREC
FIND (1) VEHI WITH PERSONNEL-ID = EMPL.PERSONNEL-ID
DISPLAY EMPL.PERSONNEL-ID FULL-NAME MAKE
END-FIND
END-FIND
END
```

The same program after being structurally indented:

```
DEFINE DATA LOCAL
1 EMPL VIEW OF EMPLOYEES
  2 PERSONNEL-ID
  2 FULL-NAME
    3 FIRST-NAME
    3 NAME
1 VEHI VIEW OF VEHICLES
  2 PERSONNEL-ID
  2 MAKE
END-DEFINE
FIND EMPL WITH NAME = 'ADKINSON'
  IF NO RECORDS FOUND
    WRITE 'NO RECORD FOUND'
  END-NOREC
  FIND (1) VEHI WITH PERSONNEL-ID = EMPL.PERSONNEL-ID
    DISPLAY EMPL.PERSONNEL-ID FULL-NAME MAKE
  END-FIND
END-FIND
END
```

40 SYSDDM

SYSDDM

This command is used to invoke the DDM Services which offer functions that are needed to create and maintain Natural data definition modules (DDMs).

For further information, see *DDM Services* in the *Editors* documentation.

Note: This command is not executable in batch mode.

Note Concerning Natural Single Point of Development:

This command is not available via Natural Studio's command line in a remote development environment, because DDMs are listed in the tree view under the node DDM and all functions of the SYSDDM utility are available via the context menu or the menu bar.

41 SYSERR

SYSERR

This command is used to invoke the SYSERR utility.

With the SYSERR utility, you can write your own application-specific messages.

- You can use the SYSERR utility to separate error or information messages from your Natural code and manage them separately.
- As well as unifying messages and defining message ranges for different kinds of messages, you can translate messages into another language and attach a long text to a message.
- You can also use the SYSERR utility to modify the texts of existing Natural system messages, although this is not recommended as modifications will be lost with new Natural releases.

For further information, see *SYSERR Utility* in the *Tools and Utilities* documentation.

42 SYSEXT

SYSEXT

This command is used to invoke the SYSEXT utility.

This utility is used to display various Natural application programming interfaces contained in the library SYSEXT.

For further information, see *SYSEXT - Natural Application Programming Interfaces* in the *Tools and Utilities* documentation.

43 SYSEXV

SYSEXV

This command is used to invoke the SYSEXV utility.

The SYSEXV utility gives you access to examples of new features available in the current and in some earlier versions of Natural.

For further information, see *SYSEXV Utility* in the *Tools and Utilities* documentation.

44 SYSFILE

SYSFILE

This command is used to display work and print files information. You can obtain information about the following:

- reports,
- logical devices,
- defined physical devices,
- defined printer profiles, and
- defined workfiles.

For further information on work and print files, see

- *Printer Profiles* in the *Configuration Utility* documentation, and
- *Device/Report Assignments* in the *Configuration Utility* documentation,
- *Work Files* in the *Operations* documentation,

45 SYSLVERS

This command is used to invoke the SYSLVERS utility. The SYSLVERS utility lists objects which have been cataloged within a selected Natural version range.

For further information, see SYSLVERS Utility in the *Tools and Utilities* documentation.

46 SYSMAIN

SYSMAIN

This command is used to invoke the SYSMAIN utility. You use this utility to perform operations such as copy, move and delete on Natural objects. The SYSMAIN utility is also used to transfer objects within the Natural system from one environment to another using the import function.

For further information, see *SYSMAIN Utility* in the *Tools and Utilities* documentation.

Note: This command is not executable in batch mode.

47 SYSNCP

SYSNCP

This command is used to invoke the SYSNCP utility.

For further information, see *SYSNCP Utility* in the *Tools and Utilities* documentation.

48 SYSOBJH

SYSOBJH

This command is used to invoke the Object Handler. You use the Object Handler to process Natural and non-Natural objects for distribution in Natural environments.

For further information, see *Object Handler* in the *Tools and Utilities* documentation.

49 SYSPROD

- About SYSPROD 132

SYSPROD

About SYSPROD

This command is used to ascertain which products are installed at your Natural site. You are given information about your current Natural version and products running with or under Natural.

When you enter the command, a dialog displays information such as the following for each product installed:

- product name
- product version (see also *Version* in the *Glossary*)
- installation date

For some of the products listed, you can get additional information by marking them with a line command in the **Cmd** column of the dialog.

Line Command	Description
EX	Display extended product information.
HI	Display history of product information.
SC	Display subcomponents of product.

Note: For some products, no line commands are allowed.

50 SYSPROF

SYSPROF

This command is used to display the current definitions of the Natural system files.

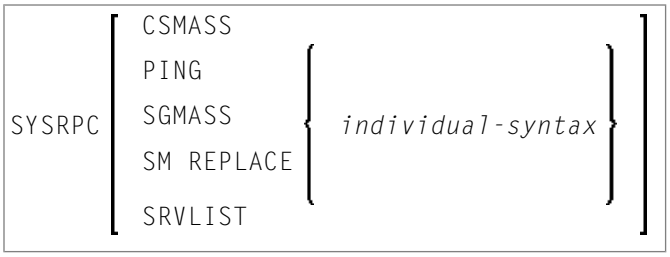
For each system file, the following information is displayed (this corresponds to pressing PF4 when one of the pages listed below is currently shown):

- the file name
- the database ID
- the file number
- the database type

In addition, the following information can be displayed for each defined combination of database ID and file number:

- the path in the file system (when you press PF5)
- the logical file number, if assigned (when you press PF6)

51 SYSRPC



This system command invokes the `SYSRPC` utility which is used to maintain remote procedure calls.

You can specify direct commands with the `SYSRPC` system command in order to perform RPC-specific tasks. These tasks and the *individual-syntax* that applies to the direct commands are described in detail in *SYSRPC Utility* in the *Tools and Utilities* documentation:

Direct Command	Purpose and Related Topics
`CSMASS`	Calculates buffer sizes required for RPC calls. See *Calculating Size Requirements* and *Using the SYSRPC CSMASS Command*.
`PING`	Pings a single or all defined servers. See *Pinging an RPC Server* and *Using the SYSRPC PING Direct Command*.
`SGMASS`	Generates multiple interface objects. See *Generating Multiple Interface Objects* and *Using the SYSRPC SGMASS Command*.
`SM REPLACE`	Replaces single or multiple items in a service directory See *Replacing Items in the Service Directory*.
`SRVLIST`	Provides information on Natural RPC servers registered on EntireX Broker: see *Listing Servers Registered on EntireX Broker*.

For information on how to apply the SYSRPC utility functions to establish a framework for communication between server and client systems, refer to the *Natural RPC (Remote Procedure Call)* documentation.

52 SYSWIZDB

SYSWIZDB

This command is used to invoke the Data Browser, a development tool wizard within Natural Studio. It enables you to display and print or store file structures.

53 SYSWIZDW

```
SYSWIZDW
```

This command is used to invoke the Dialog Wizard, a tool for creating dialogs for specific purposes. The defined dialogs can have several layouts that adapt to desired requirements.

54 TECH

TECH

This command is used to display the following technical and other information about your Natural session:

- user ID
- library ID
- Natural version (see also *Version* in the *Glossary*)
- startup transaction
- Natural Security indicator
- Natural editors: Indicates whether the Natural program, data area, and map editors are disabled or enabled (see *EDITOR system variable)
- operating system name and version
- machine class
- hardware
- IBM architecture level supported on the current IBM processor (z/OS)

  0 (zero) denotes that architecture levels are not supported.
- TP monitor (z/OS and Windows (*TPSYS) in remote configuration only)
- device type
- terminal ID (z/OS and Windows in remote configuration only)
- code page
- locale
- last command issued
- information on the last error that occurred

- names, database IDs, and file numbers of all currently active steplibs
- names, types and levels of the currently active Natural object and all objects on higher levels, as well as the line numbers of the statements invoking the subordinate objects (z/OS and Linux only).

Notes:

1. For character-user-interface applications only: To display this information from any point in an application, you can use the terminal command `%<TECH`.
2. This command is also available in a remote session. All information can be read in batch mode.

Application programming interface: `USR2026N`. See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

55 UNCATALOG

UNCATALOG [*object-name* ...]

This command is used to delete one or more object modules.

You can only delete objects which are stored in the library to which you are currently logged on. The contents of the source work area is not affected by the UNCATALOG command.

UNCAT UNCAT *	If you enter the UNCATALOG command without an *object-name* or with an asterisk (*), a list of all cataloged objects in the current library will be displayed; on the list, you can then mark the object(s) to be deleted.
UNCAT *object name*	As *object-name*, you specify the name of the object to be deleted. If more than one object is to be deleted, the *object-names* must be separated by one or more blanks (or the currently defined delimiter character). If you wish to delete all objects whose names begin with a specific string of characters, use asterisk notation (*) for the *object-name*. A list containing all objects selected will be displayed. On the list, you can then mark the object(s) to be deleted.

Note: If an FDIC system file is specified in the parameter file which is not valid, Natural will display an appropriate error message when the UNCATALOG command is issued.

56 UPDATE

```
UPDATE { ON  }
       { OFF }
```

This command is used to control database updates performed by a Natural program.

`UPDATE ON`	Database updates are allowed.
`UPDATE OFF`	Database updates are not allowed. A database update is not performed when a program with an `UPDATE`, `STORE`, or `DELETE` statement executes. Instead, a NAT1010 warning message is issued during the next screen I/O operation. In addition, a database loop that contains an `UPDATE` or `DELETE` statement does not place the records in hold status (no `read with hold`).

57 WHICH

- Direct Command Syntax 148
- WHICH Parameter Description 148
- Default Output of WHICH Command 148
- Full List of WHICH Command 149
- Batch Processing 150
- WHICH Command under Natural Security 150

```
WHICH [ object-name [FULL] ]
      [          ?         ]
```

This command lets you find objects in the libraries used at run time in the Library Search Order (LSO).

Direct Command Syntax

`WHICH`	Displays a window where you can enter an object name with the additional option to select the full output list.
`WHICH object-name`	Displays the object type, the name of the first library in which the object is found, and the DBID and FNR of this library. See *Default Output of WHICH Command*.
`WHICH object-name FULL`	Lists all libraries of the library search order. For each listed library, it indicates whether the cataloged or source object is found. If you press PF5 in the output screen, you can start a new search. See *Full List of WHICH Command*.
`WHICH ?`	Displays the help text of the `WHICH` command.

WHICH Parameter Description

Parameter	Format/Length	Default Value	Description
object-name	A8	none	The name of the object you want to search.

Default Output of WHICH Command

If you enter the `WHICH` command without the `FULL` option, the output looks like the following example:

```
Cataloged subprogram USR1002N found in library SYSEXT(12,124).
```

Full List of WHICH Command

If you enter the `WHICH` command with the `FULL` option, all libraries of the library search order are listed. The output looks like the following example:

```
12:00:49               ***** NATURAL WHICH COMMAND *****             2023-05-25
User XXXX                 - Search object USR1002N -           Library MYLIB

Library   DBID   FNR Type Save/Catalog Date   Message
-------- ----- ----- ---- ------------------ --------------------------------
MYLIB       12   123
MYLIB01     12   123
MYLIB02     12   123
PROTECT                                     Library access denied
SYSEXT      12   124  N   2022-04-20 07:59:36 Cataloged object found
                      N   2011-05-06 16:37:46 Saved object found
SYSTEM      12   123
SYSTEM      12   124









Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help        Exit        New   --    -     +     ++                Canc
```

Explanation of Column Headings

Column	Explanation
Library	The names of libraries in the library search order.
DBID FNR	The database ID and file number of the Natural system file containing the library.
Type	The Natural object type code of the object. See Object Type Code in the *Glossary*.
Save/Catalog Date	The save or catalog date of the object.
Message	The result of the search. An empty line indicates that the object was not found in that library.

Function Keys

The output screen of the `WHICH` command with `FULL` option provides the following function keys:

PF Key	Name	Function
PF1	Help	Invoke help.
PF3	Exit	Exit from WHICH command.
PF5	New	New search.
PF6	--	Top of list.
PF7	-	Page backward.
PF8	+	Page forward.
PF9	++	End of list.
PF12	Canc	Cancel the WHICH command.

Batch Processing

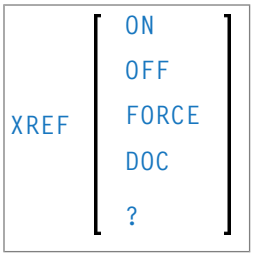
The output of WHICH without FULL option is the same as the online output.

If you call the WHICH command with the FULL option, all resulting libraries are printed in one list.

WHICH Command under Natural Security

For information on WHICH under Natural Security, see *Command Restrictions* in the *Natural Security* documentation.

58 XREF



This command is only available if Predict has been installed. It controls the usage of the Predict function "active cross-references".

The active cross-reference facility automatically creates documentation in the Data Dictionary about the objects with a program/data area reference. These objects include programs, subprograms, subroutines, helproutines, maps, data areas, database views, database fields, user-defined variables, processing rules, error numbers, work files, printers, classes and retained ISN sets.

The active cross-reference is created when a program/data area is cataloged.

To look at cross-reference data, you use the `XREF` option of the system command `LIST`.

For further information on active cross-references, see the Predict documentation.

The following command options are available:

`XREF`	If you enter the `XREF` command without parameters, a menu/dialog is displayed where you specify the desired option.
`XREF ON`	This command activates the active cross-reference function. Cross-reference data will be stored in the respective Predict entries each time a Natural program/data area is cataloged.
`XREF OFF`	This command deactivates the active cross-reference facility. No cross-reference data will be stored. Existing cross-reference data for the object being cataloged will be deleted.

XREF FORCE	The object can only be cataloged if a Predict entry exists for it. When the object is cataloged, its cross-reference data will be stored in Predict. If no Predict entry exists, the object cannot be cataloged.
XREF DOC	The object can only be cataloged if a Predict entry exists for it. However, when the object is cataloged, no cross-reference data will be stored in Predict, and existing cross-reference data for the object will be deleted. If no Predict entry exists, the object cannot be cataloged.
XREF ?	With XREF ? you can call the Help function for the XREF command.

Natural Security Considerations

If Natural Security is installed, the setting for XREF may be set for each library in the library security profile. Depending on the security profile, some options of the XREF command may not be available to you.