

## **Natural for Mainframes**

### **Utilities**

Version 4.2.6 for Mainframes (Update)

February 2010

This document applies to Natural Version 4.2.6 for Mainframes (Update).

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

Copyright © 1979-2010 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, United States of America, and/or their licensors.

The name Software AG, webMethods and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at <http://documentation.softwareag.com/legal/> 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 and license terms, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". This document is part of the product documentation, located at <http://documentation.softwareag.com/legal/> and/or in the root installation directory of the licensed product(s).

## Table of Contents

1 Utilities .....	1
2 Utility Activation .....	3
3 Utilities Grouped by Purpose .....	5
4 ADACALL Utility - Issuing Adabas Direct Calls .....	7
Invoking ADACALL .....	8
ADACALL Parameters .....	9
ADACALL Commands and PF Keys .....	11
User Exit ADAEXIT .....	13
5 DBLOG Utility - Logging Database Calls .....	15
6 Executing DBLOG .....	17
Basic Principles of Database Logging .....	18
Data Processing and Storage .....	19
Activating and Deactivating DBLOG .....	20
Using Selective DBLOG .....	21
7 DBLOG Menu .....	23
DBLOG Menu Functions .....	24
Specifying Logging Restrictions .....	26
Specifying Adabas Buffers .....	26
8 DBLOG Trace Screen .....	29
DBLOG Trace Screen for Adabas Commands .....	30
DBLOG Trace Screen for DL/I Calls .....	37
DBLOG Trace Screen for SQL Statements .....	39
9 DBLOG Snapshot Function .....	43
Snapshot Function for Adabas Commands .....	44
Snapshot Function for DL/I Calls .....	46
Snapshot Function for SQL Statements .....	48
10 TEST DBLOG Command .....	51
Syntax Diagrams .....	52
Keyword Explanations .....	53
11 INPL Utility .....	55
Introducing the INPL Utility .....	56
Load Libraries Only .....	61
Load DDMs Only .....	61
Load Error Messages Only .....	62
Load All Objects .....	63
Replace Product Installation .....	63
Scan INPL File .....	64
Natural Security Recover .....	65
12 NATPAGE Utility - Screen Capturing .....	67
13 NATRJE Utility - Natural Remote Job Entry .....	69
General Information on NATRJE .....	70
Calling NATRJE from a Natural Program .....	71
NATRJE Return Codes .....	75

---

NATRJE Features Applicable to openUTM/TIAM .....	77
14 Object Handler .....	79
15 General Information on the Object Handler .....	81
Principles of Object Transfer .....	82
Invoking the Object Handler .....	83
Batch or Direct Command Calls .....	84
Issuing Object Handler Commands from a Natural Program .....	85
Text Members for Reports, Restarts and Traces .....	86
Natural Security .....	86
Standard PF Keys .....	87
16 Functions .....	89
17 Wizards .....	91
Step 1 - Start the Procedure .....	92
Step 2 - Unload/Load/Scan Objects into/from Work Files .....	93
Step 3 - Set Parameters .....	95
Step 4 - Select Objects .....	96
Step 5 - Execute Processing .....	97
Step 6 - Continue Processing .....	98
18 Advanced User .....	99
Activating Advanced User .....	100
Processing Objects .....	100
19 Restart Load .....	103
20 View .....	105
Natural Library Objects .....	106
Natural System Error Messages .....	107
Natural Command Processor Sources .....	108
FDTs .....	109
Natural-Related Objects .....	109
DDMs .....	110
21 Find .....	113
22 Administration .....	115
List the Available Workplans in the Workplan Library .....	116
Create a New Workplan .....	118
Change the Workplan Library .....	120
23 Select System File .....	121
24 Select Library .....	123
25 Select System Error Messages .....	125
Columns and Commands .....	126
26 Select Objects .....	129
Columns and Commands on List Screens .....	130
27 Object Specification .....	133
28 Object Specification - All Objects on the Work File .....	135
29 Object Specification - Natural Library Objects .....	137
Natural Library Objects .....	138
Natural Library Object Details .....	139

---

Natural Library Object Properties .....	141
Natural Library Object Exceptions .....	142
Natural Library Object Exception Properties .....	143
30 Object Specification - Natural System Error Messages .....	145
Natural System Error Messages .....	146
Natural System Error Message Details .....	146
Natural System Error Message Exceptions .....	147
31 Object Specification - Natural Command Processors .....	149
Natural Command Processors .....	150
Natural Command Processor Source Exceptions .....	151
32 Object Specification - Natural-Related Objects .....	153
Natural Profiles .....	154
Natural Debug Environments .....	155
Natural DL/I Subfiles .....	157
33 Object Specification - DDMs .....	159
DDMs .....	160
DDM Properties .....	161
DDM Exceptions .....	162
34 Object Specification - FDTs .....	163
35 Use Selection or List Workplan .....	165
36 Settings .....	167
Settings Screen Fields .....	168
Set Additional Options .....	170
Set Global Parameters .....	176
37 Workplans .....	181
Creating, Selecting and Modifying Workplans .....	182
Contents of Workplans .....	182
Examples of Workplans .....	183
Referencing Workplans .....	184
38 Name, Date and Time Specification .....	187
Name .....	188
Date .....	189
Time .....	190
39 Work Files .....	191
Work File Assignment .....	192
Work File Format .....	192
40 Direct Commands .....	195
41 Basic Command Syntax .....	197
42 select-clause .....	201
Syntax of select-clause .....	202
SELECTION or LIST Workplan .....	202
Natural Library Object and DDM Selection .....	203
Natural-Related Debug Environment Selection .....	209
Natural-Related Profile Selection .....	210
Natural-Related DL/I Subfile Selection .....	212

---

Natural System Error Message Selection .....	214
Natural Command Processor Selection .....	215
FDT Selection .....	217
Application Selection .....	218
Object Selection for Delete Instructions .....	221
Help Text Selection .....	223
43 Object List - LIST Workplan .....	225
Syntax of object-type-and-location .....	226
Syntax of object-name-description .....	228
Example of an Object List .....	627
44 parameter-setting .....	231
Syntax of parameter-clause .....	232
Keyword Explanation of parameter-clause .....	233
45 option-setting .....	237
Syntax of option-setting .....	238
Keyword Explanation of option-setting .....	240
46 Examples of Using Direct Commands .....	245
Unloading Objects for the Same Platform .....	246
Unloading Objects for Different Platforms .....	247
Loading Objects in Internal Format .....	247
Loading Objects in Transfer Format .....	248
47 Commands for Navigation and Special Functions .....	249
48 Batch Condition Codes and User Exit Routines .....	253
Condition Codes Returned in Batch .....	254
Applying User Exit Routines .....	254
User Exit Routines Available .....	255
49 Tools .....	257
Status .....	258
Last Result .....	258
Traces .....	258
Reports .....	259
50 Profile Settings .....	261
Editing PROFILES .....	262
Parameter Specification .....	263
51 Migration from NATUNLD/NATLOAD and SYSTRANS to the Object Handler ....	267
Converting Individual Commands .....	268
Processing Commands with a User Exit Routine .....	270
Processing SYSTRANS Commands with OBJHAPI .....	270
Unsupported SYSTRANS Options .....	271
52 Recording Utility .....	273
Purpose of Recording .....	274
Data and Functions Recorded .....	274
Recording a Session .....	275
Playing Back a Recording .....	276
Manipulating a Recording .....	278

---

53	SYSAPI Utility - APIs of Natural Add-On Products .....	279
	Prerequisites .....	280
	Invoking and Terminating SYSAPI .....	280
	Listing API Example Programs .....	281
	Performing a Keyword Search .....	282
	Using an API Example Program .....	283
54	SYSBPM Utility - Buffer Pool Management .....	285
55	Invoking and Operating SYSBPM .....	287
	Invoking SYSBPM .....	288
	Online Help .....	289
	SYSBPM Main Menu - Fields, Functions and Commands .....	289
	SYSBPM in a z/OS Parallel Sysplex Environment .....	293
56	List Objects .....	295
	Invoking List Objects .....	296
	Columns and Selection Options .....	297
	PF Keys and Direct Commands .....	306
	Line Commands .....	307
57	Delete Objects .....	309
58	Directory Information .....	311
	Fields for Buffer Pool Objects .....	313
	Fields for BP Cache Objects .....	314
	PF Keys and Direct Commands .....	315
59	Hexadecimal Display .....	317
	PF Keys and Direct Commands .....	318
60	Write to Work File .....	321
61	Display Sorted Extract .....	323
62	Buffer Pool Statistics .....	325
	General Buffer Pool Statistics .....	326
	Buffer Pool Load/Locate Statistics .....	329
	Buffer Pool Fragmentation .....	333
	Internal Function Usage .....	335
	Buffer Pool Hash Table Statistics .....	335
	Performance Hints .....	338
	PF Keys and Direct Commands .....	342
63	BP Cache Statistics .....	345
	General BP Cache Statistics .....	346
	BP Cache Call Statistics .....	348
	BP Cache Hash Table Statistics .....	350
	Performance Hints .....	351
	PF Keys and Direct Commands .....	351
64	Select Buffer Pool .....	353
	Invoking Select Buffer Pool .....	354
	Display Buffer Pools .....	355
	Reset Buffer Pool .....	355
65	Blacklist Maintenance .....	357

---

Maintain Blacklist .....	358
List Object Sets .....	362
Edit Object Set .....	362
Add Object Set to Blacklist .....	365
Delete Object Set from Blacklist .....	366
Delete Object Set Source Object .....	367
Additional Object Set Maintenance with Utilities .....	367
Blacklist Maintenance in Batch Mode .....	368
66 Preload List Maintenance .....	371
List Preload Lists .....	372
Edit Preload List .....	373
Generate Preload List from Buffer Pool .....	376
Delete Preload List .....	377
Additional Maintenance Functions with Utilities .....	377
67 Performance Considerations .....	379
Internal Fast Locate Table .....	380
Searching in Steplibs .....	381
Reusing and Retaining Objects .....	381
Local versus Global Buffer Pool .....	382
68 SYSBPM Direct Commands .....	383
SORT .....	388
SORT BPC .....	389
69 Batch Processing .....	391
Related Topics .....	392
70 Application Programming Interfaces .....	393
71 SYSCP Utility - Code Page Administration .....	395
Invoking and Terminating SYSCP .....	396
Code Page Maintenance of Sources .....	398
All Code Pages .....	410
Unicode Properties .....	415
72 SYSEDT Utility - Editor Buffer Pool Administration .....	417
Defining a Natural Security Library Profile .....	418
Invoking SYSEDT and Executing a Function .....	418
General Information .....	420
Generation Parameters .....	421
Users .....	422
Logical Files .....	423
Recovery Files .....	424
Administration Facilities .....	425
Help on Direct Commands and Menu Functions .....	425
73 SYSERR Utility .....	427
74 General Information on Messages .....	429
Message Types .....	431
Message Languages .....	431
Issuing Messages .....	432

---

Retrieving Natural System Short Messages .....	433
Retrieving User-Defined Short Messages .....	433
Obtaining Message Information .....	434
75 Invoking SYSERR .....	435
76 Functions .....	437
Adding Messages .....	438
Deleting Messages .....	442
Displaying Messages .....	442
Modifying Messages .....	444
Printing Messages .....	446
Scanning Messages .....	447
Selecting Messages from a List .....	450
Translating Messages into other Languages .....	452
77 Parameters .....	455
Message Type .....	456
Library .....	456
Message Number .....	456
Language Codes .....	456
78 Direct Commands .....	457
79 Upper Case Conversion - ERRUPPER .....	459
80 Replacing Characters - ERRCHAR .....	461
81 Managing Messages in Different Libraries .....	463
Unloading Messages - ERRULDUS .....	464
Loading Messages - ERRLODUS .....	466
82 Application Programming Interface USR0020P .....	467
83 SYSEXT Utility - Natural Application Programming Interfaces .....	469
Basic Features of SYSEXT .....	470
Invoking and Terminating SYSEXT .....	472
SYSEXT Utility Screen .....	473
Performing SYSEXT Utility Functions .....	474
Using a Natural API .....	478
List of Natural APIs .....	478
84 SYSEXV Utility .....	485
Executing Example Programs of Current Versions .....	486
Executing Example Programs of Non-current Versions .....	486
Terminating the SYSEXV Utility .....	487
85 SYSMAN Utility - Object Maintenance .....	489
86 General Information on SYSMAN .....	491
Basic SYSMAN Functionality .....	492
Object Types and Storage Location .....	493
Overview of Functions .....	493
87 Invoking and Terminating SYSMAN .....	495
Invoking SYSMAN Online or Batch .....	496
Invoking SYSMAN with Appl. Programming Interface .....	497
Terminating SYSMAN .....	498

---

88 Using Menu Functions and Commands .....	501
Performing Menu Functions .....	502
Executing Commands .....	505
Description of Functions .....	506
Function Processing and Reporting .....	514
SYSMAIN Online Help .....	519
89 Processing Programming Objects .....	521
Fields in Programming Objects Menus .....	522
Using Profile Parameter RECAT .....	525
Selection Lists for Programming Objects .....	525
XRef Considerations .....	529
Specifying Additional Criteria .....	531
Direct Command Syntax for Programming Objects .....	532
90 Processing Debug Environments .....	537
Fields in the Debug Environments Menu .....	538
Selection Lists for Debug Environments .....	540
Direct Command Syntax for Debug Environments .....	541
91 Processing Error Messages .....	545
Fields in Error Message Menus .....	546
Selection Lists for Error Messages .....	548
Renumbering Error Messages .....	550
Specifying Languages .....	551
Direct Command Syntax for Error Messages .....	551
92 Processing Profiles .....	555
Fields in the Profiles Menu .....	556
Selection Lists for Profiles .....	558
Direct Command Syntax for Profiles .....	559
93 Processing Rules .....	563
Fields in the Rules Menu .....	564
Selection Lists for Rules .....	565
Direct Command Syntax for Rules .....	567
94 Processing DL/I Subfiles .....	571
Fields in the DL/I Subfiles Menu .....	572
Selection Lists for DL/I Subfiles .....	573
Direct Command Syntax for DL/I Subfiles .....	574
95 Processing DDMs .....	577
Fields in the DDMs Menu .....	578
Selection Lists for DDMs .....	580
Direct Command Syntax for DDMs .....	581
96 Keywords and Variables in Direct Commands .....	585
Description of Keywords .....	586
Specifying a Range of Names .....	596
97 Special Commands Issued to SYSMAIN .....	599
98 Processing Status and Error Notification .....	603
Object Rejection and Reasons .....	604

---

Status Messages .....	605
SYSMAIN Error Notification .....	609
99 Special Considerations for Administrators .....	613
File Security .....	614
Natural Security .....	615
User Exit Routines .....	616
100 SYSNCP Utility .....	623
Introducing the SYSNCP Utility .....	624
Invoking SYSNCP .....	631
Processor Selection .....	632
Header Records .....	633
Keyword Maintenance .....	642
Function Maintenance .....	647
Runtime Actions .....	652
Processor Cataloging .....	657
Administrator Services .....	658
Session Profile .....	665
101 SYSPARM Utility .....	669
Invoking SYSPARM .....	670
List Profiles .....	671
Display Profile .....	672
Add New Profile .....	674
Modify Profile .....	674
Editing Profiles .....	675
Copy Profile .....	678
Delete Profile .....	679
Direct Commands and Batch Processing .....	679
Maintaining Profiles in Different Environments .....	687
102 SYSRDC Utility .....	689
Functional Components of SYSRDC .....	690
Data-Collecting Events .....	691
Data Collected .....	693
Activating the Natural Data Collector .....	695
Trace Recording .....	696
User Exits for External Monitoring/Accounting .....	697
Calling the CMRDC Interface .....	698
103 SYSRPC Utility .....	705
104 Invoking and Terminating SYSRPC .....	707
Invoking SYSRPC .....	708
Terminating SYSRPC .....	709
Invoking Online Help .....	709
105 Service Directory Maintenance .....	711
Service Directory Concept .....	712
Invoking Service Directory Maintenance .....	714
Fields on the Service Directory Screen .....	716

---

Commands for Service Directory Maintenance .....	718
Defining Logical Node Names and Logical Services .....	721
106 Stub Generation .....	723
107 Generating Single Stubs with Parameter Specification .....	725
Using the Stub Generation Function .....	726
Specifying Parameters .....	729
Examples of Stub Generation .....	731
108 Generating Multiple Stubs .....	735
Using the SYSRPC SGMASS Command .....	736
Name Specification and Compression .....	737
109 Calculating Size Requirements .....	739
Using the SYSRPC CSMASS Command .....	740
Name Specification and Compression .....	742
110 Parameter Maintenance .....	743
Invoking Parameter Maintenance .....	744
Specifying NTRPC/RPC Keyword Subparameters .....	744
111 Server Command Execution .....	747
Using Server Command Execution .....	748
Pinging a Server .....	749
Terminating a Server .....	751
112 Remote Directory Maintenance .....	755
Using Remote Directory Maintenance .....	756
Fields on the Editor Screen of a Remote Directory .....	757
Commands for Remote Directory Maintenance .....	758
113 SYSTP Utility .....	761
114 Invoking SYSTP and Executing Functions .....	763
115 Using SYSTP Utility Screens .....	765
116 General SYSTP Functions .....	767
Natural Monitoring (SYSMON) .....	768
Natural Print/Work Files (SYSFILE) .....	771
Natural Swap Information .....	772
Buffer Usage Statistics (BUS) .....	776
Natural Sub-Systems and Roll Server Information .....	778
Natural Thread Usage Statistics .....	779
Natural License Information .....	781
117 SYSTP Functions under CICS .....	783
Natural User Sessions .....	784
Natural Roll Facilities .....	789
Natural Thread Groups .....	790
Natural Storage Threads .....	791
NCI Global System Information .....	792
NCI Generation Options .....	794
Natural Thread Group Definitions .....	794
Own Natural User Session .....	795
CICS Task Information .....	795

---

System Administration Facilities .....	796
118 SYSTP Functions under IMS TM .....	799
Broadcasting .....	800
Display Environment Data .....	800
Monitoring .....	801
Multi Session .....	801
Applied NII Zaps .....	801
119 SYSTP Functions under TIAM and UTM .....	803
P-Key Utility .....	804
Show Common Memory Pools .....	808
120 SYSTP in Batch for CICS Sessions .....	809
Invoking SYSTP in Batch Mode .....	810
Evaluating the Log File .....	810
Index .....	813



# 1 Utilities

---

The *Utilities* documentation explains how Natural invokes a utility and describes the utilities available in Natural.

 <b>Utility Activation</b>	Describes how Natural invokes a utility.
 <b>Utilities Grouped by Purpose</b>	Lists all utilities grouped according to their purpose.
 <b>ADACALL</b>	This utility issues Adabas direct calls (native commands) directly to an Adabas database.
 <b>DBLOG</b>	This utility logs database calls: indicates which Adabas commands, DL/I calls or SQL statements are issued by a Natural program.
 <b>INPL</b>	This utility loads or scans Natural objects supplied by Software AG.
 <b>NATPAGE Screen Capturing</b>	This utility captures screens (maps and reports) during a Natural session.
 <b>NATRJE</b>	This utility submits JCL cards from a Natural program to the operating system for scheduling and execution.
 <b>Object Handler</b>	This utility processes Natural and non-Natural objects for distribution in Natural environments. This is done by unloading the objects in the source environment into work files and loading them from work files into the target environment.
 <b>Recording</b>	This utility records commands and input data entered during a Natural session. Re-executes a recorded session.
 <b>SYSAPI</b>	This utility locates Application Programming Interfaces (APIs) provided by Natural add-on products.
 <b>SYSBPM</b>	This utility monitors and controls the Natural buffer pool.
 <b>SYSCP</b>	This utility provides code page information and can be used to administrate code pages for Natural source objects.

 <b>SYSEDT</b>	This utility displays parameters and runtime information for the editor buffer pool. Modifies parameters and deletes logical work and recovery files.
 <b>SYSERR</b>	This utility creates application-specific messages. In addition, it can be used to modify the texts of the existing Natural system messages (not recommended).
 <b>SYSEXT</b>	This utility locates Natural Application Programming Interfaces (APIs).
 <b>SYSEXV</b>	This utility contains examples of the new features of the current Natural versions.
 <b>SYSMAIN</b>	This utility performs object operations in Natural such as copy, move and delete.
 <b>SYSNCP</b>	This utility defines command-driven navigation systems for Natural applications.
 <b>SYSPARM</b>	This utility creates and maintains Natural parameter profiles.
 <b>SYSRDC</b>	This utility records monitoring and accounting data on the processing flow within a Natural application.
 <b>SYSRPC</b>	This utility establishes and maintains Natural Remote Procedure Call environments.
 <b>SYSTP</b>	This utility monitors and controls TP-monitor-specific characteristics of Natural.

## 2 Utility Activation

---

Natural invokes a Natural utility without performing a logon to the corresponding utility library in the FNAT system file. As a result, Natural preserves the global data area (GDA) and/or application-independent variables (AIV). The current user library and the settings are maintained. (To reset the GDA and/or the AIVs, see the profile parameter `FREEGDA` in the *Parameter Reference*.)

To preserve the settings of your application environment, do *not* log on to a utility library. Instead, invoke a utility by using the Natural system command that corresponds to the utility.

After terminating a utility, you will be returned to the library from which you invoked the utility. However, if you explicitly log on to a utility library before invoking the utility, you will stay in this (utility) library after utility termination.

### **Exception:**

The SYSEXT utility and the SYSEXV application still perform an implicit logon to the corresponding utility library since object sources can only be edited within an active library.

For information on how to control the use of Natural utilities with Natural Security, see the section *Protecting Utilities* in the *Natural Security* documentation.

If Natural Security is not installed, you can control the use of Natural utilities with user exit routine UTI-EX01. The program source for this user exit routine is provided as source object UTI-SX01 in library SYSEXT.

### **▶ To activate UTI-EX01**

- 1 CATALOG or STOW source object UTI-SX01 under the name UTI-EX01.

Different names are used to guarantee that the source object (possibly modified according to your requirements) and the cataloged object of the user exit routine are not overwritten by an update installation.

2 Copy UTI-EX01 to library SYSTEM in the FNAT or the FUSER system file.

For a detailed description of the user exit routine, see the source object of UTI-SX01 in the library SYSEXT.

# 3 Utilities Grouped by Purpose

---

The following is a list of all Natural utilities grouped according to their purpose:

Administration	Debugging	Monitoring	Object Transfer
SYSAPI	ADACALL	SYSBPM	INPL
SYSCP	DBLOG	SYSEDT	Object Handler
NATRJE	Debugger	SYSRDC	SYSMAIN
SYSBPM	DUMP	SYSTP	
SYSEDT	NATPAGE Screen Capturing		
SYSERR	Recording		
SYSEXT	SYSRDC		
SYSNCP			
SYSPARM			
SYSRPC			
SYSTP			

---

# 4 ADACALL Utility - Issuing Adabas Direct Calls

---

- Invoking ADACALL ..... 8
- ADACALL Parameters ..... 9
- ADACALL Commands and PF Keys ..... 11
- User Exit ADAEXIT ..... 13

The utility ADACALL can be used to issue Adabas direct calls (native commands) to an Adabas database for learning and testing and for analyzing problems.

The utility ADACALL is contained in the library SYSADA.

## Invoking ADACALL

---

► **To invoke ADACALL**

- Enter the following system command:

```
SYSADA
```

An ADACALL main screen similar to the example screen below is displayed:

```
15:53:32          ***** NATURAL ADACALL UTILITY *****          2006-12-14
User SAG          - ADABAS Direct Calls -
Mode Char
*** Control Block ***          First Byte 30          Call No. 45
  Cmd L3          Cmd ID SAG          File 316          Database 10
Resp 0          ISN 382          ISQ 0          ISL 0
  FBL 210          RBL 980          SBL 140          VBL 140          IBL 0
COP1          COP2          User Area          Cmd Time 4
Addition1          Addition2          Addition3          Addition4          Addition5
AAJ?          227 48
*** Buffer Areas ***
Format AA,AC,AE.

Record 11111003ARTHUR          DENT

Search

Value

ISN
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Main Exit          Char Hex View Prnt Run Init Canc
```

On the ADACALL main screen, specify the necessary parameter values and execute the Adabas command by either choosing PF10 (Run) or entering the ADACALL command EXEC in the Command line.

In the example screen above, the Adabas command L3 was executed for a logical read of the employees file.

Except for the control block, which is shown in full, only a part of the buffer is displayed. You can view the buffers in their entirety by using any of the ADACALL direct commands or PF keys listed below.

## ADACALL Parameters

The parameters which can be specified on the ADACALL main screen are listed below. You can use the ADACALL online help function to obtain a summarized explanation of the parameters.

### ► To invoke the online help function

- Place the cursor in the field for which you require help and enter a question mark (?) or choose PF1. (For read-only fields, only PF1 applies.)

For detailed information, see the Adabas documentation *Command Reference and Messages and Codes*.

Parameter	Explanation				
<b>Mode</b>	Indicates the display mode of the buffer contents: <table border="1" data-bbox="381 1024 1153 1255"> <tr> <td>Char</td> <td>Character values.</td> </tr> <tr> <td>Hex</td> <td>Hexadecimal values.</td> </tr> </table> <p>To change modes, see the ADACALL commands <a href="#">CHAR</a> and <a href="#">HEX</a>.</p>	Char	Character values.	Hex	Hexadecimal values.
Char	Character values.				
Hex	Hexadecimal values.				
<b>Call No.</b>	Number of commands executed since the start of the session.				
<b>First Byte</b>	The first byte of the Adabas control block.  Indicates whether 1-byte or 2-byte database IDs (DBID) and file numbers (FNR) are used:  H'00' = 1-byte DBID, FNR (file numbers 1 - 255) H'30' = 2-byte DBID, FNR (file numbers greater than 255)				
<b>Cmd</b>	Adabas command.  Enter and execute the <a href="#">Adabas OP command</a> to specify the parameters described in the relevant section below.				
<b>Cmd ID</b>	Command ID.				
<b>File</b>	File number.  If <b>First Byte</b> is set to H'00': 3-digit file number, <b>Database</b> not equal to 0.				

Parameter	Explanation
	If <b>First Byte</b> is set to H'30': 5-digit file number.
<b>Database</b>	Database ID (DBID). Defaults to the DBID of the FUSER file of the current Natural session (see <b>File</b> above).  If <b>First Byte</b> is set to H'30', then the database number will be moved to the response code field of the Adabas control block at execution time.
<b>Resp</b>	Response code returned after the command is executed.
<b>ISN</b>	Internal sequence number.
<b>ISQ</b>	ISN quantity.
<b>ISL</b>	Lowest ISN value for ISN lists.
<b>FBL</b>	Format buffer length in bytes (maximum 210).
<b>RBL</b>	Record buffer length in bytes (maximum 980).
<b>SBL</b>	Search buffer length in bytes (maximum 140).
<b>VBL</b>	Value buffer length in bytes (maximum 140).
<b>IBL</b>	ISN buffer length in bytes (maximum 200).
<b>COP1</b>	Command option 1.
<b>COP2</b>	Command option 2.
<b>User Area</b>	User area for the control block.
<b>Cmd Time</b>	The time taken to execute the command, converted to 1/100th seconds for convenience.
<b>Addition1</b>	Additions 1.
<b>Addition2</b>	Additions 2. If the call was successful, it displays the compressed length of the record being read and the decompressed length of the data requested via the format buffer. If a non-zero response is returned and the error was a result of an invalid format buffer, the field in error and its offset into the format buffer are displayed.
<b>Addition3</b>	Additions 3.
<b>Addition4</b>	Additions 4. If a VSAM file is being read, this field is set to VSAM if initialized.
<b>Addition5</b>	Additions 5.
<b>Format</b>	Format buffer. (The final period is necessary.)
<b>Record</b>	Record buffer.
<b>Search</b>	Search buffer. (The final period is necessary.)
<b>Value</b>	Value buffer.
<b>ISN</b>	ISN buffer.

## Adabas OP Command

When you execute the Adabas command `OP` (Open), ADACALL provides a window where you can specify the following parameters:

- maximum ISNs to be stored in the internal ISN buffer,
- maximum records permitted in hold status,
- maximum CIDs (command IDs) which may be active,
- maximum time permitted for execution of an `Sx` command.

In the window, enter the relevant information and choose `ENTER`.

For an explanation of the parameters and valid values, refer to the *Adabas Command Reference* documentation.

## ADACALL Commands and PF Keys

The ADACALL direct commands listed below are provided to change ADACALL parameter settings or to switch between screens by either entering a command in the Command line or choosing a corresponding PF key.

In addition to ADACALL commands, from the Command line, you can also issue Natural system commands.

In the following table, an underlined portion of a command represents an acceptable abbreviation.

Command	PF Key	Function
	PF1	Invoke the help function for ADACALL. If the cursor is positioned on one of the various ADACALL parameters and PF1 is pressed, help information on this parameter is displayed.
	PF2	Return to the ADACALL main screen. <b>Mode</b> is set to <code>CHAR</code> .
<u>BACK</u>	PF5	Page backward to the previous buffer when viewing the buffers in their entirety.  Valid only after the <code>VIEW</code> command has been applied, which means that the command is not applicable from the ADACALL main screen.
<u>CB</u>		Display the control block buffer entirely; valid in hexadecimal mode only.
<u>CHAR</u>	PF6	Change the current mode to character mode (EBCDIC).
<u>D</u>		Display extended error message text for response code received. When an Adabas response other than 0 (zero) is returned, the corresponding short error message text is displayed in the message line. The extended text can be viewed by issuing this command.
<u>EXEC</u> or <u>or</u>	PF10	Execute the direct command with the parameters specified.

Command	PF Key	Function														
RUN																
EXIT or STOP or Q or .	PF3 or PF12	Exit. If pressed while on the ADACALL main screen, ADACALL is terminated. If one of the buffer screens is being viewed, the ADACALL main screen is displayed with <b>Mode</b> unchanged.														
FB		Display the format buffer in its entirety.														
FWD	PF4	Page forward to the next buffer when viewing the buffers in their entirety.  Valid only after the VIEW command has been applied, which means that the command is not applicable from the ADACALL main screen.														
HEX	PF7	Change the current mode to hexadecimal.														
IB		Display the ISN buffer in its entirety.														
INIT	PF11	<p>Initialize/reset buffer(s). A window is displayed and one of the following values can be entered for the buffers indicated:</p> <table border="1"> <tbody> <tr> <td></td> <td></td> </tr> <tr> <td>H</td> <td>Initialize the corresponding buffer(s) with binary zeroes (H'00').</td> </tr> <tr> <td></td> <td></td> </tr> <tr> <td>any character except H or blank</td> <td>Initialize the corresponding buffer(s) with blanks (H'40').</td> </tr> <tr> <td></td> <td></td> </tr> <tr> <td>blank character</td> <td>Do not initialize the corresponding buffer(s).</td> </tr> <tr> <td></td> <td></td> </tr> </tbody> </table> <p>If you enter INIT ALL, all buffers except the control block are initialized with blanks. Alternatively, the command INIT FB RB SB VB IB (not all buffers need be listed) can be specified and all buffers in the list are initialized with blanks.</p> <p><b>Note:</b> The ISN buffer is always initialized with binary zeroes.</p>			H	Initialize the corresponding buffer(s) with binary zeroes (H'00').			any character except H or blank	Initialize the corresponding buffer(s) with blanks (H'40').			blank character	Do not initialize the corresponding buffer(s).		
H	Initialize the corresponding buffer(s) with binary zeroes (H'00').															
any character except H or blank	Initialize the corresponding buffer(s) with blanks (H'40').															
blank character	Do not initialize the corresponding buffer(s).															
PRINT	PF9	<p>Generate and display a report on the status of all buffers.</p> <p>The Natural terminal command %H can be used to obtain a hardcopy.</p>														
RB		Display the record buffer in its entirety.														
RUN		Same as EXEC.														
SB		Display the search buffer in its entirety.														

Command	PF Key	Function												
VB		Display the value buffer in its entirety.												
VIEW	PF8	<p>Display all buffers in their entirety. The first buffer to be displayed is the record buffer. The FWD command can be used to page through the other buffers.</p> <p>If you <b>VIEW</b> the record buffer in hexadecimal mode, the data are displayed on four pages:</p> <p>To page forwards, enter the command <b>FWD</b> or choose PF4.  To page backwards, enter the command <b>BACK</b> or choose PF5.  To display a specific page, enter a page number from 1 to 4 in the field <b>Specify next page number</b>.</p> <hr/> <p>To view buffers individually, enter any of the following commands:</p> <hr/> <table border="1"> <tbody> <tr> <td>FB</td> <td>Format buffer</td> </tr> <tr> <td>RB</td> <td>Record buffer</td> </tr> <tr> <td>SB</td> <td>Search buffer</td> </tr> <tr> <td>VB</td> <td>Value buffer</td> </tr> <tr> <td>IB</td> <td>ISN buffer</td> </tr> <tr> <td>CB</td> <td>Control block (default). Valid in hexadecimal mode only: change to HEX before executing VIEW.</td> </tr> </tbody> </table>	FB	Format buffer	RB	Record buffer	SB	Search buffer	VB	Value buffer	IB	ISN buffer	CB	Control block (default). Valid in hexadecimal mode only: change to HEX before executing VIEW.
FB	Format buffer													
RB	Record buffer													
SB	Search buffer													
VB	Value buffer													
IB	ISN buffer													
CB	Control block (default). Valid in hexadecimal mode only: change to HEX before executing VIEW.													
VSAM		If VSAM has been defined for the current Natural session, this direct command can be issued to access or update VSAM files. When you issue this command, you are prompted by a window for the VSAM file name. When the command is executed, it is directed to the appropriate VSAM file.												

## User Exit ADAEXIT

ADACALL allows direct commands to be issued to any database. Therefore, as a means of security, a user exit is supplied. This user exit is called ADAEXIT and is contained in the library SYSADA. You can modify ADAEXIT as required. The Adabas control block is passed as a parameter to ADAEXIT. You can change the source code of the user exit so as to modify the contents of the control block. By simply changing the database ID or file number, or by setting the Command Code to XX, you can prevent database calls from being performed.



# 5

## DBLOG Utility - Logging Database Calls

---

The DBLOG utility is used to log Adabas commands, or DL/I and SYNC/ROLB calls, or SQL statements. Logging is useful for tuning an application (controlling the flow of commands accessing the database) and for analyzing error codes that may be returned from the database.

- [Executing DBLOG](#)
- [DBLOG Menu](#)
- [DBLOG Trace Screen](#)
- [DBLOG Snapshot Function](#)
- [TEST DBLOG Command](#)



# 6 Executing DBLOG

---

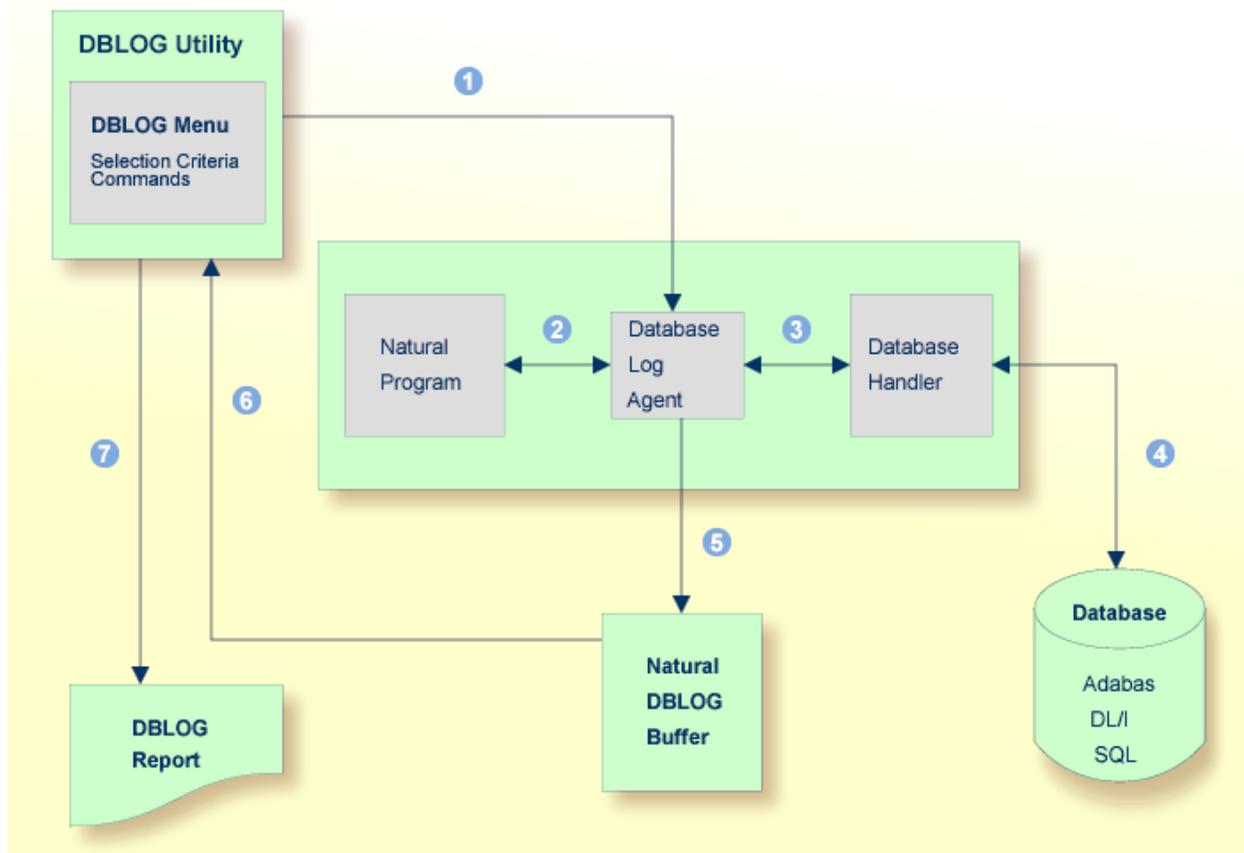
- Basic Principles of Database Logging ..... 18
- Data Processing and Storage ..... 19
- Activating and Deactivating DBLOG ..... 20
- Using Selective DBLOG ..... 21

The DBLOG utility logs each Adabas command, or DL/I and SYNC/ROLB call, or SQL statement after it has been processed by the database system. Logging starts when you activate DBLOG and execute or run a Natural program.

This section covers the following topics:

## Basic Principles of Database Logging

The following graphic illustrates the process flow when database calls are being logged from a Natural program with the DBLOG utility:



## Legend

- 1 The logging of database calls (Adabas commands, or DL/I and SYNC/ROLB calls, or SQL statements) is activated by using the corresponding **DBLOG Menu** function or the Natural system command `TEST DBLOG`.  
In the **DBLOG Menu**, you can specify individual selection criteria (for example, restrict the logging to a particular database ID or file number).
- 2 A Natural program issues a statement that initiates a database call, for example, `FIND` or `READ`.
- 3 The database log agent forwards the database call to the database handler.
- 4 The database handler adapts the database call to the particular database (Adabas, DL/I or SQL), retrieves the data requested by the database call and returns this data to the database log agent.
- 5 The database log agent records in the Natural DBLOG buffer the data returned by the database handler and forwards this data to the Natural program.
- 6 The report function of the DBLOG utility reads the data recorded in the Natural DBLOG buffer and selects records according to the selection criteria specified in the **DBLOG Menu** in 1.
- 7 From the data records selected, the report function of the DBLOG utility generates a report that can be displayed, written to a work file or used for batch processing.

## Data Processing and Storage

The data logged by the DBLOG utility is recorded in the Natural DBLOG buffer. The initial and maximum sizes of the buffer are determined by the `DSIZE` profile parameter described in the *Parameter Reference* documentation.

If there is not enough space to hold the data in the DBLOG buffer, Natural increases the DBLOG buffer size appropriately (possibly several times) until the maximum size specified with of the `DSIZE` profile parameter is reached. After the maximum is reached, only the most recent log data is held in the Natural DBLOG buffer.

If the DBLOG buffer size cannot be further increased due to lack of storage, Natural issues a NAT7545 message that indicates insufficient space for the DBLOG buffer. All data logged prior to this lack-of-storage situation will be kept in the Natural DBLOG buffer and can be displayed by using the `TEST DBLOG` command.

DBLOG can be used online or in batch mode. DL/I and SYNC/ROLB calls can be logged under CICS, under IMS TM or in batch mode. For further information on batch-mode processing, refer to *Natural in Batch Mode* described in the *Operations* documentation.

The logs recorded are displayed on the **DBLOG Trace** screen.

The DBLOG utility provides default settings for data recording. When using the **DBLOG Menu**, you can specify selection criteria for the commands, calls or statements to be logged and the information displayed. The **DBLOG Menu** also provides functions for activating or deactivating logging. You can also use the Natural system command `TEST DBLOG` to control DBLOG execution.

The fields of the **DBLOG Trace screen**, the **DBLOG Menu** and the Natural system command `TEST DBLOG` are explained in the relevant sections of the DBLOG documentation.

## Activating and Deactivating DBLOG

---

The commands used to activate or deactivate DBLOG with the default DBLOG utility settings are described in the following section. See also *TEST DBLOG Command* for additional information.

### ▶ To activate or deactivate DBLOG for Adabas

- Enter the following Natural system command (toggle command):

```
TEST DBLOG
```

Or:

Enter the following to activate:

```
TEST DBLOG ON
```

Enter the following to deactivate:

```
TEST DBLOG OFF
```

Or:

In the **DBLOG Menu**, enter function code B (to activate) or function code E (to deactivate).

### ▶ To activate or deactivate DBLOG for DL/I

- Enter the following Natural system command (toggle command):

```
TEST DBLOG D
```

Or:

Enter the following to activate:

```
TEST DBLOG D ON
```

Enter the following to deactivate:

```
TEST DBLOG D OFF
```

Or:

In the **DBLOG Menu**, enter function code B (to activate) or function code E (to deactivate).

#### ▶ To activate or deactivate DBLOG for SQL

- Enter the following Natural system command (toggle command):

```
TEST DBLOG Q
```

Or:

Enter the following to activate:

```
TEST DBLOG Q ON
```

Enter the following to deactivate:

```
TEST DBLOG Q OFF
```

Or:

In the **DBLOG Menu**, enter function code B (to activate) or function code E (to deactivate).

## Using Selective DBLOG

The following are example instructions for logging Adabas commands, DL/I calls or SQL statements with selection criteria specified in the **DBLOG Menu**.

#### ▶ To perform DBLOG with selection criteria

- 1 Invoke the **DBLOG Menu** by entering one of the following Natural system commands:

- For Adabas:

```
TEST DBLOG MENU
```

- For DL/I:

```
TEST DBLOG D MENU
```

- For SQL:

```
TEST DBLOG Q MENU
```

The **DBLOG Menu** appears.

- 2 In the **DBLOG Menu**, specify logging restrictions and activate logging: complete the input fields and enter function code B.

The message `DBLOG started now` is displayed.

- 3 Execute a Natural program which contains Adabas commands, DL/I calls or SQL statements.
- 4 Invoke the **DBLOG Trace** screen and deactivate logging by entering one of the following Natural system commands:

- For Adabas:

```
TEST DBLOG
```

- For DL/I:

```
TEST DBLOG D
```

- For SQL:

```
TEST DBLOG Q
```

The **DBLOG Trace** screen appears.

- 5 Clear the Natural DBLOG buffer and deactivate logging by entering one of the following Natural system commands:

- For Adabas:

```
TEST DBLOG OFF
```

- For DL/I:

```
TEST DBLOG D OFF
```

- For SQL:

```
TEST DBLOG Q OFF
```

DBLOG terminates and the NEXT prompt appears.

See also the section [TEST DBLOG Command](#) for additional information.

# 7 DBLOG Menu

---

▪ DBLOG Menu Functions .....	24
▪ Specifying Logging Restrictions .....	26
▪ Specifying Adabas Buffers .....	26

In the **DBLOG Menu**, you can activate or deactivate logging and specify which Adabas commands, DL/I calls or SQL statements are to be logged.

► **To invoke the DBLOG Menu**

- Enter one of the following Natural system commands:

- For Adabas:

```
TEST DBLOG MENU
```

- For DL/I:

```
TEST DBLOG D MENU
```

- For SQL:

```
TEST DBLOG Q MENU
```

This section covers the following topics:

## DBLOG Menu Functions

---

The functions provided in the **DBLOG Menu** are described in the following section. You can execute a function by either entering the code that corresponds to the required function in the **Code** field or pressing the PF key that corresponds to the required function.

Code or PF Key	Function	Explanation
B or PF4	<b>Begin Logging of Adabas Commands</b>	<p>Activates the DBLOG logging of the Adabas commands, DL/I calls or SQL statements that match the selection criteria.</p> <p>See also alternative commands in <i>TEST DBLOG Command</i>.</p> <p>See also <b>Optional Buffers for Codes B and W</b>.</p>
E or PF5	<b>End and Display Log Records</b>	<p>Deactivates logging and displays the <b>DBLOG Trace</b> screen of the current log record if data exists in the Natural DBLOG buffer. Current log data is kept in the Natural DBLOG buffer.</p> <p>See also alternative commands in <i>TEST DBLOG Command</i>.</p>

Code or PF Key	Function	Explanation
S or PF6	<b>Snapshot of Specific Adabas Commands</b>	<p>Adabas: Interrupts a program at a specified Adabas command and displays detailed information on this command only: see <a href="#">Snapshot Function for Adabas Commands</a>.</p> <p>DL/I and SQL: Collects detailed information on a specified DL/I call or SQL statement: see Snapshot Function for <a href="#">Snapshot Function for DL/I Calls</a> and <a href="#">Snapshot Function for SQL Statements</a>.</p>
W or PF7	<b>Write Log Records to Work File</b>	<p>Writes the logged data contained in the Natural DBLOG buffer to a work file. The file structure (columns and log sequence) corresponds to the structure of the <b>DBLOG Trace</b> screen described in <a href="#">DBLOG Trace Screen</a>.</p> <p>When you choose this function, a <b>Work File Selection</b> window prompts you for the following information:</p> <ul style="list-style-type: none"> <li>■ Specify the target work file: Enter N (No; this is the default setting) to output the data on Natural Work File 1.</li> <li>Or: If Entire Connection is installed, enter Y (Yes) to output the data on a PC text file by using Natural Work File 7.</li> <li>■ Specify whether to write data logged for the Adabas control block to the work file: Enter N (No; this is the default setting) to include the data.</li> <li>Or: Enter Y (Yes) to exclude the data.</li> </ul> <p>See also <a href="#">Optional Buffers for Codes B and W</a>.</p>
	<b>Optional Buffers for Codes B and W</b>	<p>Only applicable to Adabas commands.</p> <p>Selects additional Adabas buffers to be logged when using function code B or W: see <a href="#">Specifying Adabas Buffers</a>.</p>
PF3		Exits the <b>DBLOG Menu</b> . The current log records are kept in the Natural DBLOG buffer.
PF12		Clears the Natural DBLOG buffer, leaves the <b>DBLOG Menu</b> and returns to the NEXT prompt.

## Specifying Logging Restrictions

This section describes the input fields the **DBLOG Menu** provides for specifying selection criteria to restrict logging:

Field	Explanation
<b>Skip</b>	Only applicable with function code S. Number of commands, calls or statements to be skipped before logging is to start.
<b>Program</b>	Restricts logging to commands, calls or statements issued by the program specified.
<b>DBID</b>	Only applicable to Adabas commands. Restricts logging to commands issued for the database ID specified.
<b>FNR</b>	Only applicable to Adabas commands. Restricts logging to commands issued for the file number specified.
<b>Line from Line to</b>	Restricts logging to commands, calls or statements within the range of the source line numbers specified.
<b>Low Resp High Resp</b>	Only applicable to Adabas commands. Restricts logging to commands which result in a response code within the range specified.
<b>Low Stat High Stat</b>	Only applicable to DL/I calls. Restricts logging to calls which result in a status code within the range specified.
<b>Low SQLC High SQLC</b>	Only applicable to SQL statements. Restricts logging to statements which result in an SQL return code within the range specified.

## Specifying Adabas Buffers

Only applicable to Adabas commands.

The Adabas control block is logged by default. It is either the classic Adabas control block (ACB) or the extended Adabas control block (ACBX) depending on the command executed. For detailed information on Adabas control blocks, see *Adabas Control Block Structures (ACB and ACBX)* in the *Adabas for mainframes* documentation.

In addition to the control block, you can log one or more Adabas buffers listed in the **DBLOG Menu**:

<b>FB</b>	Format buffer
<b>RB</b>	Record buffer
<b>SB</b>	Search buffer
<b>VB</b>	Value buffer
<b>IB</b>	ISN buffer

You enable logging of these buffers and specify the range of bytes to be logged by using the input fields below the buffer names:

#### ■ Bytes 0-79

Mark the buffer(s) to be logged by entering any character next to the required buffer(s). By default, a maximum of 80 bytes (from 0 to 79) is logged per buffer if no value is entered in the **From** and **To** fields.

#### ■ From

You can enter a start number (for example, 100) that determines from which byte the buffer is logged.

If you want to log an entire buffer (maximum is 32 KB), enter X or \* (asterisk) and leave the **To** field blank.

#### ■ To

You can enter an end number (for example, 200) that determines up to which byte the buffer is logged. The maximum input value is 32767 (32 KB).

If the length of the buffer(s) to be logged exceeds the total limit of 2097151 KB (2 GB - 1 byte), Natural issues an appropriate message as described in [Data Processing and Storage](#).

The logs of the buffers can be displayed on the **DBLOG Trace** screen as described in [Displaying Adabas Buffers](#).



**Note:** The snapshot function (see the relevant section) logs all Adabas buffers by default. Therefore, you need not mark any of the optional buffers before you execute this function.



# 8

## DBLOG Trace Screen

---

- DBLOG Trace Screen for Adabas Commands ..... 30
- DBLOG Trace Screen for DL/I Calls ..... 37
- DBLOG Trace Screen for SQL Statements ..... 39

The **DBLOG Trace** screen displays recorded log data on Adabas commands, or DL/I and SYNC/ROLB calls, or SQL statements which are kept in the Natural DBLOG buffer.

This section covers the following topics:

## DBLOG Trace Screen for Adabas Commands

---

- [Invoking DBLOG Trace for Adabas Commands](#)
- [Screen Columns and Commands on DBLOG Trace](#)
- [Displaying Adabas Buffers](#)
- [Displaying Adabas Commands that use Multi-Fetch](#)

### Invoking DBLOG Trace for Adabas Commands

The following is an example instruction for invoking the **DBLOG Trace** screen for Adabas commands.

1. Write the following Natural program:

```
DEFINE DATA LOCAL
1 EMP-VIEW VIEW OF EMPLOYEES
  2 NAME
END-DEFINE
READ (3) EMP-VIEW BY NAME
  DISPLAY NAME
END-READ
END
```

2. Enter the following Natural system command

```
TEST DBLOG
```

The message DBLOG started now is displayed.

3. Enter the following Natural system command:

```
RUN
```

The Natural program in the source area is executed.

4. Enter again:

```
TEST DBLOG
```



Column	PF Key	Explanation
	Direct Command	
	PF3	Exits the <b>DBLOG Trace</b> screen or closes a buffer window. The current log records are kept in the Natural DBLOG buffer.
	PF5	Moves log entries to the top of the screen: In column <b>M</b> , position the cursor next to the desired command and sequence number listed in column <b>No</b> and choose PF5. The logs are repositioned starting with the sequence number selected.
	PF6 or --	Scrolls to the beginning of a list or the data in a buffer window.
	PF7 or -	Scrolls up one page in a list or the data in a buffer window.
	PF8 or +	Scrolls down one page in a list or the data in a buffer window.
	PF9 or ++	Scrolls to the end of a list or the data in a buffer window.
	PF10	Only available in a buffer window with multiple record/format buffers. Displays the previous record/format buffer.
	PF11	Only available in a buffer window with multiple record/format buffers. Displays the next record/format buffer.
	PF12	Clears the Natural DBLOG buffer and deactivates logging.

### Displaying Adabas Buffers

The Adabas control block is recorded by default. If you want to record one or more Adabas buffers, you need to mark the buffer(s) required in the **DBLOG Menu** before executing the logging function as described in [Specifying Adabas Buffers](#). For example, if only logging of the format buffer has been marked in the **DBLOG Menu**, you can only display the **Format Buffer** window but not the **Record Buffer** window.

► To display control block or buffer information

- 1 In the input field next to the required command, enter the line command that corresponds to the required buffer and press ENTER:

Line Command	Requested Buffer
C	Control block
F	Format buffer
R	Record buffer
S	Search buffer
V	Value buffer
I	ISN buffer
.	A period (.) exits the <b>DBLOG Trace</b> screen. The current log records are kept in the Natural DBLOG buffer.

A window opens with the log data of the control block or buffer requested. If you entered several line commands, you can press PF3 to view the control block or buffer of the next command.

The following is an example of a window that contains data of a record buffer:

```

16:50:05          ***** NATURAL TEST UTILITIES *****                2008-08-01
User SAG          - DBLOG Trace -                                     Library SAG
M  No Cmd  DB   FNR  Rsp      ISN      ISQ  CID  CID(Hex)  OP  Pgm   Line
_   1  RC   10
_   2  S1 20000  50          1      2232 ? ?? 02000101  ADATEST 0200
R   3  L1 20000  50          1      2232  40404040  ADATEST 0200
+-----Page 1 of 1 (logged range:0x-0x4F)-----+ 200
! _____ Seq No .. 3      Record Buffer 1/13 (length:0x7A)      ! 340
! 0000 * C1C4D2C9 D5E2D6D5 40404040 40404040 * ADKINSON          * 0000 ! 350
! 0010 * 40404040 0000000B 00000001 40404040 *          ? ?      * 0010 ! 350
! 0020 * 40404040 00000000 00000000 00000000 *                  * 0020 ! 350
! 0030 * 00000000 00000000 00000000 00000000 *                  * 0030 !
! 0040 * 00000000 F0F0F0F0 F0F0F0F0 40404040 *          00000000 * 0040 !
+-----+
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help      Exit      --      -      +      ++      <      >      Canc
  
```

The fields provided in a buffer window are explained in the following table:

Field	Explanation
Page	The number of the current page and the total number of pages generated for the buffer (in the example above, 1 of 1).
logged range	The buffer length actually logged in hexadecimal format (in the example above, 0x0-0x4F).
Seq No	The sequence number of the command. In the example above, the command was executed in the third place (3).
<i>buffer-type</i>	<i>buffer-type</i> denotes the type of buffer requested.
<i>num-current</i> <i>/num-total</i>	In addition, for a format or record buffer, the number of record or format buffers is displayed:
<i>num-current</i>	Denotes the number of the record/format buffer currently shown.
<i>num-total</i>	Denotes the total number of the record/format buffers logged.  For a database call that uses the extended Adabas control block (ACBX), multiple format/record buffers are logged. The example above shows the first record from a total of 13 records (1 / 13).  For detailed information on ACBX, see <i>Adabas Control Block Structures (ACB and ACBX)</i> in the <i>Adabas for mainframes</i> documentation.
length	The total length of the record in hexadecimal format (in the example above, 0x7A).
—	In the input field next to <b>Seq No</b> , you can enter one of the following line commands:
C	Displays the control block.
F	Displays the format (F) or record (R) buffer.
or R	If pairs of format and record buffers exist, entering F in a record buffer or R in a format buffer will display the matching record buffer or format buffer respectively. For example, if the second record buffer is currently displayed, entering F will invoke a window with the corresponding second format buffer.
I	Displays the ISN buffer.
S	Displays the search buffer.

Field	Explanation	
	V	Displays the value buffer.
	<i>buffer-number</i>	You can enter the number of the record/format buffer you want to view.  See also Step 2 below.
	.	A period (.) closes the current buffer window.

- 2 In a record/format buffer window that contains multiple record/format buffers, you can use one of the following methods to view each record/format buffer:

Press PF10 to display the previous record/format buffer.

Or:

Press PF11 to display the next record/format buffer.

Or:

In the \_\_\_\_\_ input field, enter the number that corresponds to the record/format buffer you want to view.

### Displaying Adabas Commands that use Multi-Fetch

If the **MULTI-FETCH** clause is used in a **FIND**, **READ** or **HISTOGRAM** statement, only the Adabas commands that retrieve a set of records actually access the database. The records retrieved are moved into the multi-fetch buffer from where they are fetched during the execution of the database loop. The next database call is only made for the next set of records. For details, see *Multi-Fetch Clause* in the *Programming Guide*.

The **DBLOG Trace** screen lists both database calls and non-database calls: a database call is marked with an **M** in the first position of the **OP** column, whereas a non-database call for the multi-fetch buffer is marked with the less-than sign (<). This is demonstrated in the following example.

### Example of an Adabas Command with Multi-Fetch

Execute DBLOG for the following Natural program called MFETCH:

```

DEFINE DATA LOCAL
1 EMP-VIEW VIEW OF EMPLOYEES
  2 NAME
END-DEFINE
*
READ (5) MULTI-FETCH OF 3 EMP-VIEW BY NAME = 'ADKINSON'
  DISPLAY *COUNTER NAME
END-READ
END
    
```

A DBLOG Trace screen similar to the example below appears:

```

10:04:46          ***** NATURAL TEST UTILITIES *****          2008-07-24
User SAG          - DBLOG Trace -          Library SAG
M  No Cmd  DB   FNR  Rsp      ISN      ISQ  CID  CID(Hex)  OP  Pgm      Line
_   1  L3   10   316          2108      -?? 00600101 MA MFETCH  0060
_   2  L3   10   316          2109      -?? 00600101 <A MFETCH  0060
_   3  L3   10   316          2110      -?? 00600101 <A MFETCH  0060
_   4  L3   10   316          2111      -?? 00600101 MA MFETCH  0060
_   5  L3   10   316          2112      -?? 00600101 <A MFETCH  0060
_   6  RC   10   316          -?? 00600101 SI MFETCH  0060
_   7  RC   10          00000000 F  MFETCH  0090

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Print Exit      Posi  --  -  +  ++          Canc
    
```

The L3 commands listed as sequence numbers 1 and 4 retrieve a set of records from the database (indicated by M in the **OP** column) and return the first record back to the program. The remaining records are cached in the multi-fetch buffer.

The L3 commands listed as sequence numbers 2, 3 and 5 retrieve the record from the multi-fetch buffer (indicated by < in the **OP** column) and return it to the program.

### Contents of Record Buffer for Multi-Fetch Database Calls

The record buffer of a database call that uses multi-fetch contains the data of all records retrieved from the database. They are listed in the sequence in which they are processed.

When loading a set of records, Adabas overwrites the record buffer from the first byte to the extent of the records which are returned from the database. Any space left in the buffer is not cleared but still contains data of old records loaded during a previous database call. This means, for example, that if a field defined as NAME(A20) is read and a multi-fetch factor of 5 is used, the record buffer has a length of 100 (20 \* 5) bytes. If only 3 records are returned from the database, the record

buffer is only filled properly with the first 3 records (bytes 1 to 60), whereas the last 2 records (bytes 61 to -100) remain unchanged.

## DBLOG Trace Screen for DL/I Calls

- [Invoking DBLOG Trace for DL/I Calls](#)
- [Screen Columns on DBLOG Trace](#)

### Invoking DBLOG Trace for DL/I Calls

The following are example instructions for invoking the **DBLOG Trace** screen for DL/I calls.

1. Write the following Natural program:

```

DEFINE DATA LOCAL
01 COURSE VIEW OF DNDL01-COURSE
  02 COURSEN (A3)
  02 TITLE (A33)
01 OFFERING VIEW OF DNDL01-OFFERING
  02 COURSEN-COURSE (A3)
  02 LOCATION (A31)
END-DEFINE
READ (5) COURSE BY COURSEN
  IF TITLE = 'NATURAL'
    FIND (1) OFFERING WITH COURSEN-COURSE = COURSEN
      MOVE 'DARMSTADT' TO LOCATION
      UPDATE
    END OF TRANSACTION
  END-FIND
END-IF
END-READ
END

```

2. Enter the following Natural system command:

```
TEST DBLOG D
```

The message DBLOG started now is displayed.

3. Enter the following Natural system command:

```
RUN
```

The Natural program contained in the source area is executed.

4. Enter again:

```
TEST DBLOG D
```

Logging is deactivated and the **DBLOG Trace** screen for DL/I screen is displayed:

User SAG		- DBLOG Trace -							Library SAG	
No	Func	PCB	NS	SC	DBD/PSB	First SSA (truncated)	IOA (trunc)	Program	Line	
1	PCB				PCNQA42			LOGDL1	0090	
2	GU	1	1		DNDL01	COURSE *--(COURSEN =>	.	LOGDL1	0090	
3	GN	1	1		DNDL01	COURSE *--(COURSEN =>	.Z01	LOGDL1	0090	
4	GN	1	1		DNDL01	COURSE *--(COURSEN =>	.001	LOGDL1	0090	
5	GN	1	1		DNDL01	COURSE *--(COURSEN =>	.004NATURA	LOGDL1	0090	
6	GHNP	1	2		DNDL01	COURSE *- (COURSEN =004	?010791DAR	LOGDL1	0110	
7	REPL	1			DNDL01		?010791DAR	LOGDL1	0130	
8	SYNC							LOGDL1	0140	
9	PCB				PCNQA42			LOGDL1	0110	
10	GU	1	1		DNDL01	COURSE *--(COURSEN = 004	.004NATURA	LOGDL1	0110	
11	GHNP	1	2		DNDL01	COURSE *--(COURSEN = 004	?010791DAR	LOGDL1	0110	
12	GN	1	1		DNDL01	COURSE *--(COURSEN =>	+110	LOGDL1	0090	
***** End of Log *****										
NEXT								LIB=SAG		

### Screen Columns on DBLOG Trace

The columns of fields provided on the **DBLOG Trace** screen for DL/I calls are described in the following section.

Column	Explanation
<b>No</b>	Sequence number. The calls are displayed in the sequence in which they were executed.
<b>Func</b>	DL/I function.
<b>PCB</b>	PCB number.
<b>NS</b>	Number of SSAs.
<b>SC</b>	DL/I status code.
<b>DBD/PSB</b>	DBD name for DB calls. PSB name for scheduling calls.
<b>First SSA</b>	First 25 bytes of the first SSA.
<b>IOA</b>	First 13 bytes of the I/O area.
<b>Program</b>	Natural program name.
<b>Line</b>	Source-code line number.

## DBLOG Trace Screen for SQL Statements

- Invoking DBLOG Trace for SQL Statements
- Screen Columns and Commands on DBLOG Trace

### Invoking DBLOG Trace for SQL Statements

The following is an example of invoking the **DBLOG Trace** screen for SQL statements.

1. Write the following Natural program:

```

DEFINE DATA LOCAL
01 EMP VIEW OF DSN8810-EMP
  02 EMPNO
  02 FIRSTNME
  02 MIDINIT
  02 LASTNAME
  02 EDLEVEL
  02 SALARY
01 EMPPROJECT VIEW OF DSN8810-EMPPROJECT
  02 EMPNO
  02 PROJNO
  02 ACTNO
  02 EMPTIME
END-DEFINE
FIND (1) EMP WITH EMPNO > '000300'
  FIND (1) EMPPROJECT WITH EMPNO = EMPNO(0150)
  MOVE 0.75 TO EMPTIME
  UPDATE
  END-FIND
  ADD 1 TO EDLEVEL
  UPDATE
  END-FIND
*
FIND (1) EMP WITH EMPNO > '000300'
  FIND (1) EMPPROJECT WITH EMPNO = EMPNO(0240)
  DISPLAY EMPPROJECT EMP.EDLEVEL
  END-FIND
END-FIND
ROLLBACK
END

```

2. Enter the following Natural system command:

```
TEST DBLOG Q
```

The message DBLOG started now is displayed.

3. Enter the following Natural system command:

RUN

The Natural program in the source area is executed.

4. Enter again:

TEST DBLOG Q

Logging is deactivated and a **DBLOG Trace** screen for SQL statements similar to the example below appears:

```

11:28:58          ***** NATURAL Test Utilities *****                2008-07-28
User SAG          - DBLOG Trace -                                     Library SAG
M No   R SQL Statement (truncated)  CU SN SREF M Typ  SQLC/W Program  Line LV
-   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -
1     SELECT EMPNO,FIRSTNME,MIDINIT 01 01 0150 D DB2      LOGSQL   0150 01
2     FETCH CURSOR NEX              01 01 0150 D DB2      LOGSQL   0150 01
3     SELECT EMPNO,PROJNO,ACTNO,EMP 02 02 0160 D DB2      LOGSQL   0160 01
4     FETCH CURSOR NEX              02 02 0160 D DB2      LOGSQL   0160 01
5     UPDATE DSN8810.EMPPROJACT SET 02 03 0160 D DB2      LOGSQL   0180 01
6     CLOSE CURSOR                  02 02 0160 D DB2      LOGSQL   0160 01
7     UPDATE DSN8810.EMP SET EDLEVE 01 04 0150 D DB2      LOGSQL   0210 01
8     CLOSE CURSOR                  01 01 0150 D DB2      LOGSQL   0150 01
9     SELECT EMPNO,FIRSTNME,MIDINIT 05 05 0240 D DB2      LOGSQL   0240 01
10    FETCH CURSOR NEX              05 05 0240 D DB2      LOGSQL   0240 01
11    SELECT EMPNO,PROJNO,ACTNO,EMP 06 06 0250 D DB2      LOGSQL   0250 01
12    FETCH CURSOR NEX              06 06 0250 D DB2      LOGSQL   0250 01
13    CLOSE CURSOR                  06 06 0250 D DB2      LOGSQL   0250 01
14    CLOSE CURSOR                  05 05 0240 D DB2      LOGSQL   0240 01
15    ROLLBACK                      00 00 0000 D DB2      LOGSQL   0290 01
-
-
Command ==>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Print Exit Top  Posi Bot  -   +                               Canc
    
```

## Screen Columns and Commands on DBLOG Trace

The columns of fields and commands provided on the **DBLOG Trace** screen for SQL statements are described in the following section. You execute a command by either pressing a PF key or entering a direct command in the Command line.

Column	PF Key	Explanation
	Direct Command	
M		Input option for line commands:
	E	Executes the EXPLAIN command which provides information on the DB2 or SQL/DS optimizer's choice of strategy for executing SQL statements.  See also subsection <i>Using the EXPLAIN Command with Natural for DB2</i> and <i>Using the EXPLAIN Command with Natural for SQL/DS</i> both in the documentation on the LISTSQL command in the <i>System Commands</i> documentation.
	L	Executes the LISTSQL command which lists the Natural statements in the source code of an object and the corresponding SQL statements into which they have been translated. An SQL statement is identified by the library name, program name, and line number taken from the Natural DBLOG buffer.  See also <i>LISTSQL Command</i> in the <i>System Commands</i> documentation.
		<b>Important:</b> Since both commands obtain their information from the Natural system file, unwanted results may occur if the corresponding Natural program has been recataloged after the logging function was executed with the TEST DBLOG Q command. These unwanted results may be caused by statements modified after the logging.
No		Sequence number; the statements are displayed in the sequence in which they were executed.
R		Only applicable if the Natural file server for DB2 is in use.  Indicates by an asterisk in front of the corresponding statement that a reselection has been performed; if not, the column is left blank.  See also <i>Concept of the File Server</i> in the <i>Natural for DB2</i> documentation.

Column	PF Key	Explanation
	Direct Command	
SQL Statement		The first 29 characters of the logged SQL statement.
CU		Cursor number.
SN		Internal statement number.
SREF		Statement reference number.
M		Mode: D for dynamic or S for static.
Typ		Database type: DB2 or /DS.
SQLC/W		Either the SQL return code in the SQLCODE field of the SQLCA, or the warning in the SQLWARN0 field of the SQLCA if SQLCODE is 0.
Pgm		Natural program name.
Line		Source code line number.
LV		Program level.
	PF2	Prints a hardcopy of a screen shot.
	PF3	Exits the <b>DBLOG Trace</b> . The current log records are kept in the Natural DBLOG buffer.
	PF4	Scrolls to the beginning of the list.
	PF5	Moves log entries to the top of the screen: In column M, position the cursor next to the desired command and sequence number listed in column No and choose PF5. The logs are repositioned starting with the sequence number selected.
	PF6	Scrolls to the end of the list.
	PF7 or -	Scrolls up one page in the list.
	PF8 or +	Scrolls down one page in a list.
	PF12	Clears the Natural DBLOG buffer and deactivates logging.

# 9 DBLOG Snapshot Function

---

- Snapshot Function for Adabas Commands ..... 44
- Snapshot Function for DL/I Calls ..... 46
- Snapshot Function for SQL Statements ..... 48

The snapshot function provides detailed information on one particular Adabas command, DL/I call or SQL statement.

This section covers the following topics:

## Snapshot Function for Adabas Commands

---

This snapshot function interrupts program execution after executing the first Adabas command that matches the selection criteria specified in the **DBLOG Menu**. The **Snapshot Report** (see the following example screen) generated for the specified Adabas command is displayed immediately after program interruption.

The snapshot function automatically logs *all* Adabas buffers. Therefore, you do not have to mark any of the optional buffers in the **DBLOG Menu** before you start the snapshot function. The default **Snapshot Report** displays the control block (CB), which is either the classic control block (ACB) or the extended Adabas control block (ACBX).

This section covers the following topics:

- [Invoking Snapshot Report for Adabas Commands](#)
- [Displaying Buffers on Snapshot Report](#)

### Invoking Snapshot Report for Adabas Commands

#### ▶ To invoke the Snapshot Report screen for Adabas commands

- 1 In the **DBLOG Menu**, specify an Adabas command and additional criteria, if desired, and enter function code S.

The message `DBLOG snapshot facility started now` is displayed.

- 2 Execute a Natural program which contains the Adabas command specified in the **DBLOG Menu**.

The program stops executing and a **Snapshot Report** screen similar to the example below appears:

```

16:36:39          ***** NATURAL TEST UTILITIES *****          2006-12-12
                    - Snapshot Report -

Command Code : L3          Command ID   :   ??? 00200101 File Number  : 013C
Response Code:      0      ISN          :           1300
ISN Low Limit: 00000000   ISN Quantity:           0
FB Length   : 0009       RB Length   : 0014           SB Length   : 0008
VB Length   : 0014       IB Length   : 0000           Com. Option 1:
Com. Option 2: V         Additions 1  : AE]?           Additions 2  : ? ?
Additions 3  :           Additions 4  :
Global FID   : 0000000000000000 Command Time : 00000019 Pgm: SAGTEST Lin: 0020
Control Block
0000 * 30D5D3F3 00200101 013C0000 00000514 * ?NL3 ?????   ?? * 0000
0010 * 00000000 00000000 00090014 00080014 *           ? ? ? ? * 0010
0020 * 000000E5 C1C5BBCA 40404040 00120014 *   VAE]?   ? ? * 0020
0030 * 00000000 00000000 00000000 00000000 *           * 0030
0040 * 00000000 00000000 00000019 00000000 *           ?   * 0040
0050 * 00000000 00000000 00000000 00000000 *           * 0050
0060 * 00000000 00000000 00000000 00000000 *           * 0060
0070 * 00000000 00000000 00000000 00000000 *           * 0070

Command ==> CB
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help      Exit  CB   FB   RB   -   +   SB   VB   IB   Canc
  
```

### Displaying Buffers on Snapshot Report

The **Snapshot Report** screen shows the control block (CB) by default. If you want to display different Adabas buffers or scroll through a report, choose the appropriate PF key or, in the Command line, enter its equivalent direct command described in this section.

The availability of a PF key depends on the buffer currently displayed. If a buffer extends beyond the screen or contains multiple format/record buffers, PF keys required to scroll through the buffer are provided on the screen.

PF Key	Direct Command	Buffer
PF4	CB	Displays the control block. This is the default.
PF5	FB	Displays the format buffer.
PF6	RB  or --	Displays the record buffer (RB) or scrolls (- -) to the beginning of a long buffer.
PF7	-	Scrolls up one page in a long buffer.
PF8	+	Scrolls down one page in a long buffer

PF Key	Direct Command	Buffer
PF9	SB or ++	Displays the search buffer (SB) or scrolls (++) to the end of a long buffer.
PF10	VB or <	Displays the value buffer (VB). For multiple format/record buffers, shows the previous (<) record/format buffer.
PF11	IB or >	Displays the ISN buffer (IB). For multiple format/record buffers, shows the next (>) record/format buffer.

For information on the fields displayed in a control block or buffer, see [Displaying Adabas Buffers](#).

## Snapshot Function for DL/I Calls

---

This snapshot function generates the **Snapshot Report** (see the following example screen) of the first DL/I call that matches the selection criteria specified in the **DBLOG Menu**. A snapshot does not interrupt the program flow. The snapshot data is kept in the Natural DBLOG buffer to be displayed only if the user enters the appropriate DBLOG command as described below.

This section covers the following topics:

- [Invoking Snapshot Report for DL/I Calls](#)
- [Snapshot Report Information for DL/I Calls](#)

### Invoking Snapshot Report for DL/I Calls

#### ▶ To invoke the Snapshot Report screen for DL/I calls

- 1 In the **DBLOG Menu**, specify a DL/I call and additional criteria, if desired, and enter function code S.

The message `DBLOG snapshot facility started now is displayed`.

- 2 Execute a Natural program which contains the DL/I call specified in the **DBLOG Menu**. (Log data is written to the Natural DBLOG buffer.)
- 3 Display the snapshot data by entering the following command:

```
TEST DBLOG D
```

Or:



- Processing Options
  - Segment Name
  - Length of KFBA (Key Feedback Area)
  - Number of SENSEGs (Sensitive Segments)
  - KFBA:
    - Key Feedback Area
  - Number of SSAs (Segment Search Argument)
- 
- all SSAs
  - the I/O Area

The first 120 bytes of the Key Feedback Area, of all SSAs (up to 15 SSAs are possible) and of the I/O area are displayed, both in decimal and hexadecimal format.

The DBD Name in the PCB is used to read the corresponding NDB (Natural equivalent of DBD) from the Natural FDIC system file. In this NDB, the segment whose name is given in the PCB is located and its minimum/maximum length and segment level number are displayed. The segment level number should match the number in the PCB. In this way, it is possible to detect inconsistencies between Natural NDBs and DL/I DBDs.

The PSB name is used to read the corresponding NSB (Natural equivalent of PSB) from the Natural FDIC system file. From this NSB, the number of sensitive segments is displayed. This number should match the number in the PCB. In this way, it is possible to detect inconsistencies between Natural NSBs and DL/I PSBs.

The snapshot function checks whether the DL/I DBD/PSB and the Natural NDB/NSB contain the same values in the fields **Level Number** and **Number of SENSEGs**. The same values, however, do not necessarily ensure that the DL/I DBD/PSB and the Natural NDB/NSB are fully consistent.

In the [example above](#), the values in the **Number of SENSEGs** fields are different, because the Natural NATPSB procedure was not executed after the PSB had been changed by the DL/I PSBGEN procedure.

## Snapshot Function for SQL Statements

---

The snapshot function generates the [Snapshot Report](#) (see the following example screen) of the first SQL statement that matches the selection criteria specified in the **DBLOG Menu**. A snapshot does not interrupt the program flow.

Unlike the statements displayed with the DBLOG trace function, the snapshot shows the statement in its entirety (limited to 13 lines).

The snapshot data is kept in the Natural DBLOG buffer to be displayed only if the user enters the appropriate DBLOG command as described below.

This section covers the following topics:

- [Invoking Snapshot Report for SQL Statements](#)
- [Snapshot Report Information for SQL Statements](#)

## Invoking Snapshot Report for SQL Statements

### ▶ To invoke the Snapshot Report screen for SQL statements

- 1 In the **DBLOG Menu**, specify an SQL statement and additional criteria, if desired, and enter function code S.

The message DBLOG snapshot facility started now is displayed.

- 2 Execute a Natural program which contains the SQL statement specified in the **DBLOG Menu**. (Log data is written to the Natural DBLOG buffer.)
- 3 Display the snapshot data by entering the following command:

```
TEST DBLOG Q
```

Or:

In the **DBLOG Menu**, enter function code E.

A **Snapshot Report** screen for SQL statements similar to the example below appears:

```
10:59:28          ***** NATURAL Test Utilities *****          2006-12-12
User SAG                - Snapshot Report -                Library SAG

CU SN M Typ R SQLC/W      Library  Program  Store Clock Value   Line LV CID(Hex)
01 01 D DB2              SAG     SAGTEST  2002/04/03 14:23:06 0150 01 01500101

SQL Statement
SELECT EMPNO,FIRSTNME,MIDINIT,LASTNAME,EDLEVEL,SALARY FROM DSN8510.EMP WHERE EM
PNO > '000300' FOR UPDATE OF EDLEVEL

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Print Exit                                Canc
```

## Snapshot Report Information for SQL Statements

The following information is provided on the **Snapshot Report** screen for SQL statements:

Column	Explanation
CU	Cursor number.
SN	Internal statement number.
M	Mode: D for dynamic or S for static.
Typ	Database type: DB2 or SQL/DS.
R	<p>Only applicable if the Natural File Server for DB2 is in use.</p> <p>Indicates by an asterisk in front of the corresponding statement that a reselection has been performed; if not, the column is left blank.</p> <p>See also <i>Concept of the File Server</i> in the <i>Natural for DB2</i> documentation.</p>
SQLC/W	Either the SQL return code in the SQLCODE field of the SQLCA, or the warning in the SQLWARN0 field of the SQLCA if SQLCODE is 0.
Library	The library where the Natural program with the logged statement was cataloged.
Program	The name of the Natural program which contains the logged statement.
Store Clock Value	The time stamp of the Natural program which contains the logged statement.
Line	The source code line number of the logged statement.
LV	The call level of the Natural program which contains the logged statement.
CID (Hex)	The command ID of the logged statement in hexadecimal format.

# 10 TEST DBLOG Command

---

- Syntax Diagrams ..... 52
- Keyword Explanations ..... 53

The Natural system command `TEST DBLOG` is used to execute DBLOG and display or delete the log records currently stored in the Natural DBLOG buffer. Note that `TEST DBLOG` does not provide any parameters to specify selection criteria. Selection criteria can only be specified in the [DBLOG Menu](#).

The keywords that apply to the command `TEST DBLOG` are explained in the following syntax diagrams and tables. There are keywords that can be used to do both, activate and deactivate DBLOG (toggle effect). Activating and deactivating depends on whether or not there is data stored in the Natural DBLOG buffer as described in [Keyword Explanations](#).

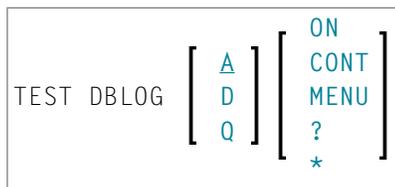
For explanations of the symbols used in the syntax diagrams, refer to *System Command Syntax* in the *System Commands* documentation.

This section covers the following topics:

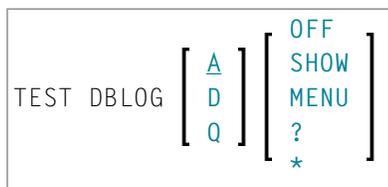
## Syntax Diagrams

---

### Activating DBLOG



### Deactivating DBLOG



## Keyword Explanations

Keyword	Function
A	<p>Default value.</p> <p>Toggle function:</p> <p>Activates logging of Adabas commands if no data exists in the Natural DBLOG buffer.</p> <p>Deactivates logging of Adabas commands and displays the <b>DBLOG Trace</b> screen of the current log record if data exists in the Natural DBLOG buffer.</p>
D	<p>Toggle function:</p> <p>Activates logging of DL/I calls if no data exists in the Natural DBLOG buffer.</p> <p>Deactivates logging of DL/I calls and displays the <b>DBLOG Trace</b> screen of the current log record if data exists in the Natural DBLOG buffer.</p>
Q	<p>Toggle function:</p> <p>Activates logging of SQL statements if no data exists in the Natural DBLOG buffer.</p> <p>Deactivates logging of SQL statements and displays the <b>DBLOG Trace</b> screen of the current log record if data exists in the Natural DBLOG buffer.</p>
CONT	Activates or reactivates (restarts) logging. A restart causes DBLOG to continue logging with the next program executed or run after DBLOG execution was stopped, and to add the new logs to the data that exists from previous recordings.
MENU	Invokes the <b>DBLOG Menu</b> which provides the options to activate or deactivate logging and to specify the commands, calls or statements to be logged; see the relevant section.
?	
*	
SHOW	Deactivates logging and displays the <b>DBLOG Trace</b> screen of the current log record if data exists in the Natural DBLOG buffer. Log record data is not deleted but kept in the Natural DBLOG buffer.
ON	Clears the Natural DBLOG buffer and activates logging.
OFF	Clears the Natural DBLOG buffer and deactivates logging.



# 11 INPL Utility

---

▪ Introducing the INPL Utility .....	56
▪ Load Libraries Only .....	61
▪ Load DDMs Only .....	61
▪ Load Error Messages Only .....	62
▪ Load All Objects .....	63
▪ Replace Product Installation .....	63
▪ Scan INPL File .....	64
▪ Natural Security Recover .....	65

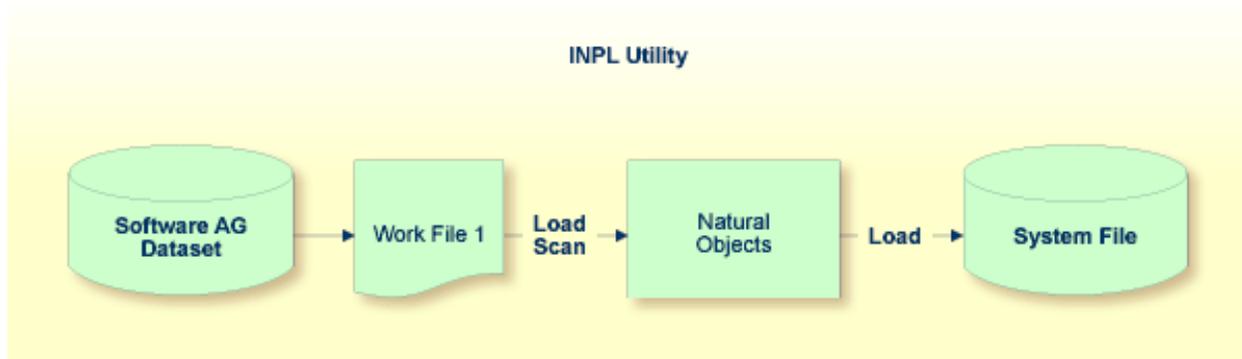
The INPL utility (Initial Natural Program Load) is used to load or scan Natural objects from datasets supplied by Software AG.

## Introducing the INPL Utility

---

The INPL utility processes Natural objects provided by Software AG.

The following diagram is a basic illustration of the INPL functionality:



The Natural objects are delivered as installation or update datasets which are assigned to Work File 1. The INPL utility loads the Natural objects from Work File 1 into Natural system files.

The Natural objects include cataloged objects and source objects that are contained in libraries in the Natural system files FNAT and FUSER, or in the system file FDIC where DDMS (data definition modules) are stored.

In addition to loading Natural objects, the INPL utility provides a scan function to check the contents of the dataset assigned to Work File 1 and a **Natural Security Recover** function which forces initialization of the Natural Security environment.

If an error occurs during INPL execution, the INPL will be interrupted and terminate abnormally with Condition Code 40.

This section covers the following topics:

- [Restrictions](#)
- [Special Cases](#)
- [Invoking INPL](#)
- [Options Available](#)
- [INPL Report](#)

- [Check Commands](#)

## Restrictions

You can process only datasets which are marked as “SAG system INPL dataset”.

## Special Cases

In all of the following cases, the INPL command must be specified using the dynamic Natural profile parameter `STACK`:

- when an INPL is to be performed on an empty FNAT system file;
- when a new system maintenance upgrade INPL is to be performed;
- when an existing product installation is to be replaced;
- when an INPL is to be performed in a Natural Security environment.

## Invoking INPL

### ▶ To invoke the INPL utility

- 1 Enter the following Natural system command:

```
INPL
```

An INPL menu similar to the example below is displayed:

```
18:32:19          ***** NATURAL INPL UTILITY *****          2005-04-21
User: SAG                                               Library: SYSTEM

                Code  Function
                L    Load Libraries Only
                D    Load DDMs Only
                E    Load Error Messages Only
                B    Load All Objects
                P    Replace Product Installation
                S    Scan INPL File
                R    Natural Security Recover
                ?    Help
                .    Exit

Code ..... B
Replace .... Y (Y/N/P/O)  Load Except . N (Y/N)
DDM Name ....
Library .....
```

Object Name .	Date .....	(YYYY-MM-DD)
Check Date .. N (Y/N)	Number .....	0

2 From the INPL menu, you can choose one of the following functions by entering the corresponding function code in the **Code** field:

- **Load Libraries Only**
- **Load DDMs Only**
- **Load Error Messages Only**
- **Load All Objects**
- **Replace Product Installation**
- **Scan INPL File**
- **Natural Security Recover**

For detailed information on these functions, refer to the corresponding sections.

3 Modify or complete the remaining input fields as described in *Options Available*.

4 Choose ENTER to confirm your entries.

**Options Available**

The following section describes the input fields on the INPL menu where you can specify one or more parameters as object selection criteria for the INPL function specified in the **Code** field. The use of a parameter depends on the respective function as indicated in the relevant documentation sections.

Field	Description
<b>Replace</b>	<p>Specifies whether the Natural objects to be processed are to replace any that already exist on the system files.</p> <p>Possible settings are:</p> <p>Y All existing objects are replaced. This is the default setting.</p> <p>N Existing objects are <i>not</i> replaced.</p> <p>P All existing objects are replaced. Additionally, existing Natural or Natural add-on product installations are replaced depending on the corresponding <b>check commands</b> (see below) executed during INPL processing.</p> <p>0 Resets the owner information of specified objects. Only applies to the function <b>Natural Security Recover</b>.</p> <p>See also <b>Check Date</b> to replace only objects that are older than the objects to be processed.</p> <p>If you use the function <b>Natural Security Recover</b>, you can enter option 0 in this field to reset the owner information of specified objects.</p>

Field	Description
<b>DDM Name</b>	<p>The name of a DDM or a range of names.</p> <p>If you enter a value that ends with an asterisk (*), each DDM with a name that starts with the specified value is processed. If only an asterisk (*) is entered or if this field is empty, all DDMs are processed.</p>
<b>Library</b>	<p>The name of a library or a range of names.</p> <p>If you enter a value that ends with an asterisk (*), each library with a name that starts with the specified value is processed. The library name is mandatory if <b>Object Name</b> is specified.</p>
<b>Object Name</b>	<p>The name of a Natural object (except DDMs) or a range of names.</p> <p>If the value ends with an asterisk (*), each object with a name that starts with the specified value is processed.</p> <p>If this field is empty, all objects contained in the library specified in the <b>Library</b> field are processed.</p>
<b>Check Date</b>	<p>Specifies whether existing Natural objects are to be replaced depending on their time stamp.</p> <p>This parameter has no effect if <b>Replace</b> is set to N.</p> <p>Possible settings are:</p> <p>Y Only objects which are older than the objects of the same name are replaced. An object is older if it was saved or cataloged before the object to be loaded.</p> <p>N All objects are replaced. This is the default setting.</p>
<b>Load Except</b>	<p>Specifies whether to exclude Natural objects from processing.</p> <p>This parameter does not apply to error messages.</p> <p>Possible settings are:</p> <p>Y All objects are processed except for the objects specified in the fields <b>DDM Name</b>, <b>Library</b> and/or <b>Object Name</b>.</p> <p>N No exceptions; all objects are processed. This is the default setting.</p> <p><b>Examples of load exceptions:</b></p> <p>All libraries except the library ABC are loaded:  <b>Code</b> = L  <b>Library</b> = ABC</p> <p>All DDMs with a prefix other than XY are loaded:  <b>Code</b> = D  <b>DDM Name</b> = XY*</p> <p>All objects contained in libraries with a prefix other than AB and all DDMs with a prefix other than CD are loaded:  <b>Code</b> = B</p>

Field	Description
	<b>Library</b> = AB* <b>DDM Name</b> = CD*
<b>Date</b>	Restricts processing to Natural objects which were saved or cataloged on or after the date entered in this field.  The date must be entered in the format <i>YYYY-MM-DD</i> ( <i>YYYY</i> = year, <i>MM</i> = month, <i>DD</i> = day).
<b>Number</b>	Limits processing of Natural objects to a specified number. All objects are counted which are loaded or scanned according to the selection criteria specified in the INPL menu.  If the number of objects processed has reached the value entered in the <b>Number</b> field, processing is terminated with a corresponding message.

## INPL Report

When the selected INPL function is complete, a corresponding INPL report is displayed on the screen.

## Check Commands

The INPL utility processes internal check commands which are executed automatically when performing an INPL. Check commands are used to control the load or the scan process and react on certain events. The check commands which are executed during an INPL are written into the INPL report.

The parameters used by the check command to react on an event are, for example, STOP, LOAD, CONTINUE, SKIP, USERLOAD or USERCONTINUE. You can display all parameters by using the **Scan INPL File** function. These parameters, for example, are used to verify that the version of a product to be installed is higher than the version of the product already installed.

You cannot modify a check command but you can influence the effect of the parameters USERLOAD and USERCONTINUE by selecting the function Replace Product Installation (function code P) or by setting the **Replace** option to P.

USERLOAD (or USERCONTINUE) means: Load (or continue loading) only if function code P has been selected or if the **Replace** option is set to P.

## Example of USERLOAD and USERCONTINUE

In the example of a check command below, the Natural version currently installed is checked before the INPL is performed:

```
CHECK VERSION NAT 421 LT USERLOAD EQ LOAD GT USERLOAD
```

If the Natural (NAT) version installed is below (LT) or above (GT) 421, the INPL function is only performed if function code P has been specified or if the **Replace** option has been set to P. Otherwise,

the INPL utility stops or terminates with Condition Code 40 in batch and writes a corresponding message into the INPL report.

If the version is equal to 421, the INPL function is always performed, irrespective of whether function code P has been specified or the **Replace** option has been set to P.

## Load Libraries Only

---

This function of the INPL utility is used to load Natural cataloged objects and source objects into specified libraries in the Natural system file FNAT or FUSER.

### ▶ To load libraries

- 1 In the INPL menu, enter function code L. You can specify parameters to be valid during execution of this function:
  - **Replace** (Y/N)
  - **Load Except** (Y/N)
  - **Library**
  - **Object Name**
  - **Date** (YYYY-MM-DD)
  - **Check Date** (Y/N)
  - **Number**

For detailed information on these parameters, refer to *Options Available* in the section *Introducing the INPL Utility*.

- 2 Confirm your entries.

When the function is complete, a corresponding **INPL report** (see the section *Introducing the INPL Utility*) is output.

## Load DDMs Only

---

This function of the INPL utility is used to load DDMs into the system file FDIC.

### ▶ To load DDMs

- 1 In the INPL menu, enter function code D. You can specify parameters to be valid during execution of this function:

- **Replace** (Y/N)
- **Load Except** (Y/N)
- **DDM Name**
- **Number**

For detailed information on these parameters, refer to *Options Available* in the section *Introducing the INPL Utility*.

- 2 Confirm your entries.

When the function is complete, a corresponding **INPL report** (see the section *Introducing the INPL Utility*) is output.

## Load Error Messages Only

---

This function of the INPL utility is used to load user-defined error messages or system error messages into specified libraries in the Natural system file FUSER or FNAT respectively.

### ▶ To load error messages

- 1 In the INPL menu, enter function code E. You can specify parameters to be valid during execution of this function:

- **Replace** (Y/N)
- **Library**

For detailed information on these parameters, refer to *Options Available* in the section *Introducing the INPL Utility*.

- 2 Confirm your entries.

When the function is complete, a corresponding **INPL report** (see the section *Introducing the INPL Utility*) is output.

---

## Load All Objects

---

This function of the INPL utility is used to load all Natural objects (including error messages and DDMs) into the libraries indicated in Work File 1. DDMs are loaded into the system file FDIC.

▶ **To load all objects**

- 1 In the INPL menu, enter function code B. You can specify parameters to be valid during execution of this function:
  - **Replace** (Y/N)
  - **Load Except** (Y/N)
  - **DDM Name**
  - **Library**
  - **Object Name**
  - **Date** (YYYY-MM-DD)
  - **Check Date** (Y/N)
  - **Number**

For detailed information on these parameters, refer to *Options Available* in the section *Introducing the INPL Utility*.

- 2 Confirm your entries.

When the function is complete, a corresponding **INPL report** (see the section *Introducing the INPL Utility*) is output.

---

## Replace Product Installation

---

In addition to the function **Load All Object**, this function replaces any existing Natural or Natural add-on products installed in the current system environment. The replacement of existing product installations depends on the execution of the corresponding **check commands** described in the section *Introducing the INPL Utility*.

▶ **To load all objects and replace existing product installations**

- 1 In the INPL menu, enter function code P. You can specify parameters to be valid during execution of this function:
  - **Replace** (Y/N)

- **Load Except** (Y/N)
- **DDM Name**
- **Library**
- **Object Name**
- **Date** (YYYY-MM-DD)
- **Check Date** (Y/N)
- **Number**

For detailed information on these parameters, refer to *Options Available* in the section *Introducing the INPL Utility*.

- 2 Confirm your entries.

When the function is complete, a corresponding **INPL report** (see the section *Introducing the INPL Utility*) is output.

## Scan INPL File

---

This function of the INPL utility is used to scan the contents of the dataset assigned to Work File 1.

### ▶ To scan an INPL File

- 1 In the INPL menu, enter function code S. You can specify parameters to be valid during execution of this function:
  - **Load Except** (Y/N)
  - **DDM Name**
  - **Library**
  - **Object Name**
  - **Date** (YYYY-MM-DD)
  - **Number**

For detailed information on these parameters, refer to *Options Available* in the section *Introducing the INPL Utility*.

- 2 Confirm your entries.

When the function is complete, a corresponding **INPL report** (see the section *Introducing the INPL Utility*) is output.

---

## Natural Security Recover

---

This function of the INPL utility is used to force initialization of the Natural Security environment.

The following options are provided:

- [Reset Environment](#)
- [Remove Owners](#)

### Reset Environment



**Caution:** Execution of this function will reset the user profile DBA and the library profile SYSSEC as well as the link between these two objects as they were after the initial installation; all other links to the library SYSSEC will be canceled. Other Natural Security profiles and links will not be modified. Contact Software AG technical support for further information.

#### ▶ To reset the environment

- In the INPL menu, enter function code R.

### Remove Owners

#### ▶ To remove owners

- In the INPL menu, enter function code R and enter an 0 in the **Replace** field to reset the owner information of specified objects.



# 12 NATPAGE Utility - Screen Capturing

---

The utility NATPAGE is used to capture screen output data (maps and reports) during a Natural session. The term screen in this context means the contents of the page buffer; that is, the logical page output by Natural.

The screen captures are stored in the Natural scratch-pad file as described in the relevant section in the *Operations* documentation.

The maximum number of screens that can be captured is determined by the session parameter PD, which is described in the *Parameter Reference* documentation.

The NATPAGE utility consists of the following Natural terminal commands:

Command	Function
%P	Activates NATPAGE and captures the contents of the current screen and all subsequent screens. Screens captured previously are deleted.
%I	Activates NATPAGE and captures the contents of the current screen.
%O	Deactivates NATPAGE.
%S	Resumes NATPAGE.
%E	Displays the screens captured with NATPAGE.

See the *Terminal Commands* documentation for a detailed description of these terminal commands.



# 13 NATRJE Utility - Natural Remote Job Entry

---

- General Information on NATRJE ..... 70
- Calling NATRJE from a Natural Program ..... 71
- NATRJE Return Codes ..... 75
- NATRJE Features Applicable to openUTM/TIAM ..... 77

The NATRJE utility (Natural Remote Job Entry) can be used to submit JCL cards from a Natural program to the operating system for scheduling and execution. For example, it is possible to start a Natural batch job with NATRJE.

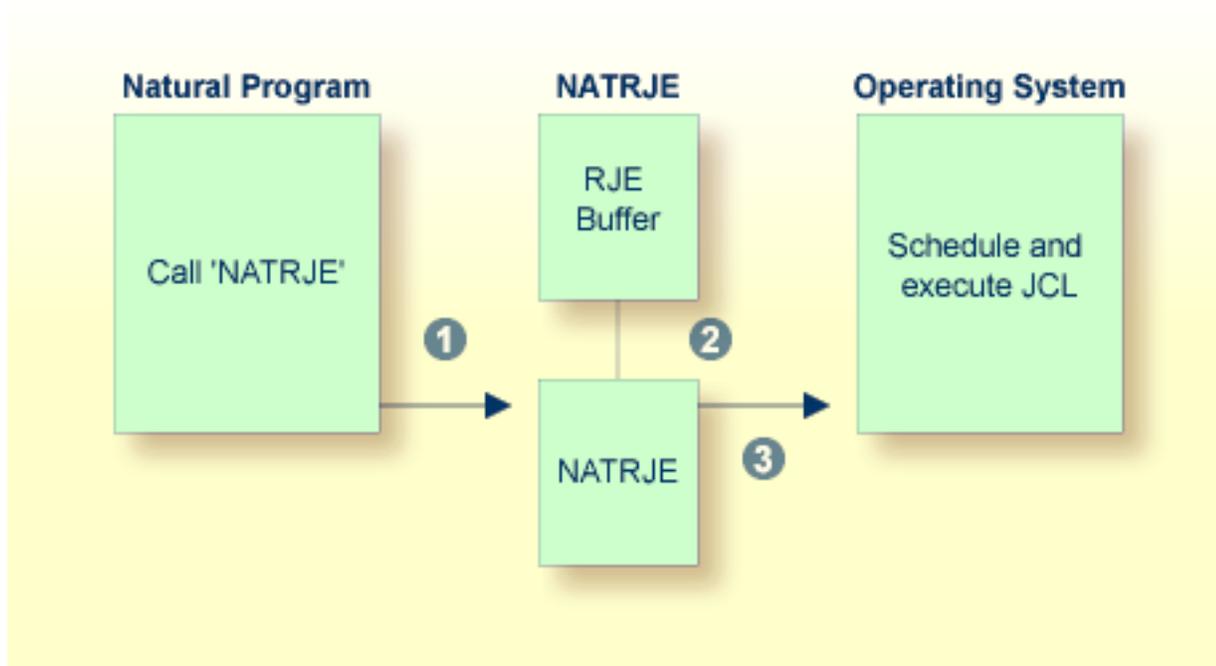
**Related Topic:**

- *NREXPG - User Exit for NATRJE* in the *Operations* documentation

## General Information on NATRJE

---

The following graphic is a simplified functional diagram of the NATRJE utility:



**Legend**

- 1 The Natural program calls the NATRJE utility for the purpose of submitting JCL cards to be executed by the operating system.
- 2 NATRJE collects the JCL cards into the RJE buffer until the Natural program indicates that the job is complete. The RJE buffer holds the JCL cards before they are submitted. The initial size of the RJE buffer is determined by the `RJESIZE` profile parameter (as described in the *Parameter Reference* documentation). If a given job does not fit into the RJE buffer, the buffer is automatically enlarged. The maximum size of a job is determined by the thread or region size.

- 3 NATRJE transfers the JCL cards to the operating system internal job queue for scheduling and execution by the operating system.

Note for BS2000/OSD:

When the job generation is complete, NATRJE transfers the JCL cards to a BS2000/OSD dataset, which is generated by NATRJE. The dataset is a SAM file that contains the generated JCL cards and is submitted by using the ENTER command.

## Calling NATRJE from a Natural Program

This section covers the following topics:

- [Invoking NATRJE](#)
- [Example Programs](#)

### Invoking NATRJE

#### ▶ To invoke the NATRJE utility

- In the Natural program that invokes the NATRJE utility, specify a CALL statement by using the following syntax:

```
CALL 'NATRJE' parm1 parm2 parm3 parm4
```

The parameters specified in the CALL statement are explained in the following table:

Parameter	Explanation	
<i>parm1</i>	The starting JCL card of the table which contains one or more 80-character JCL cards to be submitted.	
<i>parm2</i>	A 4-byte binary field which contains the number of 80-character JCL cards to be submitted.	
<i>parm3</i>	A 1-byte alphanumeric field used to indicate if all JCL cards have been submitted:	
	' '	Not the last call for the current job. A further JCL card follows with the next CALL statement. The JCL cards are collected into the RJE buffer.
	B	BS2000/OSD and z/OS only: Last call for the current job.  Under BS2000/OSD: The job is generated, written to the dataset, but not started automatically.

Parameter	Explanation
	Under z/OS (batch and TSO, IMS TM and CICS): The job is written to the internal reader dataset but not submitted. If function L (see below) is called subsequently, the internal reader is closed and the job(s) are submitted. In addition, the internal reader is closed and the job is submitted either on a screen I/O (IMS TM) or during session termination (z/OS batch, TSO and IMS TM).
C	Flush the current job. The job is not submitted to the system. Under BS2000/OSD, no dataset is created.
L	Last call for the current job. The job is submitted to the system.
	BS2000/OSD environments: see <a href="#">Additional Values for parm3</a> .
parm4	A 2-byte binary field in which NATRJE returns a response code.

## Example Programs

This section provides Natural example programs that submit JCL cards:

- [Example Program for z/OS](#)
- [Example Program for z/VSE](#)
- [Example Program 1 for BS2000/OSD](#)
- [Example Program 2 for BS2000/OSD](#)

### Example Program for z/OS

The following is a Natural example program that submits, in one call to NATRJE, a three-card JCL stream.

```

DEFINE DATA LOCAL
  01 COUNT  (B4)
  01 FLAG   (A1)
  01 RETHEX (B2)
  01 CARDS  (A240)
  01 REDEFINE CARDS
    02 CARD1 (A80)
    02 CARD2 (A80)
    02 CARD3 (A80)
END-DEFINE
MOVE '//JOBN JOB CLASS=G,MSGCLASS=X' TO CARD1
MOVE '//XXX EXEC PGM=IEFBR14' TO CARD2
MOVE '//DD1 DD DSN=NATRJE.SOURCE,DISP=SHR' TO CARD3
MOVE 3 TO COUNT
    
```

```

MOVE 'L' TO FLAG
CALL 'NATRJE' CARDS COUNT FLAG RETHEX
IF RETHEX = H'0000'
  WRITE 'JOB submitted successfully'
ELSE
  WRITE 'ERROR from NATRJE' RETHEX
END-IF
END

```

### Example Program for z/VSE

The following is a Natural example program that submits, in three calls to NATRJE, a seven-card JCL stream.

```

DEFINE DATA LOCAL
  01 COUNT (B4)
  01 FLAG (A1)
  01 RETHEX (B2)
  01 CARDS (A240)
  01 REDEFINE CARDS
    02 CARD1 (A80)
    02 CARD2 (A80)
    02 CARD3 (A80)
END-DEFINE
MOVE '* $$ JOB JNM=DSERV,CLASS=0,DISP=D' TO CARD1
MOVE '* $$ LST CLASS=A,DISP=D' TO CARD2
MOVE '// JOB DSERV TO DSERV SOURCE MEMBERS' TO CARD3
MOVE 3 TO COUNT
CALL 'NATRJE' CARDS COUNT FLAG RETHEX
PERFORM RETCODE-CHECK
MOVE '// EXEC PROC=NATSPLP' TO CARD1
MOVE '// EXEC DSERV' TO CARD2
MOVE ' DSPLYS SD' TO CARD3
MOVE 3 TO COUNT
CALL 'NATRJE' CARDS COUNT FLAG RETHEX
PERFORM RETCODE-CHECK
MOVE '/*' TO CARD1
MOVE '/&' TO CARD2
MOVE '* $$ EOJ' TO CARD3
MOVE 3 TO COUNT
MOVE 'L' TO FLAG
CALL 'NATRJE' CARDS COUNT FLAG RETHEX
DEFINE SUBROUTINE RETCODE-CHECK
IF RETHEX NE H'0000'
  WRITE 'ERROR from NATRJE:' RETHEX
STOP
END-IF
END-SUBROUTINE
END

```

**Example Program 1 for BS2000/OSD**

The following is a Natural example program that submits, in three calls to NATRJE, a nine-card JCL stream.

```
DEFINE DATA LOCAL
  01 COUNT  (B4)
  01 FLAG   (A1)
  01 RETHEX (B2)
  01 CARDS  (A240)
  01 REDEFINE CARDS
    02 CARD1 (A80)
    02 CARD2 (A80)
    02 CARD3 (A80)
END-DEFINE
MOVE '/LOGON' TO CARD1
MOVE '/SYSFILE SYSDTA=(SYSCMD)' TO CARD2
MOVE '/SYSFILE SYSIPT =IPT.PARM' TO CARD 3
MOVE 3 TO COUNT
CALL 'NATRJE' CARDS COUNT FLAG RETHEX
  IF RETHEX NE H'0000' DO
    WRITE RETHEX (EM=HH)
  END-IF
MOVE '/SETSW ON=2' TO CARD1
MOVE '/EXEC NATBATCH' TO CARD2
MOVE 'LOGON APPLIC' TO CARD3
MOVE 3 TO COUNT
CALL 'NATRJE' CARDS COUNT FLAG RETHEX
  IF RETHEX NE H'000' DO
    ...
    ...
  END-IF
MOVE 'RUNPGM' TO CARD1
MOVE 'FIN' TO CARD2
MOVE '/LOGOFF' TO CARD3
MOVE 3 TO COUNT
MOVE 'L' TO FLAG
CALL 'NATRJE' CARDS COUNT FLAG RETHEX
  ...
  ...
  ...
END
```

## Example Program 2 for BS2000/OSD

The following is a Natural example program that submits, in one call to NATRJE, a nine-card JCL stream.

```
DEFINE DATA LOCAL
  01 COUNT  (B4)
  01 FLAG   (A1)
  01 RETHEX (B2)
  01 CARD1  (A80)
  01 CARD2  (A80)
  01 CARD3  (A80)
  01 CARD4  (A80)
  ...
  01 CARD9  (A80)
END-DEFINE
MOVE '/LOGON' TO CARD1
MOVE '/SYSFILE SYSDTA=(SYSCMD)' TO CARD2
...
MOVE '/LOGOFF' TO CARD9
MOVE 9 TO COUNT
MOVE 'L' TO FLAG
  CALL 'NATRJE' CARD1 COUNT FLAG RETHEX
...
END
```

## NATRJE Return Codes

A Natural program that issues a `CALL` statement to NATRJE results in one of the following return codes being returned in the fourth parameter of the statement. There are return codes that apply to all environments and additional codes that are dependent on the operating system:

- [Return Codes Common to all Environments](#)
- [Additional Return Codes for CICS and Batch under z/VSE](#)
- [Additional Return Codes for CICS under z/OS](#)

▪ [Additional Return Codes under BS2000/OSD](#)

**Return Codes Common to all Environments**

Return Code/Hexadecimal	Return Code/Decimal	Explanation
00	00	Normal return.
04	04	NATRJE utility not available.
08	08	NATRJE utility disabled; a possible reason is that the RJE SIZE profile parameter is set to 0 (see also the <i>Parameter Reference</i> documentation).
0C	12	Invalid number of JCL cards.
10	16	Invalid function code.
14	20	No RJE buffer space available.
18	24	Invalid number of parameters.
1C	28	I/O error during submit.
20	32	Job flushed by user exit NREXPG (see <i>NREXPG - User Exit for NATRJE</i> in the <i>Operations</i> documentation).

**Additional Return Codes for CICS and Batch under z/VSE**

Return Code	Explanation
<i>ffrr</i>	<i>ff</i> is the XPCC request function code and <i>rr</i> the associated return code.

**Additional Return Codes for CICS under z/OS**

Return Code	Explanation
01 <i>nn</i>	CICS WRITEQ TD failure; <i>nn</i> is the CICS response code in hexadecimal format.
01 <i>nn</i>	CICS CLOSE TD failure; <i>nn</i> is the CICS response code in hexadecimal format.

**Additional Return Codes under BS2000/OSD**

Return Code	Explanation
9001	No RJE buffer found.
9002	No buffer space available.
9003	Missing LOGON command.
9004	Only LOGON cards generated.
9005	Too many LOGON parameters.
D010	Error in ENTER macro.

Return Code	Explanation
Dxxx	Operating system error: The error message is sent directly to the user program; the BS2000/OSD HELP command provides additional information.

## NATRJE Features Applicable to openUTM/TIAM

This section covers the following topics:

- SDF Command SET-LOGON-PARAMETER
- Additional Values for parm3
- Name of BS2000/OSD Dataset

### SDF Command SET-LOGON-PARAMETER

You can replace the ISP command LOGON by the SDF command SET-LOGON-PARAMETER in the first JCL card of the job to be executed. However, the following restrictions apply when using the SDF command:

- For the SET-LOGON-PARAMETER command, you must use the following command abbreviation:

```
STLGP
```

- If you want to specify a user ID, account number and/or password with the STLGP command, you must use the syntax of the ISP LOGON command. This means that you are not allowed to use the SDF keyword operands USER-IDENTIFICATION, ACCOUNT and PASSWORD. All other keyword operands are allowed in the SDF command.

### Examples of ISP and SDF Commands:

ISP command

```
/ .job-id LOGON user-id,account-number,'password'
```

and corresponding SDF command:

```
/ .job-id STGLP user-id,account-number,'password'
```

SDF command with additional keyword operands:

```
/ .job-id STGLP user-id,account-number,'password', -  
/RESOURCES=*PARAMETERS(RUN-PRIORITY=220)
```

### Additional Values for parm3

Value	Explanation
A	Combination of values T and E (see below).
E	The job is generated and completed. Before submission to the BS2000/OSD operating system, the parameter ERASE=YES is added to the ENTER parameter.
T	The job is generated and completed. Before submission to the BS2000/OSD operating system, a time limit is calculated using the Natural MT parameter (see also the <i>Parameter Reference</i> documentation). If MT is set to 0, the time limit is generated as NTL. The calculated time limit is added to the ENTER parameter via the TIME= <i>operand</i> .

When using the values T, E or A, NATRJE does not check whether the parameters TIME= or ERASE= exist in the user-created LOGON cards.

### Name of BS2000/OSD Dataset

The name of the BS2000/OSD dataset created by NATRJE for the JCL cards is as follows:

```
E.DDMMYY.HHMMSSSS.program-name.user-id
```

Parameter	Explanation
DD	The day of the dataset creation.
MM	The month of the dataset creation.
YY	The year of the dataset creation.
HH	The hour of the dataset creation.
MM	The minute of the dataset creation.
SSSS	The seconds and milliseconds of the dataset creation.
<i>program-name</i>	The name of the Natural program that creates the dataset.
<i>user-id</i>	The corresponding Natural user ID.

# 14 Object Handler

---

The Object Handler is designed to process Natural and non-Natural objects for distribution in Natural environments. This is done by unloading the objects in the source environment into work files and loading them from work files into the target environment.

- **General Information on the Object Handler** Invoking the Object Handler in batch or online mode; applying Natural Security.
- **Functions** Using the Object Handler menu functions: unload, load, restart load, scan, view, find and administration.
- **Object Specification** Specifying the objects to be processed with Object Handler menu functions.
- **Settings** Specifying option and parameter settings for Object Handler menu functions.
- **Workplans** Using standard procedures to execute Object Handler functions.
- **Name, Date and Time Specification** Specifying names, dates, times and ranges.
- **Work Files** Work files used by the Object Handler.
- **Direct Commands** Using direct commands to perform Object Handler functions.
- **Batch Condition Codes and User Exit Routines** Condition codes and user exit routines provided in batch mode.
- **Tools** Displaying status information and setting trace and report options.
- **Profile Settings** Setting up a profile to define individual defaults and standard procedures.
- **Migration from NATUNLD/NATLOAD and SYSTRANS to the Object Handler** Migrating from the utilities NATUNLD/NATLOAD and SYSTRANS to the Object Handler.



# 15

## General Information on the Object Handler

---

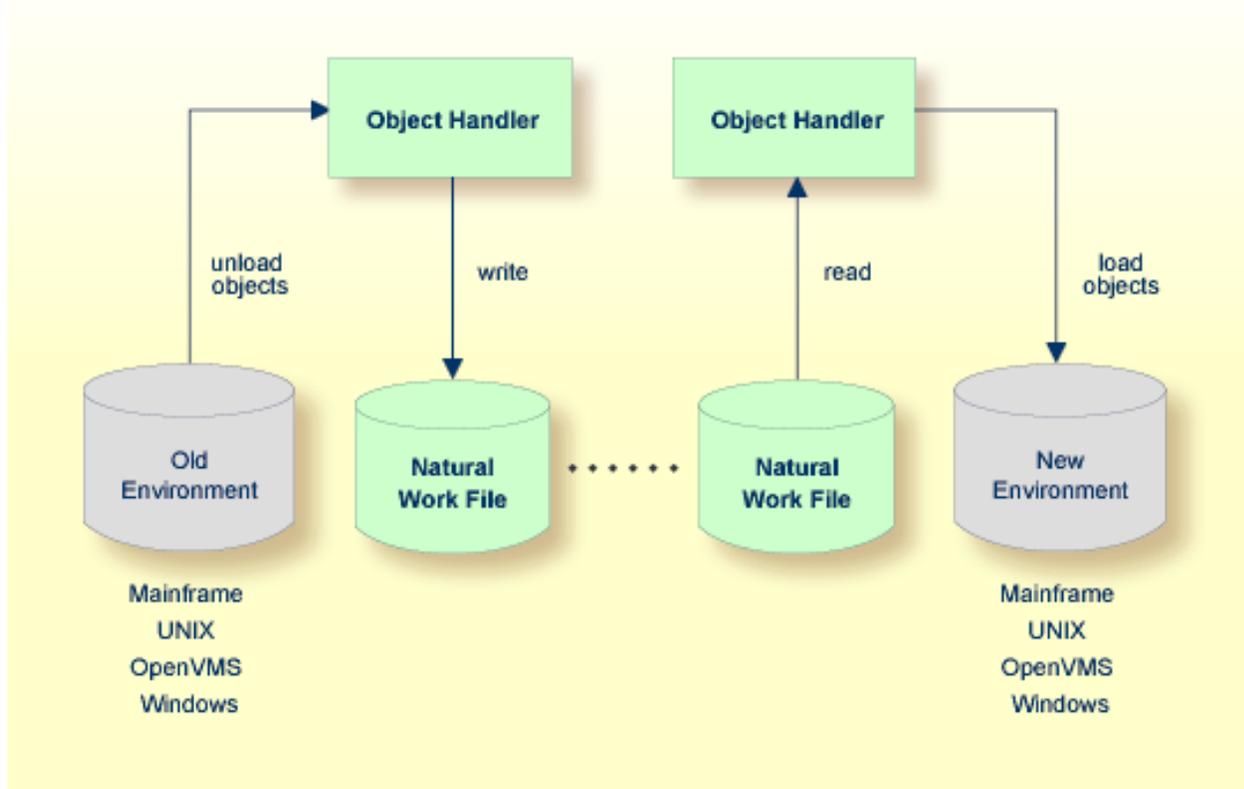
▪ Principles of Object Transfer .....	82
▪ Invoking the Object Handler .....	83
▪ Batch or Direct Command Calls .....	84
▪ Issuing Object Handler Commands from a Natural Program .....	85
▪ Text Members for Reports, Restarts and Traces .....	86
▪ Natural Security .....	86
▪ Standard PF Keys .....	87

The Object Handler consists of the utility SYSOBJH which is located in the Natural system library SYSOBJH, and the direct command interface. Additionally, the Application Programming Interface OBJHAPI is provided for executing Object Handler functions from a Natural program.

## Principles of Object Transfer

---

The diagram below illustrates how the Object Handler transfers objects by unloading them from the source environment into work files and loading them from work files into the target environment. If required, an application protocol such as FTP can be used for transferring work files from source to target environments.



This section covers the following topics:

- Transfer Environment and File Security
- Objects Processed by the Object Handler

- [Formatting Options](#)

## Transfer Environment and File Security

An old or a new environment is an FNAT, FUSER or FDIC system file contained in an Adabas database or a VSAM file system on a mainframe, or in the file system on a UNIX, an OpenVMS or a Windows platform. Natural objects on the FNAT or FUSER system file can be contained in libraries as indicated in the following section.

The file security (that is, passwords and cipher codes) relates to the security that has been defined for a system file in an Adabas or a VSAM environment. If file security has been defined for a system file, you need to specify a password, cipher code and/or VSAM name for the source and/or target system file required before you perform an Object Handler function. Otherwise, Adabas or VSAM will issue an appropriate error message. You do not have to provide security information for the default system files assigned to the Natural session at the start of the Object Handler.

## Objects Processed by the Object Handler

The Object Handler transfers Natural source objects (also referred to as saved objects) and cataloged objects which are contained in Natural libraries, Natural error messages, Natural command processor sources, Natural DDMs (data definition modules), Natural-related objects, and Adabas FDTs (Field Definition Tables).

## Formatting Options

You can transfer data of binary or text format, depending on the source and target environment where the objects are processed.

Binary format can be used for source objects and cataloged objects (including DDMs), error messages, Natural command processor sources, Natural-related objects and Adabas FDTs.

Text format applies to source objects, Natural command processor sources, error messages, DDMs and Adabas FDTs. You can only transfer text data between mainframe and UNIX/OpenVMS/Windows platforms. You can transfer binary data between identical platforms.

## Invoking the Object Handler

---

This section provides instructions for invoking the Object Handler by using either menu functions or direct commands:

- ▶ **To invoke the Object Handler online from any Natural library**

- 1 Enter the following system command:

## SYSOBJH

The **Main Menu** of the Object Handler appears with the following options:

- Unload
- Load
- Scan
- View
- Administration

See the section *Functions* for descriptions of these functions, and how to process the functions in advanced-user mode or by using wizards.

- 2 Select a function by choosing one of the following methods:

Enter any character in the input field next to the item that corresponds to the function required.

Or:

Choose the PF key that corresponds to the function required.

Or:

In the Command line, enter the Object Handler command that corresponds to the function required. For information on the commands provided, see the section *Direct Commands*.

### ► To invoke the Object Handler in batch or direct command online mode

- Enter the system command `SYSOBJH` followed by a direct command as described in *Batch or Direct Command Calls* and in *Direct Commands*.

After execution of a direct command, you can enter either another direct command or a period (.) to exit the Object Handler.

## Batch or Direct Command Calls

---

Several commands can be issued to the Object Handler online or in batch mode. The last command in the command sequence must be a period (.), `STOP`, `END`, `QUIT` or `FIN`, where `FIN` ends the Natural session.

The section covers the following topics:

- [Batch Mode](#)

- [Online Mode](#)

## Batch Mode

The commands to the Object Handler are read from standard input. Each command can be separated into a maximum of 20 command parts/strings by entering input delimiters (session parameter `ID`) after any keyword or keyword value. Each command part/string must not exceed 248 bytes.

If the command is longer than a single line, at the end of every line except the last that belongs to the command, enter the character defined with the session parameter `CF` (default is `%`) This indicates continuation on the next line. However, this is only possible if you specify the command `SYSOBJH` in a line by itself. That is, you cannot use `CF`, if you enter `SYSOBJH` in the same line where a multi-line command starts. In addition, we recommend that you set the `LS` profile parameter to 250.

**Example (assuming `ID` is set to `.`):**

```
SYSOBJH
UNLOAD PROG* LIBRARY MYLIB1, OBJTYPE N,%
WITH NEWLIBRARY MYLIB2%
WHERE REPORT TRANSFER
STOP
```

### Related Topics:

- [Direct Commands](#)
- [Natural in Batch Mode - Operations](#) documentation

## Online Mode

The command to the Object Handler in the Command line can consist of up to 20 command parts.

**Example:**

```
SYSOBJH UNLOAD * LIB EXAMPLE WHERE TRANSFER
```

## Issuing Object Handler Commands from a Natural Program

You can issue commands to the Object Handler with a Natural program by using the `OBJHAPI` Application Programming Interface, which is supplied as a subprogram in the Natural system library `SYSOBJH`. For the parameters required and examples, see the Natural program `DOC-API` supplied in the library `SYSOBJH`.

## Text Members for Reports, Restarts and Traces

---

Report, restart and trace data created by the Object Handler are stored as Natural text members (Natural objects of the type Text) in the Workplan library. The Object Handler generates names for text members that have not been explicitly specified in the **Options** window. The names generated are a combination of the weekday and the time. For example: a member with the name 21415568 was created on Tuesday (the second day of the week) at 14:15:56,8.

You can specify the Workplan library in which the text members are stored by using the `Workplan-Library` profile option of the Object Handler described in [Profile Settings](#). If your Natural session uses a read-only FNAT or FUSER system file as specified with the `ROSY=ON` profile parameter (see the *Parameter Reference* documentation), the text members are stored in a scratch-pad file (see the *Operations* documentation) under one of the following conditions:

- The database ID and file number specified for the Workplan library is identical to the current FNAT or FUSER system file.
- No database ID and file number are specified for the Workplan library.

## Natural Security

---

The use of the Object Handler under Natural Security requires that utility profiles be defined for it in Natural Security. At least, a default profile must be defined. For information on utility profiles, see the section *Protecting Utilities* in the *Natural Security* documentation.

If Natural Security is installed, the Object Handler checks the SYSOBJH utility profiles in Natural Security to find out whether the requested function is allowed.

Should a Natural Security error occur during the load function, the following applies:

- If the **Write report** option is set, in online mode, the error message is written to the report file and processing continues for the current load command.
- If the **Write report** option is set, in batch mode, the error message is written to the report file and the Object Handler terminates after the load command where the error occurred has finished processing.
- If the **Write report** option is not set, an error message is issued and the load command is terminated.

## Standard PF Keys

The following PF keys are available on all full-screen maps:

PF Key	Explanation
PF1	Invokes the help function for the field at which the cursor is positioned.
PF3	Exits the current screen and returns to the previous screen.
PF6	Goes to the top of a list.
PF7	Scrolls up one page in a list.  On wizard screens: goes back one screen/step.
PF8	Scrolls down one page in a list.  On wizard screens: goes to the next screen/step.
PF9	Goes to the bottom of a list.
PF10	Invokes the <b>Commands</b> menu to select commands for navigation purpose and to assign special settings. See also <i>Commands for Navigation and Special Functions</i> in <i>Direct Commands</i> .
PF12	Cancels the current function.
PF20	Lists all active programs of the Object Handler. This can be helpful information for reporting technical problems to Software AG.



# 16 Functions

---

This section describes the main functions provided by the Object Handler.

You can take advantage of the Object Handler wizards to guide you through the steps required to execute the unload, load and scan functions. The wizards are activated by default. If you prefer the unload, load or scan mode for the experienced user instead, in the **Main Menu**, mark the field next to **Advanced user**. You can also set advanced-user mode as the default by using the appropriate Object Handler profile option. See also the section *Profile Settings*.

You can create standard procedures to define recurring settings and object specifications which automate the processing of the unload, load or scan function. See the section *Workplans* for further information.

This section covers the following topics:

- **Wizards**
- **Advanced User**
- **Restart Load**
- **View**
- **Find**
- **Administration**
- **Change the Workplan Library**
- **Select System File**
- **Select Library**

-  **List and Select Workplan**
-  **Select System Error Messages**
-  **Select Objects**



**Notes:**

1. The topic *Change the Workplan Library* is described in the section *Administration*.
2. The topic *List and Select Workplan* is described in *List the Available Workplans in the Workplan Library* in the section *Administration*.

# 17 Wizards

---

▪ Step 1 - Start the Procedure .....	92
▪ Step 2 - Unload/Load/Scan Objects into/from Work Files .....	93
▪ Step 3 - Set Parameters .....	95
▪ Step 4 - Select Objects .....	96
▪ Step 5 - Execute Processing .....	97
▪ Step 6 - Continue Processing .....	98

The Object Handler provides wizards that determine the processing sequence for the following:

- Unloading data from the Natural system environment into Natural work files.
- Loading data from work files into the Natural system environment.
- Scanning the contents of Natural work files.

▶ **To activate the wizards**

- In the **Main Menu**, select the **Advanced user** field if required (the field is not selected by default).

The wizards provide the keys PF8 and PF7 to navigate between the screens (steps). Use PF12 to cancel the processing sequence.

The steps described in this section show the processing sequence performed with the unload, load or scan wizard.

## Step 1 - Start the Procedure

---

▶ **To start the unload, load or scan procedure**

- 1 From the **Main Menu**, choose **Unload, Load or Scan** by entering any single character next to the function required or by using the corresponding PF key.

The initial **Wizard** screen appears with the following options:

- **Unload/Load/Scan objects into/from Natural work file(s).**
- **Start Object Handler command procedure.**

- 2 If you want to unload objects into a work file, load object from a work file or scan objects in a work file, proceed with [Step 2 - Unload/Load/Scan Objects into/from Work Files](#) below.

Or:

If you want to use a command procedure for unloading, loading or scanning objects, choose **Start Object Handler command procedure** and proceed as follows:

1. On the initial **Wizard** screen, choose **Start Object Handler command procedure**. The **Procedure** screen appears.
2. In the **Name** field, enter the name of a Workplan of the type PROCEDURE by using either of the following options:

- Type in the name of a Workplan of the type PROCEDURE (see also [Workplans](#)) that should be used for the transaction.
  - Choose **Select Workplan** or choose PF5 to display a list of available Workplans of the type PROCEDURE. In the line next to the Workplan you want to select, enter the command S or SE. Choose ENTER to execute the command and fill the **Name** field on the **Procedure** screen.
3. Select **List Workplan** or choose PF4 if you want to display the Workplan specified.

See also [List the Available Workplans in the Workplan Library](#) in the section *Administration*.

- 3 Choose ENTER to continue.
- 4 Proceed with [Step 5 - Execute Processing](#).

## Step 2 - Unload/Load/Scan Objects into/from Work Files

### ▶ To unload, load or scan objects into/from Natural work files

- 1 On the initial **Wizard** screen, choose **Unload/Load/Scan objects into/from Natural work file(s)**.
- 2 Choose ENTER or choose PF8 (Next) to continue. The **Options** screen of the wizard appears with the following fields, commands and alternative PF keys:

Field	PF Key	Explanation
<b>Transfer format</b>		<p>Only valid if <a href="#">Use default options</a> (this is the default) has been selected.</p> <p>If selected, the data to be processed is written in Transfer format to/from the work file. See also <a href="#">Work File Format</a> in <i>Work Files</i>.</p> <p>Unload function: The data to be unloaded is written in Transfer format to the work file. Note that if you want to change the setting of this field for a subsequent unload, you need to return to the <b>Main Menu</b> or enter the command GO UNLOAD END (see <a href="#">Commands for Navigation and Special Functions</a> in <i>Direct Commands</i>) and restart the unload function.</p> <p>Load and scan functions: The data to be loaded or scanned is expected to be in Transfer format.</p>

Field	PF Key	Explanation
Unicode work file		Only applies to the unload function and if <b>Transfer format</b> has been selected.  If this option is selected, all object sources are converted to Unicode/UTF-8 (Universal Transformation Format, 8-bit form) before they are written to the work file.
Use PC File		Only applies if Entire Connection is installed.  If selected, the data to be processed is written to an Entire Connection work file.
PC File		Only valid if <b>Use PC File</b> has been selected.  The name of the path and the Entire Connection work file to be used.
Use default options		Default options are (this is the default). For the options available, see <a href="#">Profile Settings</a> and <a href="#">Set Additional Options</a> in <i>Settings</i> .
Set additional options	PF4	Only valid if <b>Use default options</b> has been selected.  Invokes the <b>Options</b> screen of the wizard where you can modify the default settings and enter additional options for the processing sequence. See <a href="#">Set Additional Options</a> in <i>Settings</i> .
Use Option Workplan		If selected, a Workplan of the type OPTION is used (see <a href="#">Workplans</a> ).
Name		Only valid if <b>Use Option Workplan</b> has been selected.  The name of a Workplan of the type OPTION to be used.
List Option Workplan	PF6	Only valid if <b>Use Option Workplan</b> has been selected.  Displays the contents of the Workplan specified in the field Name.
Select Option Workplan	PF5	Only valid if <b>Use Option Workplan</b> has been selected.  Displays a selection list of available Workplans of the type OPTION (see also <a href="#">List the Available Workplans in the Workplan Library</a> in <i>Administration</i> ).

- 3 Select any of the options provided and (if necessary) complete the fields to be used for the processing sequence.
- 4 Choose ENTER or choose PF8 (Next) to continue.

The **Parameters** screen of the wizard appears.

## Step 3 - Set Parameters

### ▶ To set parameters for the processing procedure

- 1 On the **Parameters** screen, select any of the following options and (if necessary) complete the fields to be used for the processing sequence:

Field	PF Key	Explanation
<b>Do not use parameters</b>		If selected (default setting), no parameters are set.
<b>Use global parameters</b>		If selected, global parameters are used. See <a href="#">Set Global Parameters</a> in <i>Settings</i> .
<b>Set global parameters</b>	PF4	Only valid if <b>Use global parameters</b> has been selected.  If selected, the <b>Parameters</b> screen is invoked. See <a href="#">Set Global Parameters</a> ( <i>Settings</i> ) and <a href="#">parameter-setting</a> ( <i>Direct Commands</i> ) for descriptions of keywords and valid input values.
<b>Use Parameter Workplan</b>		If selected, a Workplan of the type PARAMETER is used (see <a href="#">Workplans</a> ).
<b>Name</b>		Only valid if <b>Use Parameter Workplan</b> has been selected.  The name of a Workplan of the type PARAMETER to be used.
<b>List Parameter Workplan</b>	PF6	Only valid if <b>Use Parameter Workplan</b> has been selected.  If selected, the contents of the Workplan specified in the field <b>Name</b> is displayed.
<b>Select Parameter Workplan</b>	PF5	Only valid if the field <b>Use Parameter Workplan</b> has been selected.  If selected, a selection list of available Workplans of the type PARAMETER is displayed (see <a href="#">List the Available Workplans in the Workplan Library</a> in <i>Administration</i> ).

- 2 Choose ENTER or choose PF8 (Next) to continue.

The **Select Unload/Load/Scan Type** screen appears.

## Step 4 - Select Objects

---

▶ **To select the type of object you want to process**

1 On the **Select Unload/Load/Scan Type** screen, choose one of the three options described below. Note that the first option only applies to the load and scan functions. For the keywords and valid values that apply to each object type, see the relevant explanations in the section *Object Specification*.

1. Select **Load/Scan all objects** to process all objects from the work file.
2. Select a particular type of object:
  - **Natural library objects**
  - **Natural system error messages**
  - **Natural command processor sources**
  - **Natural-related objects**
  - **DDMs**
  - **FDTs**

Choose ENTER or choose PF8 (Next) to continue.

Depending on the type of object selected, a screen appears where you can specify selection criteria for the objects to be processed.

Enter selection criteria and choose **Details** (if available) for further object specifications, if required. For information on **Details**, see the relevant explanation in the section *Object Specification*.

3. Select **Use Selection or List Workplan** to use a Workplan of the type SELECTION or LIST. See also *Workplans*.

Choose ENTER or choose PF8 (Next) to continue.

The **Selection or List** screen appears. In the **Name** field, enter the name of a Workplan of the type SELECTION or LIST by using either of the following options:

- Type in the name of a Workplan.

Or:

Choose **Select Workplan** or PF5 (SelWP) to display a list of all Workplans available. In the line next to the Workplan you want to select, enter either the command S or SE.

Choose ENTER to execute the command and fill the **Name** field on the **Selection or List** screen. See also [List the Available Workplans in the Workplan Library](#) in the section *Administration*.

Choose **List Workplan** or PF4 (Li-WP) if you want to display the contents of the Workplan entered in the **Name** field.

- 2 Choose ENTER or PF8 (Next) to continue.

The wizard displays the processing command generated from the input data.

You can save the command displayed as a Workplan of the type PROCEDURE (see also [Workplans](#)), by entering the command SAVE or by choosing PF5 (Save).

## Step 5 - Execute Processing

---

### ▶ To execute the processing procedure

- 1 On the command execution screen, choose ENTER or choose PF8 (Next) to confirm the settings and to process the objects specified.

If required, choose PF7 (Back) and modify the processing settings before you confirm command execution.

The Object Handler performs the function and displays a confirmation message.

- 2 Choose ENTER to continue.

A report screen appears with a list of the objects processed.

- 3 Choose PF3 (Exit) to leave the report screen or choose PF12 (Canc) to terminate the function.

A window appears where you can choose whether to continue processing data.

- 4 Choose **No** and then ENTER to terminate the function.

Or:

Choose PF12 to terminate the function.

The **Main Menu** appears.

## Step 6 - Continue Processing

---

▶ **To continue processing**

- 1 On the report screen, choose PF3 (Exit).

A window appears where you choose whether to proceed with the next processing step.

- 2 Choose **Yes**.

A screen appears with the option to reuse or change previous settings.

# 18

## Advanced User

---

▪ Activating Advanced User .....	100
▪ Processing Objects .....	100

This section describes how to invoke advanced-user mode and perform the unload, load and scan functions.

## Activating Advanced User

---

### ▶ To activate advanced-user mode

- From the **Main Menu**, select the **Advanced user** field (the field is not selected by default).

Or:

Set advanced-user mode as the default by specifying the `Advanced-Mode` parameter in your Object Handler profile (see [Profile Settings](#)).

## Processing Objects

---

### ▶ To process objects in advanced-user mode

- 1 In the **Main Menu**, check the **Advanced user** field and select **Unload, Load** or **Scan**.
- 2 Choose ENTER to continue.

The **Unload/Load/Scan Settings** screen appears with the sections **Options** and **Parameters**.

- 3 Set the options and parameters as described in the section [Settings](#).
- 4 Choose ENTER to continue.

The **Select Unload/Load/Scan Type** screen appears.

- 5 Select the objects you want to process: see also the section [Object Specification](#).
- 6 Choose **Details** to specify additional selection criteria: see the relevant sections in [Object Specification](#).
- 7 Choose ENTER to continue.

- If the parameter `Display-Cmd-in-Advanced-Mode` is set to N (No) in the Object Handler profile (this is the default), or if no such profile exists, the command generated from the input data is executed immediately after you have specified the selection data. See also [Profile Settings](#).

The **Display Unload/Load/Scan Report** screen appears with a list of the objects processed if the field **Write report** was selected (this is the default). See also [Work File Options](#) in the section [Settings](#).

- If the parameter `Display-Cmd-in-Advanced-Mode` is set to Y (Yes) in the Object Handler profile (see *Profile Settings*), or if the command `SET ADVANCEDCMD ON` (see *Commands for Navigation and Special Functions*) was executed earlier, a screen appears, which displays the command generated from the input data.

You can save the command displayed as a Workplan of the type PROCEDURE (see also *Workplans*), by entering the command `SAVE` or by choosing PF5 (Save).

Choose ENTER to confirm command execution or choose PF3 (Exit) to modify the processing settings before confirming command execution.

The **Display Unload/Load/Scan Report** screen appears with a list of the objects processed if the field **Write report** was selected (this is the default). See also *Work File Options* in the section *Settings*.



# 19 Restart Load

---

You can use the restart load function to resume load functions that terminated abnormally. If the load function terminates before the work file has been processed completely, with the restart load you can continue from the point of termination.

The restart load requires that restart information is written to a text member (Natural object of the type Text) in accordance with the selection criteria, options and parameter settings specified for the load.

## ▶ To set up the environment during the load

1 On the **Load Options** screen:

- Mark the **Write restart information** option.
- In the **Restart text member** field, enter the name of the text member to which the restart information data is written. The text member must be contained in the Workplan library.

Or:

Mark the **Select text member** field next to **Restart text member** and select a text member from the list of Workplans contained in the Workplan library.

The **Load Options** screen is described in *Work File and Report Options* in the section *Settings*.

2 Execute the load function.

## ▶ To execute the restart load after an interrupted load

- In the Command line of the Object Handler screen, enter the following command:

```
GO RESTART
```

The **Restart Options** screen appears, where you can specify a text member by entering a name in the **Restart text member** field or by marking **Select text member** and selecting a text member from a list.

Or:

Use the following direct command:

```
RESTART
```

The syntax of RESTART is shown in the section *Basic Command Syntax*.

**Related Topics:**

*Change the Workplan Library* in the section *Administration*.

GO RESTART in the section *Commands for Navigation and Special Functions*.

# 20

## View

---

▪ Natural Library Objects .....	106
▪ Natural System Error Messages .....	107
▪ Natural Command Processor Sources .....	108
▪ FDTs .....	109
▪ Natural-Related Objects .....	109
▪ DDMs .....	110

This function is used to view all objects contained in your Natural system environment. Depending on the type of object selected, you can also use this function to delete an object if required.

▶ **To invoke the view function**

- In the **Main Menu**, choose **View**.

Or:

On any other Object Handler screen, enter the following direct command:

```
GO VIEW
```

(See also *Commands for Navigation and Special Functions* in the section *Direct Commands*.)

The **Select View Type** screen appears with all types of object available for selection.

This section describes how to view the object types listed on the **Select View Type** screen:

## Natural Library Objects

---

Natural library objects are programming objects and user-defined error messages.

▶ **To view Natural library objects**

- 1 On the **Select View Type** screen, select **Natural library objects**.

The **View System Files** screen appears with a list of all system files available in the current Natural environment.

For explanations of the screen columns, see the **Select System File** screen with identical columns, which are described in *Select System File*.

- 2 In the **Cmd** column, enter any single character next to the system file you want to select. The current FUSER or FNAT system file is selected by default.

The **View Libraries** screen appears with a list of all libraries available in the system file specified.

You can start the list of libraries from a particular library, or filter objects by entering a library name or a range of names in the **Library** field. For valid name ranges, see *Name* in the section *Name, Date and Time Specification*.

3

- 4 In the **Cmd** column, next to the library you want to select, enter one of the following line commands:

L

LI

S

SE

The **View Library Objects** screen appears with a list of all objects contained in the library specified.

For explanations of this screen, see the description of the **List** screen, which has identical **columns** described in *Select Objects*.

- 5 In the **Cmd** column, next to the object you want to view, enter either of the following line commands:

L

LI

Or:

If required, next to an object you want to delete, enter the following line command:

DE

Depending on the command entered, either the source code of the object selected is displayed on the screen or a confirmation window appears, which is used to execute the delete function.

## Natural System Error Messages

### ▶ To view Natural system error messages

- 1 On the **Select View Type** screen, select **Natural system error messages**.

The **View System Error Messages** screen appears.

- 2 If the system error messages required are not stored in the current FNAT or FUSER system file, replace the database ID in the **DBID** field and the file number in the **FNR** field. If required, specify the Adabas password (**Password**) and the cipher code (**Cipher**) for the system file.

If required, specify selection criteria in the fields **Error number range**, **Languages** and **Short/Long/All**. These fields correspond to the **columns** **Number**, **Language** and **S/L** respectively that appear when the view function is executed and the list of error messages is displayed on the **View System Error Messages** screen.

For explanations of this **View System Error Messages** screen, see the description of the **List System Error Messages** screen, which has identical **columns**.

- When the list of the system error messages selected is displayed on the screen, in the **Cmd** column, next to the error message you want to view, enter either of the following line commands:

```
L
```

```
LI
```

Or:

You can delete an error message by entering the following line command in the **Cmd** column, next to the object required:

```
DE
```

Depending on the command entered, either the source code of the error message selected is displayed on the screen or a confirmation window appears, which is used to execute the delete function.

## Natural Command Processor Sources

### ▶ To view Natural command processor sources stored in an Adabas file

- On the **Select View Type** screen, select **Natural command processor sources**.

The **View Natural Command Processors** screen appears.

- If the Natural command processor sources required are not stored in the current FUSER system file, enter the required database ID in the **DBID** field and the file number in the **FNR** field.

If required, enter an Adabas password in the **Password** field and a cipher code in the **Cipher** field.

The **View Libraries** screen appears with a list of all libraries where Natural command processor sources are stored.

- You can start the list of libraries from a particular library, or filter Natural command processor sources by entering a library name or a range of names in the **Library** field. For valid name ranges, see **Name** in the section *Name, Date and Time Specification*.
- In the **Cmd** column, next to the library you want to select, enter one of the following line commands:

```
L
```

```
LI
```

S

SE

The **View Command Processors** screen appears with a list of all Natural command processor sources contained in the library specified.

For explanations of this screen, see the description of the **List** screen, which has identical **columns**.

- 5 You can delete an object by entering the following line command in the **Cmd** column, next to the object required:

DE

A confirmation window appears, which is used to execute the delete function.

## FDTs

### ▶ To view the FDTs available in an Adabas database

- 1 On the **Select View Type** screen, select **FDTs**.

The **View FDTs** screen appears.

- 2 If the objects required are not stored in the current FNAT or FUSER system file, replace the database ID in the **DBID** field and, if required, the range of file numbers entered in the **FNR from** and **FNR to** fields.

The **View FDTs for DBID** screen appears with a list of all FDTs in the file range and for the database specified.

## Natural-Related Objects

### ▶ To view Natural-related objects

- 1 On the **Select View Type** screen, select **Natural-related objects**.

The **Select Natural-Related Type** screen appears.

- 2 Select the Natural-related type of object you want to view: profile, debug environment or DL/I subfile.

A **View** screen appears where you can specify the location of the Natural-related objects you want to view.

- 3 If the objects required are not stored in the current system file (FNAT for profiles, FUSER for debug environments, FDIC for DL/I subfiles), enter the required database ID in the **DBID** field and a file number in the **FNR** field.

If required, enter an Adabas password in the **Password** field and a cipher code in the **Cipher** field.

A list of all Natural-related objects of the type and system file specified is displayed on the screen.

For explanations of this screen, see the description of the **List** screen, which has identical **columns**.

- 4 In the **Cmd** column, next to the object you want to view, enter either of the following line commands:

```
L
```

```
LI
```

Or:

You can delete an object by entering the following line command in the **Cmd** column, next to the object required:

```
DE
```

Depending on the command entered, either the source code of the Natural-related object selected is displayed on the screen or a confirmation window appears, which is used to execute the delete function.

## DDMs

---

### ▶ To view Natural DDMs

- 1 On the **Select View Type** screen, select **DDMs**.

The **View DDMs** screen appears.

- 2 If the DDMs required are not stored in the current FDIC system file, enter the required database ID in the **DBID** field and the file number in the **FNR** field.

If required, enter an Adabas password in the **Password** field and a cipher code in the **Cipher** field.

An alphabetical list of all DDMs contained in the specified system files is displayed on the screen.

---

For explanations of this **View DDMs** screen, see the description of the **List** screen, which has identical **columns**.

- 3 In the **Cmd** column, next to the DDM you want to view, enter either of the following line commands:

```
L
```

```
LI
```

Or:

You can delete a DDM by entering the following line command in the **Cmd** column, next to the DDM required:

```
DE
```

Depending on the line command entered, either the source code of the DDM selected is displayed on the **List DDM** screen or a confirmation window appears, which is used to execute the delete function.



# 21 Find

---

This function is used to locate objects in your Natural environment and generate a list of the objects found.

▶ **To invoke the find function**

- On any Object Handler screen, in the Command line, enter the following:

```
GO FIND
```

For information on the columns that appear on the report screen generated by the find function, refer to the section *Object Specification*. For the subcommands provided with `GO FIND`, refer to *Commands for Navigation and Special Functions* in the section *Direct Commands*.



# 22 Administration

---

- List the Available Workplans in the Workplan Library ..... 116
- Create a New Workplan ..... 118
- Change the Workplan Library ..... 120

This function is used to maintain Object Handler Workplans.

For information on Workplans and the syntax that applies, refer to the sections [Workplans](#) and [Direct Commands](#).

This section describes the options provided on the **Administration** screen. Instructions for modifying a Workplan are provided in [List the Available Workplans in the Workplan Library](#).

## List the Available Workplans in the Workplan Library

---

This function is used to list all Workplans contained in the Workplan library and to select a Workplan for further processing such as editing or executing the Workplan.

### ▶ To list Workplans

- On the **Administration** screen, select **List the available Workplans in the Workplan library** or choose PF4 (List).

The **List Workplans** screen appears with a list of all Workplans contained in the Workplan library.

If the Natural object of the type Text is a Workplan, the type of Workplan and the first 50 bytes of the Workplan description are listed. You can choose PF5 to display additional information.

The **List Workplans** screen is also invoked with the select function, which is provided, for example, on the **Unload/Load/Scan Settings** screen.

The columns displayed on the **List Workplans** screen and the commands that can be executed on a Workplan are described in the following section.

- [Columns and Commands on the List Workplans screen](#)

### Columns and Commands on the List Workplans screen

The columns and commands provided on the **List Workplans** screen are explained in the following table.

You can use the input fields below each column heading to start the list from a particular Workplan or filter Workplans. Valid input values are mentioned in the table below.

Column	PF Key	Explanation
<b>Cmd</b>		The following line commands can be entered in the input field next to the Workplan required:
	C or CH	Checks the syntax. Only applies to Workplans of the types PROCEDURE, SELECTION, PARAMETER and OPTION.
	DE	Deletes the Workplan.
	ED	Edits the Workplan.  You can modify the name of a Workplan or its description in the <b>Save Workplan</b> window described in <a href="#">Saving a Workplan</a> .
	EX	Executes the Workplan. Only applies to Workplans of the type PROCEDURE.
	L or LI S or SE	Lists the Workplan.  Selects the Workplan to be used for the current function. Only applies if the <b>List Workplans</b> screen is invoked with the select function, for example, from the <b>Unload/Load/Scan Settings</b> screen.
<b>Name</b>		The name of the Workplan.  You can enter a name or a range of names as described in <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .
<b>Type</b>		The type of Workplan such as PROCEDURE.  Valid input values are:  PROCEDURE or P SELECTION or S LIST or L PARAMETER or A OPTION or O TEXT or T  You can also enter an asterisk (*) for all types, or any combination of the short types, for example SL.

Column	PF Key	Explanation
<b>Description</b>		The description of the Workplan.  You can enter a description or a range of descriptions as described in <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>User ID</b>		Only displayed with PF5.  The ID of the user who created the Workplan.  You can enter a user ID or a range of user IDs as described in <i>Name</i> .
<b>Date</b>		Only displayed with PF5.  The date when the Workplan was created.  You can enter a date or a range of dates as described in <i>Date</i> in <i>Name, Date and Time Specification</i> .
<b>Time</b>		Only displayed with PF5.  The time when the Workplan was created.  You can enter a time or a range of times as described in <i>Time</i> in <i>Name, Date and Time Specification</i> .
	PF4	Switches from the additional information display (PF5) to the standard display.
	PF5	Displays additional information: <b>user ID</b> , <b>date</b> and <b>time</b> .

## Create a New Workplan

This function invokes the **Create a new Workplan** screen where you can specify the type of the new Workplan and the format to be used for editing the Workplan.

If you do not select the **Free Format Editing** option (field not marked; this is the default setting), for Workplans of the types OPTION, PARAMETER and SELECTION, screens with input fields are provided.

If you select the **Free Format Editing** option (field marked) or if you create a Workplan of another type, you will obtain a map with an edit area where you can enter the contents of the Workplan; see also *Contents of Workplans* in the section *Workplans*.

For alternative direct command that can be used to set free format editing on and off, see the command **SET** in *Commands for Navigation and Special Functions* in the section *Direct Commands*.

This section covers the following topics:

- [Creating a PROCEDURE Workplan](#)
- [Creating a LIST Workplan](#)

- [Saving a Workplan](#)

## Creating a PROCEDURE Workplan

You can create a Workplan of the type PROCEDURE from the command generated for the current Object Handler function.

### ▶ To create a PROCEDURE Workplan from a generated command

- 1 Execute the function you want to use for the Workplan with an Object Handler wizard until the command generated for the function to be executed is displayed on the screen.

Or:

In advanced-user mode, activate the display of the generated command by choosing either of the following methods:

- Enter the following Object Handler command:

```
SET ADVANCEDCMD ON
```

Or:

In the Object Handler profile, set the parameter `Display-Cmd-in-Advanced-Mode` to Y (Yes). For details, see [Profile Settings](#).

- Execute the function you want to use for the Workplan until the command generated for the function to be executed is displayed on the screen.

- 2 Choose PF5 (Save).

The **Save Workplan** window appears.

- 3 Enter name and description of the new Workplan and choose ENTER.

The Workplan is saved as a PROCEDURE Workplan in the Workplan library. It contains the command generated for the current function.

## Creating a LIST Workplan

For details on creating a Workplan of the type LIST, refer to the section [Object List - LIST Workplan](#).

## Saving a Workplan

### ▶ To save a Workplan

- 1 When you have finished editing a Workplan, in the Command line, enter the following command:

```
SAVE
```

Or:

Choose PF5 (Save).

The **Save Workplan** window appears.

- 2 Enter or modify name and description of the Workplan and choose ENTER.

The Workplan is saved under the specified name in the Workplan library.

## Change the Workplan Library

This function is used to change the Workplan library. All Workplans must be stored in a Workplan library.



**Note:** You can also set the default library for Workplans by specifying the `Workplan-Library` parameter in your Object Handler profile (see [Profile Settings](#)).

The **Change Workplan Library** screen provides the following fields:

Field	Explanation
<b>Library</b>	The name of the Workplan library. Default is the library WORKPLAN.
<b>Select library</b>	Displays a list of all Workplan libraries available: see also <a href="#">Select Library</a> . Equivalent PF key: PF4 (SeLib)
<b>DBID/FNR</b>	Specifies the database ID (DBID) and file number (FNR) where the Workplan library is located. If no values are specified, the current FUSER or FNAT system file is used.
<b>Passw./Ciph.</b>	The password and cipher code of the Adabas file where the Workplan library is located.

# 23

## Select System File

---

You can select the system files to be used for the unload function from a list.

You can produce this selection list from an object specification screen of the unload function when performed in advanced-user mode.

The instructions below are an example of using the function when unloading Natural library objects.

▶ **To select a system file from a list**

- 1 On the **Unload Natural Library Objects** screen, choose **Select DBID/FNR** or choose PF5 (DBIDs).

The **Select System File** window appears where the system files available in the current Natural environment are listed with their names (**System File**), database IDs (**DBID**) and file numbers (**FNR**). *User defined* denotes a system file specified by the user.

- 2 Select the system file you want to use for function processing by entering any single character in the **Sel** column next to the system file required. The FUSER system file is selected by default.

Or:

In the **DBID** and **FNR** fields next to *User-defined*, you can enter the database ID and file number of the system file you want to select. If required, in the **Password/VSAM name** and **Cipher Code** columns, enter the Adabas password or VSAM name and the Adabas cipher code for the system file.

The database ID and file number of the system file selected are entered in the **DBID/FNR** fields of the **Unload Natural Library Objects** screen. If you selected the default system file, these fields remain empty.



# 24

## Select Library

---

You can select the library to be used for the unload function from a list.

The selection list is produced with the **Select library** (or **Select**) function, which is provided on the object specification screen of the unload function when performed in advanced-user mode.

The instructions below are examples of selecting single or multiple libraries when unloading Natural library objects.

### ▶ To select a single library from a list

- 1 On the **Unload Natural Library Objects** screen, choose **Select library** or PF4 (SeLib).

The **Select Library** window appears with a list of all libraries and the database IDs (**DBID**) and file numbers (**FNR**) of the system file where the libraries are stored.

- 2 In the **Cmd** column, next to the library required, enter any single character.
- 3 Choose ENTER.

The **Library** field and the **DBID/FNR** fields of the **Unload Natural Library Objects** screen are filled with the specified name and numbers respectively. If no values (or 0) are entered in the **DBID/FNR** fields, the current FUSER and FNAT system files are selected.

### ▶ To list and select multiple libraries

- 1 On the **Unload Natural Library Objects** screen, choose **Select library** or PF4 (SeLib).

The **Select Library** window appears with a list of all libraries and the database IDs (**DBID**) and file numbers (**FNR**) of the system file where the libraries are stored.

- 2 In the **Library** field, enter a name or a range of names to filter the libraries you want to select. If you enter a single library name, the list will start with this library. For valid name ranges, see *Name* in the section *Name, Date and Time Specification*.

Or:

In the **DBID** and **FNR** fields, enter the database ID and file number of the system file that contains the libraries you want to select. If no values (or 0) are entered, the current FUSER and FNAT system files are used.

Note that **DBID** and **FNR** are read-only fields when the **Select Library** window has been invoked from an **Exceptions** screen.

3 Choose ENTER.

The **Select Library** window now lists all libraries of the specified range.

4 Choose PF4 (Se Rng).

The **Library** field and the **DBID/FNR** fields of the **Unload Natural Library Objects** screen are filled with the specified name (or range) and numbers respectively. If no values (or 0) are entered in the **DBID/FNR** fields, the current FUSER and FNAT system files are selected.

# 25

## Select System Error Messages

---

- Columns and Commands ..... 126

You can select the Natural system error messages to be unloaded from a list.

You can produce this selection list from an object specification screen of the unload function when performed in advanced-user mode.

▶ **To select Natural System error messages**

- On the **Unload Natural System Error Messages** screen, if required, change the message numbers in the **Error number from/to** fields (default is the full range of numbers) and select **Select system error messages**.

The **List System Error Messages** screen appears with a list of all system error messages contained in the system file specified.

This screen is described in the following section.



**Note:** The select function for user-defined error messages is described in the section [Select Objects](#).

## Columns and Commands

The columns and commands provided on the **List System Error Messages** screen are explained in the following table.

You can use the input fields below each column heading to start the list from a particular system error message or filter messages. Valid input values are mentioned in the table below.

Column	PF Key	Explanation
Cmd		One of following line commands can be entered in the input field next to the system error message required:
	L or LI	Lists the short and long texts of the message.
	S or SE UL or U	Selects the message for subsequent unloading. <b>Attention:</b> Any of these commands only marks the message selected for subsequent processing. To execute the unload function, you need to choose <b>PF2</b> (Unloa) described below.
	DE	Deletes the message.

Column	PF Key	Explanation						
		DL Only deletes the long text of the message.						
<b>Number</b>		<p>The number of the system error message.</p> <p>You can enter a number or a range of numbers. Valid ranges are:</p> <p><i>value</i>* All messages with numbers that begin with <i>value</i>.</p> <p><i>value</i>&gt; All messages with numbers greater than or equal to <i>value</i>. Example: 10&gt;</p> <p><i>value</i>&lt; All messages with numbers less than or equal to <i>value</i>. Example: 100&lt;</p>						
<b>S/L</b>		<p>The kind of system error message text:</p> <table border="1"> <tr> <td>S</td> <td>Short text.</td> </tr> <tr> <td>L</td> <td>Long text.</td> </tr> <tr> <td>A</td> <td>Short and/or long text.</td> </tr> </table>	S	Short text.	L	Long text.	A	Short and/or long text.
S	Short text.							
L	Long text.							
A	Short and/or long text.							
<b>Language</b>		<p>The language code of the system error message.</p> <p>You can enter up to 8 valid language codes (for example, 1 for English) for the error messages to be selected.</p> <p>An asterisk (*) selects all language codes.</p>						
<b>Error Message Text</b>		The short text of the system error message.						
	PF2	<p>Starts unloading the system error messages selected for processing.</p> <p>As an alternative, in the Command line, you can enter either of the following direct commands:</p> <p>UNLOAD or UNLD</p>						
	PF11	<p>Marks all system error messages listed for subsequent unloading with <b>PF2</b>.</p> <p>As an alternative, in the Command line, you can enter either of the following direct commands:</p> <p>SELECT ALL or SEL ALL</p>						



# 26

## Select Objects

---

- Columns and Commands on List Screens ..... 130

You can select the objects to be unloaded from a list. This selection list can also be used for other purposes such as listing the source of an object or deleting it.

The selection list is produced with the **Select objects** (or **Select**) function, which is provided on the object specification screen of the unload function when performed in advanced-user mode.

The selection list is displayed on a **List** screen, which is described in the following section.



**Note:** The select function for Natural system error messages is described in the section [Select System Error Messages](#).

## Columns and Commands on List Screens

The columns and commands provided on a **List** screen are explained in the following table.

The display of the columns contained on a **List** screen depends on the type of object selected from the **Select Unload Type** menu. The type of object processed is contained in the screen title, for example, **List Library Objects** or **List Command Processors**.

You can use the input fields below each column heading to start the list from a particular object or filter objects. Valid input values are mentioned in the table below.

Column	PF Key	Explanation
<b>Cmd</b>		One of following line commands can be entered in the input field next to the object required:
	L or LI	Lists the source code of the object (not applicable to Natural command processor sources).  For a user-defined error message: lists the short and the long texts of the error message.
	S or SE UL or U	Selects the object for subsequent unloading.  <b>Attention:</b> Any of these commands only marks the object selected for subsequent processing. To execute the unload function, you need to choose <b>PF2</b> (Unloa) described below.
	DE	Deletes the object.
<b>Name</b>		The object name.  You can enter a name or a range of names: see <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .

Column	PF Key	Explanation												
		For a user-defined error message, the message number and the language code is displayed. For example: 10 (Lang =1) denotes message number 10 in language 1 (English).												
<b>Type</b>		<p>The type of Natural library object such as Program.</p> <p>Valid input values are one or more object-type codes such as P for program. For a list of codes, see <a href="#">NATTYPE</a> in the section <i>select-clause</i>.</p> <p>For a Natural-related object of the type profile or DL/I subfile, this column contains the type of profile (such as Editor) or DL/I subfile (such as NBD Subfile).</p>												
<b>S/C</b>		<p>The kind of Natural library object: by default, all source (S) objects and/or cataloged (C) objects available are displayed on the screen.</p> <table border="1" data-bbox="423 674 1474 1073"> <tr> <td colspan="2">Valid input values are one or more of the following codes:</td> </tr> <tr> <td>S</td> <td>Source objects only.</td> </tr> <tr> <td>C</td> <td>Cataloged objects only.</td> </tr> <tr> <td>S/C</td> <td>Both source and cataloged objects if both exist.</td> </tr> <tr> <td>W</td> <td>All STOWed objects: source and cataloged objects with identical date and time.</td> </tr> <tr> <td>*</td> <td>All source objects and/or cataloged objects.</td> </tr> </table> <p>For a user-defined error message, this column contains the short text of the error message.</p>	Valid input values are one or more of the following codes:		S	Source objects only.	C	Cataloged objects only.	S/C	Both source and cataloged objects if both exist.	W	All STOWed objects: source and cataloged objects with identical date and time.	*	All source objects and/or cataloged objects.
Valid input values are one or more of the following codes:														
S	Source objects only.													
C	Cataloged objects only.													
S/C	Both source and cataloged objects if both exist.													
W	All STOWed objects: source and cataloged objects with identical date and time.													
*	All source objects and/or cataloged objects.													
<b>M</b>		<p>The programming mode of the Natural library object. By default, any mode is displayed.</p> <p>Valid input values are one or more of the following codes:</p> <p>S     Structured mode only.  R     Reporting mode only.  *     Any mode, structured and/or reporting.</p> <p>For a user-defined error message, this column contains the short text of the error message.</p>												
<b>Version</b>		<p>The Natural version under which the Natural library object was saved and/or cataloged.</p> <p>You can enter a version using the following format: <i>V.R.S</i> where <i>V</i> is the 1-digit version, <i>R</i> the 1-digit release, and <i>S</i> the 2-digit system maintenance level. For valid ranges of versions, see <a href="#">Name</a> in <i>Name, Date and Time Specification</i>.</p> <p>For a user-defined error message, this column contains the short text of the error message.</p>												
<b>User ID</b>		<p>The ID of the user who saved or cataloged the Natural library object or DDM.</p> <p>You can enter a single user ID or a range of user IDs: see <a href="#">Name</a>.</p> <p>For a user-defined error message, this column contains the short text of the error message.</p>												

Column	PF Key	Explanation
<b>Date</b>		The date when the Natural library object or DDM was saved or cataloged. You can enter a date or a range of dates: see <i>Date</i> in <i>Name, Date and Time Specification</i> .  For a user-defined error message, this column contains the short text of the error message.
<b>Time</b>		The time when the Natural library object or DDM was saved or cataloged. You can enter a time or a range of times: see <i>Time</i> in <i>Name, Date and Time Specification</i> .  For a user-defined error message, this column contains the short text of the error message.
<b>DBID</b>		The database ID of the system file where DDMs or Natural-related objects are stored.
<b>FNR</b>		The file number of the system file where DDMs or Natural-related objects are stored.
	PF2	Starts unloading the objects selected for processing.  As an alternative, in the Command line, you can enter either of the following direct commands:  UNLOAD or UNLD
	PF11	Marks all objects listed for subsequent unloading with <b>PF2</b> .  As an alternative, in the Command line, enter either of the following direct commands:  SELECT ALL or SEL ALL

# 27

## Object Specification

---

The Object Handler provides the **Select Unload/Load/Scan Type** screen where you can select the types of object to be processed or specify a Workplan of the type SELECTION or LIST.

For each type of object selected, you are provided individual object-specification screens. These screens are used to specify selection criteria for the objects to be processed.

This section describes the options provided on each object-specification screen. If a field or function key (PF key) described in this section only appears with a particular function and/or in advanced-user mode, this is indicated by an appropriate remark such as “Only applies to the unload function in advanced-user mode”.

- **All Objects on the Work File**
- **Natural Library Objects**
- **Natural System Error Messages**
- **Natural Command Processor Sources**
- **Natural-Related Objects**
- **DDMs**
- **FDTs**
- **Use Selection or List Workplan**



## 28 Object Specification - All Objects on the Work File

---

**Only applies to the load or scan function.**

The option **Load/Scan All Objects on the Work File** is used to select all objects available in the work file for processing. In advanced-user mode, from the **Load/Scan All Objects** screen, you can invoke the **Settings** screen where you can specify option and parameter settings. See the section [Settings](#).



# 29

## Object Specification - Natural Library Objects

---

- Natural Library Objects ..... 138
- Natural Library Object Details ..... 139
- Natural Library Object Properties ..... 141
- Natural Library Object Exceptions ..... 142
- Natural Library Object Exception Properties ..... 143

This section describes the options provided on the object-specification screens for processing Natural library objects. Natural library objects are programming objects and user-defined error messages.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

## Natural Library Objects

The screen **Unload/Load/Scan Natural Library Objects** provides the following fields and PF keys:

Field	PF Key	Explanation
DBID/FNR		Only applies to the unload function.  The database ID (DBID) and file number (FNR) of the system file where the Natural libraries are stored. If no values (or 0) are specified, the current FUSER or FNAT system file is used.
Select DBID/FNR	PF5  (advanced-user mode only)	Only applies to the unload function.  Displays a selection list of system files available.
Password/Cipher		Only applies to the unload function.  The password and cipher code for the Adabas file where the Natural libraries are stored.
Library		The name of a library or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
Select library	PF4	Displays a selection list of all libraries available. See also <i>Select Library</i> .
Object name		The name of a Natural programming object or a range of names: see <i>Name</i> .  Only evaluated if the field <b>Natural programming objects</b> (default setting) is selected on the screen <b>Natural Library Objects, Details</b> . See also <i>Natural Library Object Details</i> .
Select objects		Only applies to the unload function in advanced-user mode.  If no library range is specified, a selection list of all Natural objects available is displayed (see also <i>Select Objects</i> ).
Error number from/to		A valid range (1 - 9999) of user-defined error messages delimited by the first and the last message number.  Only evaluated if the field <b>Error messages</b> (default setting) is selected on the screen <b>Natural Library Objects, Details</b> (see also <i>Natural Library Object Details</i> ).

Field	PF Key	Explanation
<b>Details</b>	PF6	Invokes the screen <b>Natural Library Objects, Details</b> where you can enter more detailed object specifications. See <a href="#">Natural Library Object Details</a> .
<b>Settings</b>	PF7	Only applies to functions executed in advanced-user mode.  Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings: see <a href="#">Settings</a> .

## Natural Library Object Details

The screen **Unload/Load/Scan Natural Library Objects, Details** is used to specify further selection criteria for Natural library objects.

For descriptions of keywords and valid input values, see also [select-clause](#) in the section *Direct Commands*.

The screen **Unload/Load/Scan Natural Library Objects, Details** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>Library</b>		The name of a library or a range of names: see <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .  Ranges are not allowed if the <a href="#">Use Predict set</a> option is selected.
<b>Select</b> (Library)	PF4	Displays a selection list of all libraries available. See also <a href="#">Select Library</a> .
<b>DBID/FNR</b>		See <a href="#">DBID/FNR</a> in <i>Natural Library Objects</i> above.
<b>Passw./Ciph.</b>		Only applies to the unload function.  The password and cipher code of the Adabas file where the Natural libraries are stored.
Object Types: <b>Natural programming objects</b>		Natural programming objects.
Object Types: <b>Error messages</b>		User-defined error messages.
<b>Object name</b>		See <a href="#">Object name</a> in <i>Natural Library Objects</i> above.
<b>Use Predict set</b>		Only applies to the unload and find functions and if Predict is installed.

Field	PF Key	Explanation														
		<p>This option is used to read the names of the objects to be processed from a retained set. A retained set is created with the save set option of the LIST XREF command.</p> <p>If the <b>Use Predict set</b> option is selected, the following applies:</p> <ul style="list-style-type: none"> <li>■ The <b>Object name</b> field must contain asterisk (*) indicating all objects. This is the default setting.</li> <li>■ The <b>Library</b> field must contain the name of a single library. Name ranges are not allowed.</li> <li>■ The <b>Set number</b> field must be filled.</li> </ul> <p>For detailed information on Predict sets, refer to the <i>Predict</i> documentation.</p>														
<b>Set number</b>		<p>Only applies if <b>Use Predict set</b> is selected.</p> <p>A one- or two-digit number that identifies the retained set to be used.</p>														
<b>Set library</b>		<p>Only applies if <b>Use Predict set</b> is selected.</p> <p>The name of the library to be searched for a Predict set. If you do not specify a name, the library entered in the <b>Library</b> field is used by default.</p>														
<b>Set user</b>		<p>Only applies if <b>Use Predict set</b> is selected.</p> <p>The ID of the user who created the retained set. If no ID is entered, the ID specified with the system variable *USER (see the <i>System Variables</i> documentation) is used.</p>														
Programming Object Options:  <b>S/C-Kind</b>		<p>The kind of Natural programming object:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 20%;"></td> <td></td> </tr> <tr> <td>S</td> <td>Source objects only.</td> </tr> <tr> <td>C</td> <td>Cataloged objects only.</td> </tr> <tr> <td>A or *</td> <td>All source objects and/or cataloged objects. This is the default setting.</td> </tr> <tr> <td>W</td> <td>All STOWed objects: source and cataloged objects with identical date and time.</td> </tr> <tr> <td>B</td> <td>Both source and cataloged objects if both exist.</td> </tr> <tr> <td></td> <td></td> </tr> </table> <p><b>Note:</b> W and B are valid for the unload function only. Though W and B can also be entered for the load or scan function, they are treated like A.</p>			S	Source objects only.	C	Cataloged objects only.	A or *	All source objects and/or cataloged objects. This is the default setting.	W	All STOWed objects: source and cataloged objects with identical date and time.	B	Both source and cataloged objects if both exist.		
S	Source objects only.															
C	Cataloged objects only.															
A or *	All source objects and/or cataloged objects. This is the default setting.															
W	All STOWed objects: source and cataloged objects with identical date and time.															
B	Both source and cataloged objects if both exist.															
Programming Object Options:  <b>Natural types</b>		<p>A Natural object-type code such as P for program. For a list of valid codes, see NATTYPE in the section <i>select-clause</i>.</p>														

Field	PF Key	Explanation								
Select Natural types	PF6	Invokes a window where you can select one or more types of Natural object.								
Properties	PF7	Invokes an extra screen where you can specify additional properties of Natural programming objects: see <a href="#">Natural Library Object Properties</a> .								
Error Messages: Error number from/to		A range of user-defined error messages as entered in the <b>Error number from/to</b> fields (see <i>Natural Library Objects</i> above).								
Error Messages: Language codes		Up to 8 valid language codes (for example, code 1 for English) of the specified error messages.  An asterisk (*) selects all language codes.								
Error Messages: S/L-Kind		The kind of error message text: <table border="1" data-bbox="641 730 1474 961"> <tbody> <tr> <td>S</td> <td>Short text.</td> </tr> <tr> <td>L</td> <td>Long text.</td> </tr> <tr> <td>A</td> <td>Short and/or long text. This is the default.</td> </tr> <tr> <td>B</td> <td>Short and long texts if both exist (unload function only).</td> </tr> </tbody> </table>	S	Short text.	L	Long text.	A	Short and/or long text. This is the default.	B	Short and long texts if both exist (unload function only).
S	Short text.									
L	Long text.									
A	Short and/or long text. This is the default.									
B	Short and long texts if both exist (unload function only).									
Exceptions	PF8	Invokes an extra screen where you can specify exceptions to the selection of Natural programming objects: see <a href="#">Natural Library Object Exceptions</a> .								

## Natural Library Object Properties

The screen **Unload/Load/Scan Library Objects, Properties** is used to specify properties for the Natural library objects selected for processing.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan Library Objects, Properties** provides the following fields:

Field	Explanation				
User ID	The ID of the user who saved or cataloged a Natural programming object. Specify a single user ID or a range of user IDs: see <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .				
Programming mode	The programming mode of the Natural programming objects: <table border="1" data-bbox="641 1747 1474 1883"> <tbody> <tr> <td>R</td> <td>Reporting mode only.</td> </tr> <tr> <td>S</td> <td>Structured mode only.</td> </tr> </tbody> </table>	R	Reporting mode only.	S	Structured mode only.
R	Reporting mode only.				
S	Structured mode only.				

Field	Explanation
	A No mode check performed. This is the default setting.
<b>Natural version</b>	The Natural version of the Natural programming objects.  Valid version format: <i>VRS</i> where <i>V</i> is the 1-digit version, <i>R</i> the 1-digit release, and <i>S</i> the 2-digit system maintenance level.  You can also specify a range of versions: see <i>Name</i> .
Object Date: <b>Select all objects (no date check)</b>	Selects all objects, regardless of their date.
Object Date: <b>Select objects modified between/and</b>	Selects all objects with a save or catalog date and/or time within the range specified in these fields by entering a precise start date and/or time and/or an end date and/or time.  For valid input values, see <i>Date</i> and <i>Time</i> in <i>Name, Date and Time Specification</i> . Special dates allowed are: TODAY, YESTERDAY, MONTH and YEAR.
Object Date: <b>Select objects modified on</b>	Selects all objects with a save or catalog date and/or time that fits the date/time specified in these fields by entering a precise date and/or time.  For valid input values, see <i>Date</i> and <i>Time</i> . Special dates allowed are: TODAY and YESTERDAY.
Object Size: <b>Select all objects (no size check)</b>	Selects all objects, regardless of their size.
Object Size: <b>Select objects with size between/and</b>	Selects all objects with a size within the range specified in these fields by entering a start size and/or an end size.
Object Size: <b>Select objects with size</b>	Selects all objects with a size that fits the size specified in this field.

## Natural Library Object Exceptions

The screen **Unload/Load/Scan Library Objects, Exceptions** is used to specify exceptions to the selection of Natural library objects.

All objects that match the selection criteria specified in *Natural Library Objects*, *Natural Library Object Details* and *Natural Library Object Properties* are checked against the specifications made on the screen **Unload/Load/Scan Library Objects, Exceptions**. Objects that match *all* specifications defined as exceptions, are exempted from processing.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan Library Objects, Exceptions** is basically identical to the screen **Unload/Load/Scan Natural Library Objects, Details**. See the relevant section for explanations of the fields, commands and alternative PF keys listed in the table below. The field **Add/change properties for selection** is used to specify additional properties for Natural programming object exceptions: see *Natural Library Object Exception Properties*.

Field	PF Key
Library	PF4
Select (Library)	
Object Types: Natural programming objects Error messages	
Object name	
S/C-Kind	
Natural types	
Select Natural types	PF6
Properties	PF7
Error number	
S/L-Kind	
Languages	

## Natural Library Object Exception Properties

The screen **Unload/Load/Scan Library Objects, Exceptions** is used to specify exceptions to the properties of the Natural library objects selected for processing.

The screen provides the following fields:

Field	Explanation
<b>User ID</b>	See <b>User ID</b> in <i>Natural Library Object Properties</i> .
<b>Programming mode</b>	See <b>Programming mode</b> in <i>Natural Library Object Properties</i> .
<b>Natural version</b>	See <b>Natural version</b> in <i>Natural Library Object Properties</i> .
Object Date: <b>Ignore object date</b>	Performs no date check. Objects are processed, regardless of their date.
Object Date: <b>Exclude objects modified between/and</b>	Exempts from processing all objects with a save or catalog date and/or time within the range specified in these fields by entering a precise start date and/or time and/or an end date and/or time.  For valid input values, see <i>Date</i> and <i>Time</i> in <i>Name, Date and Time Specification</i> . Special dates allowed are: TODAY, YESTERDAY, MONTH and YEAR.
Object Date: <b>Exclude objects modified on</b>	Exempts from processing all objects with a save or catalog date and/or time that fits the date/time specified in these fields by entering a precise date and/or time.  For valid input values, see <i>Date</i> and <i>Time</i> . Special dates allowed are: TODAY and YESTERDAY.
Object Size: <b>Ignore object size</b>	Performs no size check. Objects are processed, regardless of their size.
Object Size: <b>Exclude objects with size between/and</b>	Exempts from processing all objects with a size within the range specified in these fields by entering a start size and/or an end size.
Object Size: <b>Exclude objects with size</b>	Exempts from processing all objects with a size that fits the size specified in this field.

# 30

## Object Specification - Natural System Error Messages

---

- Natural System Error Messages ..... 146
- Natural System Error Message Details ..... 146
- Natural System Error Message Exceptions ..... 147

This section describes the options provided on the object-specification screens for processing Natural system error messages from the specified system file.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

## Natural System Error Messages

---

The screen **Unload/Load/Scan Natural System Error Messages** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>Error number from/to</b>		A range of Natural system error messages delimited by the first and the last message number. Select <b>Select system error messages</b> for a list of all system error messages available.
<b>Details</b>	PF6	Invokes the screen <b>Unload/Load/Scan Natural Library Objects, Details</b> where you can enter more detailed object specifications: see <i>Natural System Error Message Details</i> .
<b>Settings</b>	PF7	Invokes the screen <b>Unload/Load/Scan Settings</b> where you can specify option and parameter settings. See <i>Settings</i> .

## Natural System Error Message Details

---

The screen **Unload/Load/Scan System Error Messages, Details** is used to specify further selection criteria for Natural system error messages.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan System Error Messages, Details** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>DBID/FNR</b>		Only applies to the unload function.  The database ID (DBID) and the number of the Adabas (FNR) file where the Natural error messages are stored.
<b>Password/Cipher</b>		Only applies to the unload function.  The password and cipher code for the Adabas file where the Natural error message sources are stored.

Field	PF Key	Explanation
Error number from/to		See <a href="#">Error number</a> in <i>Natural System Error Messages</i> above.
Language codes		See <a href="#">Language codes</a> in <i>Natural Library Object Details</i> .
S/L-Kind		See <a href="#">S/L-Kind</a> in <i>Natural Library Object Details</i> .
Exceptions	PF8	Invokes an extra screen where you can specify exceptions to the selection of Natural system error messages: see <a href="#">Natural System Error Message Exceptions</a> .

## Natural System Error Message Exceptions

The screen **Unload/Load/Scan System Error Messages, Exceptions** is used to specify exceptions to the selection of Natural system error messages.

All Natural system error messages that match the selection criteria specified in [Natural System Error Messages](#) and [Natural System Error Message Details](#) are checked against the specifications made on the screen **Unload/Load/Scan System Error Messages, Exceptions**. Error messages that match *all* specifications defined as exceptions, are exempted from processing.

For explanations of the fields provided on the exceptions screen, see [Natural System Error Message Details](#) above.

For descriptions of keywords and valid input values, see also [select-clause](#) in the section *Direct Commands*.



# 31 Object Specification - Natural Command Processors

---

- Natural Command Processors ..... 150
- Natural Command Processor Source Exceptions ..... 151

This section describes the options provided on the object-specification screens for processing Natural command processor sources.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

## Natural Command Processors

The screen **Unload/Load/Scan Natural Command Processors** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>Library</b>		The name of a Natural command processor library or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>Select library</b>	PF4	Invokes a selection list of Natural command processor libraries available. See also <i>Select Library</i> .
<b>DBID/FNR</b>		Only applies to the unload function.  The database ID (DBID) and the file number of the Adabas (FNR) file where the Natural command processor sources are stored.
<b>Password/Cipher</b>		Only applies to the unload function.  The password and cipher code for the Adabas file where the Natural command processor sources are stored.
<b>Object name</b>		The name of a Natural command processor source or a range of names: see <i>Name</i> .
<b>Select objects</b>		Only applies to the unload function.  If no library range has been specified and this field is selected, a selection list of Natural command processor sources available is displayed (see also <i>Select Objects</i> ).
<b>Exceptions</b>	PF8	Invokes an extra screen where you can specify exceptions to the selection of Natural command processor sources: see <i>Natural Command Processor Source Exceptions</i> .
<b>Settings</b>	PF7	Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings. See <i>Settings</i> .

## Natural Command Processor Source Exceptions

---

The screen **Unload/Load/Scan Natural Command Processors, Exceptions** is used to specify exceptions to the selection of Natural command processor sources.

All objects that match the selection criteria specified in *Natural Command Processor Sources* are checked against the specifications made on the screen **Unload/Load/Scan Natural Command Processors, Exceptions**. Natural command processor sources that match *all* specifications defined as exceptions, are exempted from processing.

For explanations of the fields provided in the exceptions window, see *Natural Command Processor Sources* above.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.



# 32

## Object Specification - Natural-Related Objects

---

▪ Natural Profiles .....	154
▪ Natural Debug Environments .....	155
▪ Natural DL/I Subfiles .....	157

This section describes the options provided on the object-specification screens for processing Natural-related objects. Natural-related objects are profiles, debug environments and DL/I subfiles.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

When you select Natural-related objects on the **Unload/Load Type** screen, the **Select Related Type** screen appears where you can specify the type of the Natural-related object: Natural profiles, debug environments or DL/I subfiles.

## Natural Profiles

The screen **Unload/Load/Scan Natural Profiles** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>DBID/FNR</b>		Only applies to the unload function.  The database ID (DBID) and file number (FNR) of the Adabas file where the Natural profiles are stored.  If no values (or 0) are specified, the current FNAT system file is used.
<b>Password/Cipher</b>		Only applies to the unload function.  The password and cipher code for the Adabas file where the Natural profiles are stored.
<b>Select</b> (DBID/FNR)	PF5	Only applies to the unload function.  Invokes the <b>Select System File</b> window with a list of all system files available in your Natural environment: see also <i>Select System File</i> .
<b>Profile Types</b>		The type(s) of profile to be processed:  parameter, editor, map and/or device.
<b>Object name</b>		The name of a profile or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>Select</b> (Object name)		Only applies to the unload function.  Invokes the <b>List Profiles</b> screen with a selection list of profiles available (see also <i>Select Objects</i> ).
<b>Exceptions</b>	PF8	Invokes an extra screen where you can specify exceptions to the selection of profiles: see <i>Natural Profile Exceptions</i> .
<b>Settings</b>	PF7	Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings: see <i>Settings</i> .

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

- [Natural Profile Exceptions](#)

## Natural Profile Exceptions

The screen **Unload/Load/Scan Natural Profiles, Exceptions** is used to specify exceptions to the selection of Natural profiles.

All objects that match the selection criteria specified in *Natural Profiles* are checked against the specifications made on the screen **Unload/Load/Scan Natural Profiles, Exceptions**. Objects that match *all* specifications defined as exceptions, are exempted from processing.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan Natural Profiles, Exceptions** provides the following fields:

Field	Explanation
<b>Object name</b>	The name of a profile or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>Profile Types</b>	The type(s) of profile to be processed: parameter, editor, map and/or device.

## Natural Debug Environments

The screen **Unload/Load/Scan Natural Debug Environments** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>Library</b>		The name of a library or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>Select library</b>	PF4	Displays a selection list of all libraries available. See also <i>Select Library</i> .
<b>DBID/FNR</b>		Only applies to the unload function.  The database ID (DBID) and file number (FNR) of the Adabas file where the Natural debug environments are stored.  If no values (or 0) are specified, the current FNAT system file is used.

Field	PF Key	Explanation
Password/Cipher		Only applies to the unload function.  The password and cipher code for the Adabas file where the Natural debug environments are stored.
Object name		The name of a debug environment or a range of names: see <i>Name</i> .
Select objects		Only applies to the unload function.  Displays a selection list of debug environments available (see also <i>Select Objects</i> ).
Exceptions	PF8	Invokes an extra screen where you can specify exceptions to the selection of profiles: see <i>Natural Debug Environment Exceptions</i> .
Settings	PF7	Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings: see <i>Settings</i> .

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

- [Natural Debug Environment Exceptions](#)

## Natural Debug Environment Exceptions

The screen **Unload/Load/Scan Debug Environments, Exceptions** is used to specify exceptions to the selection of Natural debug environments.

All objects that match the selection criteria specified in *Natural Debug Environments* are checked against the specifications made on the screen **Unload/Load/Scan Debug Environments, Exceptions**. Objects that match *all* specifications defined as exceptions, are exempted from processing.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan Debug Environments, Exceptions** provides the following fields:

Field	Explanation
Library	The name of a library or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
Select (Library)	Displays a selection list of all libraries available. See also <i>Select Library</i> .
Object name	The name of a debug environment or a range of names: see <i>Name</i> .

## Natural DL/I Subfiles

The screen **Unload/Load/Scan Natural DL/I Subfiles** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>DBID/FNR</b>		Only applies to the unload function.  The database ID (DBID) and file number (FNR) of the Adabas file where the Natural DL/I subfiles are stored.  If no values (or 0) are specified, the current FDIC system file is used.
<b>Password/Cipher</b>		Only applies to the unload function.  The password and cipher code for the Adabas file where the Natural DL/I subfiles are stored.
<b>Select</b> (DBID/FNR)	PF5	Only applies to the unload function.  Invokes the <b>Select System File</b> window with a list of all system files available in your Natural environment: see also <a href="#">Select System File</a> .
<b>Subfile Types</b>		The type(s) of DL/I subfile to be processed: NSB and/or NDB.
<b>Object name</b>		The name of a DL/I subfile or a range of names: see <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .
<b>Select</b> (Object name)		Only applies to the unload function.  Displays a selection list of DL/I subfiles available (see also <a href="#">Select Objects</a> ).
<b>Exceptions</b>	PF8	Invokes an extra screen where you can specify exceptions to the selection of profiles: see <a href="#">Natural DL/I Subfile Exceptions</a> .
<b>Settings</b>	PF7	Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings: see <a href="#">Settings</a> .

- [Natural DL/I Subfile Exceptions](#)

### Natural DL/I Subfile Exceptions

The screen **Unload/Load/Scan Natural DL/I Subfiles, Exceptions** is used to specify exceptions to the selection of Natural DL/I subfiles.

All objects that match the selection criteria specified in [Natural DL/I Subfiles](#) are checked against the specifications made on the screen **Unload/Load/Scan Natural DL/I Subfiles, Exceptions**. Objects that match *all* specifications defined as exceptions, are exempted from processing.

For descriptions of keywords and valid input values, see also [select-clause](#) in the section *Direct Commands*.

The screen **Unload/Load/Scan Natural DL/I Subfiles, Exceptions** provides the following fields:

Field	Explanation
<b>Object name</b>	The name of a DL/I subfile or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>Subfile Types</b>	The type(s) of DL/I subfile to be processed: NSB and/or NDB.

# 33 Object Specification - DDMs

---

- DDMs ..... 160
- DDM Properties ..... 161
- DDM Exceptions ..... 162

This section describes the options provided on the object-specification screens for processing Natural DDMs (data definition modules).

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

## DDMs

The screen **Unload/Load/Scan DDMs** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>FDIC DBID/FNR</b>		Only applies to the unload function.  The database ID (DBID) and file number of the Adabas file where the DDMs are stored.  If no values (or 0) are specified, the current FDIC system file is used.
<b>FDIC Password/Cipher</b>		Only applies to the unload function.  The password and cipher code for the Adabas file where the DDMs are stored.
<b>DDM name</b>		Only applies to the unload function.  The name of a DDM or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>Select objects</b>		Only applies to the unload function.  Displays a selection list of DDMs available (see also <i>Select Objects</i> ).
<b>Properties</b>	PF7	Invokes an extra screen where you can specify additional properties of DDMs: see <i>DDM Properties</i> .
<b>Exceptions</b>	PF8	Invokes an extra screen where you can specify exceptions to the selection of DDMs: see <i>DDM Exceptions</i> .
<b>Settings</b>	PF7	Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings: see <i>Settings</i> .

## DDM Properties

The screen **Unload/Load/Scan DDMs, Properties** is used to specify properties for the DDMs selected for processing.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan DDMs, Properties** provides the following fields:

Field	Explanation
<b>User ID</b>	The ID of the user who saved or cataloged a DDM.  Specify a single user ID or a range of user IDs: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>DDM DBID</b>	The database ID (DBID) of the DDMs.  Valid entries are: 1 to 65535 or 0 (all DBIDs)
<b>DDM FNR</b>	The file number (FNR) of the DDMs:  Valid entries are: 1 to 65535 or 0 (all FNRs).
Object Date: <b>Select all objects (no date check)</b>	Selects all DDMs, regardless of their date.
Object Date: <b>Select objects modified between/and</b>	See <b>Object Date</b> in <i>Natural Library Object Properties</i> .
Object Date: <b>Select objects modified on</b>	See <b>Object Date</b> in <i>Natural Library Object Properties</i> .
Object Size: <b>Select all objects (no size check)</b>	Selects all DDMs, regardless of their size.
Object Size: <b>Select objects with size between/and</b>	Selects all DDMs with a size within the range specified in these fields by entering a start size and/or an end size.
Object Size: <b>Select objects with size</b>	Selects all DDMs with a size that fits the size specified in this field.

## DDM Exceptions

The screen **Unload/Load/Scan DDMs, Exceptions** is used to specify exceptions to the selection of DDMs.

All objects that match the selection criteria specified in *DDMs* and *DDM Properties* are checked against the specifications made on the screen **Unload/Load/Scan DDM, Exceptions**. Objects that match *all* specifications defined as exceptions, are exempted from processing.

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan DDMs, Exceptions** provides the following fields:

Field	Explanation
<b>DDM name</b>	The name of a DDM or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> .
<b>DDM DBID</b>	See <b>DDM DBID</b> in <i>DDM Properties</i> .
<b>DDM FNR</b>	See <b>DDM FNR</b> in <i>DDM Properties</i> .
<b>User ID</b>	See <b>User ID</b> in <i>DDM Properties</i> .
Object Date: <b>Ignore object date</b>	Performs no date check. DDMs are processed, regardless of their date.
Object Date: <b>Exclude objects modified between/and</b>	See <b>Object Date</b> in the section <i>Natural Library Object Exception Properties</i> .
Object Date: <b>Exclude objects modified on</b>	See <b>Object Date</b> in the section <i>Natural Library Object Exception Properties</i> .
Object Size: <b>Ignore object size</b>	Performs no size check. DDMs are processed, regardless of their size.
Object Size: <b>Exclude objects with size between/and</b>	Exempts from processing all DDMs with a size within the range specified in these fields by entering a start size and/or an end size.
Object Size: <b>Exclude objects with size</b>	Exempts from processing all DDMs with a size that fits the size specified in this field.

# 34 Object Specification - FDTs

---

This section describes the options provided on the object-specification screen for processing Adabas FDTs (Field Definition Tables).

For descriptions of keywords and valid input values, see also *select-clause* in the section *Direct Commands*.

The screen **Unload/Load/Scan FDTs** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>DBID</b>		The database ID where the FDT is located.  Load and scan: A valid DBID or 0 for all DBIDs.
<b>FNR</b>		The file number where the FDT is located.  Load and scan: A valid FNR or 0 for all FDTs.
<b>Password/Cipher</b>		Only applies to the unload and load functions.  The Adabas password and the cipher code of the Adabas file where the FDT is located.
<b>Settings</b>	PF7	Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings. See <i>Settings</i> .

---

# 35 Use Selection or List Workplan

---

This option is used to specify a Workplan of the type SELECTION or LIST. These Workplans specify selection criteria for the objects to be processed. See also the section [Workplans](#).

The screen **Unload/Load/Scan Selection or List** provides the following fields and PF keys:

Field	PF Key	Explanation
<b>Name</b>	PF4	The name of the Workplan to be processed.
<b>List Workplan</b>		Displays the contents of the Workplan specified in the <b>Name</b> field.
<b>Select Workplan</b>	PF5	Displays a list of all Workplans available. See also <a href="#">List the Available Workplans in the Workplan Library</a> in <i>Administration</i> .
<b>Settings</b>	PF7	Invokes the <b>Unload/Load/Scan Settings</b> screen where you can specify option and parameter settings. See <a href="#">Settings</a> .



# 36 Settings

---

- Settings Screen Fields ..... 168
- Set Additional Options ..... 170
- Set Global Parameters ..... 176

The settings option is used to specify option settings for the unload, load, find or scan function and parameter settings for the unload or load function.

► **To invoke the Unload/Load/Scan Settings screen**

- On any of the unload, load or scan screens, enter the following internal command:

SETTINGS

See also *Commands for Navigation and Special Functions* in the section *Direct Commands*.

Or:

Activate advanced-user mode, choose a function and choose ENTER to start the processing procedure.

Or:

On advanced-user screens, choose PF7 (Setti).

Unless selected by default, to activate the options provided on the **Unload/Load/Scan Settings** screen described below, mark the corresponding input field with any single character.

## Settings Screen Fields

---

The **Unload/Load/Scan Settings** screen provides the following fields and PF keys:

Field	PF Key	Explanation
Transfer format		<p>Only valid if <b>Use default options</b> (this is the default) has been selected.</p> <p>If selected, the data to be processed is written/read in Transfer format to/from the work file. See also <i>Work File Format</i> in <i>Work Files</i>.</p> <p>Unload function: The data to be unloaded is written in Transfer format to the work file. Note that if you want to change the setting of this field for a subsequent unload, you need to return to the <b>Main Menu</b> or enter the command GO UNLOAD END (see <i>Commands for Navigation and Special Functions</i> in <i>Direct Commands</i>) and restart the unload function.</p> <p>Load and scan functions: The data to be loaded or scanned are expected to be in Transfer format.</p>

Field	PF Key	Explanation
<b>Unicode work file</b>		<p>Only applies to the unload function and if <b>Transfer format</b> has been selected.</p> <p>If this option is selected, all object sources are converted to Unicode/UTF-8 (Universal Transformation Format, 8-bit form) before they are written to the work file.</p> <p>If a Unicode work file is specified, you cannot use the transfer options <b>Use conversion table</b>, <b>Substitute line references</b> and <b>Incorporate free rules</b>.</p>
<b>Use PC File</b>		<p>Only applies if Entire Connection is installed.</p> <p>Writes data to an Entire Connection work file.</p>
<b>PC File</b>		<p>Only applies if Entire Connection is installed.</p> <p>The complete path name to the Entire Connection work file. If your system environment does not accept a backslash (\) separator, use a slash (/) instead.</p>
<b>Use default options</b>		<p>Default options are used (this is the default). See also <i>Profile Settings</i> and <i>Set Additional Options</i>.</p>
<b>Set additional options</b>	PF4	<p>Only valid if <b>Use default options</b> has been selected.</p> <p>Invokes the <b>Options</b> screen where you can modify the default settings and enter additional options for the processing sequence. For the options available, see <i>Set Additional Options</i>.</p>
<b>Use Option Workplan</b>		<p>A Workplan of the type OPTION is used. See also <i>Workplans</i>.</p>
<b>Name</b> (next to Use Option Workplan)		<p>Only valid if <b>Use Option Workplan</b> has been selected.</p> <p>The name of a Workplan of the type OPTION to be used.</p>
<b>List Option Workplan</b>	PF6	<p>Only valid if <b>Use Option Workplan</b> has been selected.</p> <p>Displays the contents of the Workplan specified in the <b>Name</b> field next to <b>Use Option Workplan</b>.</p>
<b>Select Option Workplan</b>	PF5	<p>Only valid if <b>Use Option Workplan</b> has been selected.</p> <p>Displays a selection list of available Workplans of the type OPTION (see also <i>List the Available Workplans in the Workplan Library in Administration</i>).</p>
<b>Do not use parameters</b>		<p>If selected (default setting), no parameters are set.</p>
<b>Use global parameters</b>		<p>Global parameters are used. See <i>Set Global Parameters</i>.</p>

Field	PF Key	Explanation
<b>Set global parameters</b>	PF7	Only valid if <b>Use global parameters</b> has been selected.  Invokes the <b>Parameters</b> screen. See <a href="#">Set Global Parameters</a> and <a href="#">parameter-setting (Direct Commands)</a> for descriptions of keywords and valid input values.
<b>Use Parameter Workplan</b>		A Workplan of the type PARAMETER is used. See also <a href="#">Workplans</a> .
<b>Name</b>  (next to Use Parameter Workplan)		Only valid if <b>Use Parameter Workplan</b> has been selected.  The name of a Workplan of the type PARAMETER to be used.
<b>List Parameter Workplan</b>	PF9	Only valid if <b>Use Parameter Workplan</b> has been selected.  Displays the contents of the Workplan specified in the <b>Name</b> field next to <b>Use Parameter Workplan</b> .
<b>Select Parameter Workplan</b>		Only valid if <b>Use Parameter Workplan</b> has been selected.  Displays a selection list of available Workplans of the type PARAMETER. See also <a href="#">List the Available Workplans in the Workplan Library in Administration</a> .

## Set Additional Options

The sections contained in the **Options** screen are described below. Note that not all of the sections may appear on the screen, because they depend on the function used, the settings defined and the products installed.

For descriptions of keywords and valid input values, see also [option-setting](#) in the section *Direct Commands*.

This section covers the following topics:

- [Work File and Report Options](#)
- [XREF Options](#)
- [Version Check](#)
- [Transfer Options](#)
- [Replace Options](#)
- [Number to Process](#)
- [FDIC Settings](#)

- FSEC Settings

## Work File and Report Options

The options provided for work files and reports are described in the following section.

Field	Explanation
<b>Use PC File</b>	Only applies if Entire Connection is installed.  Writes data to an Entire Connection work file.
<b>PC File</b>	Only applies if Entire Connection is installed.  The complete path name assigned to the Entire Connection work file. If your system environment does not accept a backslash (\) separator, use a slash (/) instead.
<b>Write report</b>	Writes a report of the objects processed to the report text member specified in the <b>Report text member</b> field.  The <b>Write report</b> option is selected by default.  To display the report, enter the internal command SHOW REPORT FILE (see <i>Commands for Navigation and Special Functions</i> in <i>Direct Commands</i> ).
<b>Start new report</b>	Only valid if <b>Write report</b> has been selected.  Deletes the contents of the report text member before a new report is written. Otherwise, a new report is appended to the existing one.
<b>Report text member</b>	Only valid if <b>Write report</b> has been selected.  The name of the text member stored in the Workplan library to which the report is written.
<b>Select text member</b>	Displays a list of all text members stored in the Workplan library. From this list, you can select a <b>Report text member</b> .
<b>Write restart information</b>	Only applies to the load function.  When this option is set, restart information is provided for the <b>restart load</b> function.  For details, see <i>Restart Load</i> in <i>Functions</i> .
<b>Restart text member</b>	Only applies to the load function and if <b>Write restart information</b> has been selected.  The name of the text member in the Workplan library to which the restart information is written.  If you do not specify a name, the Object Handler generates a name and assigns it to the text member.

Field	Explanation
<b>Select text member</b>	Displays a list of all text members stored in the Workplan library. From this list, you can select a <b>Restart text member</b> .

## XREF Options

XREF options are only available when unloading or loading data in internal format, that is, if the field **Transfer format** has *not* been selected. Predict must be installed to process XRef data.

The XREF options provided and the functions to which they apply are described in the following section.

Field	Explanation	Function
<b>Yes (unload XRef data)</b> or <b>Yes (load XRef data)</b>	Unloads cataloged objects and their cross-reference data, if any. Loads cataloged objects and their cross-reference data if cross-references exist in the work file.	Unload Load
<b>No (ignore XRef data)</b>	No XRef data is processed.	Unload Load
<b>Force</b>	Loads cataloged objects and their cross-reference data only if cross-references exist in the work file and if Predict entries exist for the objects in the FDIC system file.	Load
<b>Doc</b>	Loads cataloged objects only if Predict entries exist for the objects in the FDIC system file.	Load
<b>Special</b>	Loads cataloged objects and their cross-reference data (if any).	Load

## Version Check

The **Version check** option is only available when loading data in internal format, that is, if the field **Transfer format** has *not* been selected.

If **Version check** is selected, the Natural version under which the objects were cataloged and written to the work file is compared with the current Natural version. Objects cataloged under a Natural version higher than the current one will be rejected.

## Transfer Options

Transfer options are only available when processing data in Transfer format, that is, if the field **Transfer format** has been selected.

The transfer options provided and the functions to which they apply are described in the following section.

Option	Explanation	Function
<b>Substitute line references</b>	Only applies if source-code line numbers are used for statement references.  If line numbers are used as references in the source code, the line numbers of referenced lines and the line number references are replaced with labels. The sources are not modified in the database.	Unload
<b>Include line numbers</b>	If you choose this option, the line numbers will be transferred. (By default, line numbers in Natural objects are not transferred.)	Unload
<b>Incorporate free rules</b>	If Predict is installed, Predict rules associated with a map are incorporated into the map source.	Unload
<b>Use conversion table</b>	<b>Caution:</b> Use this option only in special cases, such as when performing a non-FTP transfer between environments with different character sets, where no conversion is performed by the transfer tool.  Unload: Converts data to ASCII format by using the internal Natural conversion table ( <b>System table</b> ) or a conversion table defined by the user ( <b>User table</b> ).  Load: Converts data to EBCDIC format by using the internal Natural conversion table ( <b>System table</b> ) or a conversion table defined by the user ( <b>User table</b> ). Note that this only applies if the data in the work file is in ASCII format or if a conversion program is specified (see <b>User table</b> ).	Unload Load
<b>System table</b>	Only valid if <b>Use conversion table</b> has been selected.  Unload: Converts data to ASCII format by using the internal Natural conversion table.  Load: Converts data to EBCDIC format by using the internal Natural conversion table.	Unload Load
<b>User table</b>	Only valid if <b>Use conversion table</b> has been selected.  If the name of a conversion program has been entered in the field, data is converted to EBCDIC or ASCII format by using the conversion program defined. To specify an individual conversion program, the program must be located in the library SYSOBJH or one of its steplibs. See the example subprograms OTNCONAE and OTNCONEA in the library SYSOBJH.	Unload Load

Option	Explanation	Function										
	If no conversion program is specified, by default, the corresponding conversion table in the Natural file <i>NATCONV.INI</i> is used for the unload ([ISO8859_1->EBCDIC] ) and the load ([EBCDIC->ISO8859_1]) functions.											
<b>Use load code page</b>	<p>If you choose this option, a window appears where you can enter the name of the code page to be used for the load function.</p> <p>If this option is selected, all object sources unloaded into a work file in UTF-8 will be converted with the specified code page when they are loaded into a work file. See also <a href="#">Unicode work file</a>.</p> <p>If you enter *CODEPAGE as the code page name, the value assigned to the system variable *CODEPAGE is used (see the <i>System Variables</i> documentation).</p> <p>If no code page name is specified, the source objects are converted with the code page used when unloading them.</p> <p>If <b>Use load code page</b> is specified, you cannot use the options <b>Use conversion table</b> and <b>Translate to upper case</b>.</p>	Load										
<b>Translate to upper case</b>	Translates any source code to be loaded into upper case.	Load										
<b>Data area format</b>	<p>Only applies to data areas.</p> <table border="1" data-bbox="358 978 1279 1570"> <thead> <tr> <th data-bbox="358 978 623 1024"></th> <th data-bbox="628 978 1279 1024"></th> </tr> </thead> <tbody> <tr> <td data-bbox="358 1031 623 1108">Specifies the format in which to unload or load data area sources. Possible input values are:</td> <td data-bbox="628 1031 1279 1108"></td> </tr> <tr> <td data-bbox="358 1115 623 1192">N</td> <td data-bbox="628 1115 1279 1192">Converts data areas to the new internal data area format.</td> </tr> <tr> <td data-bbox="358 1199 623 1486">0</td> <td data-bbox="628 1199 1279 1486">Converts data areas to the old internal data area format. If one or more data area sources cannot be converted to the old internal data area format, the Object Handler issues a corresponding message when unloading is complete. In addition, in the <b>Status</b> column of the unload report generated by the unload function, a corresponding remark appears next to the names of the data area sources affected.</td> </tr> <tr> <td data-bbox="358 1493 623 1570">*</td> <td data-bbox="628 1493 1279 1570">Does not convert data areas. This is the default.</td> </tr> </tbody> </table> <p>For details, see <i>Data Area Editor</i> in the <i>Editors</i> documentation.</p>			Specifies the format in which to unload or load data area sources. Possible input values are:		N	Converts data areas to the new internal data area format.	0	Converts data areas to the old internal data area format. If one or more data area sources cannot be converted to the old internal data area format, the Object Handler issues a corresponding message when unloading is complete. In addition, in the <b>Status</b> column of the unload report generated by the unload function, a corresponding remark appears next to the names of the data area sources affected.	*	Does not convert data areas. This is the default.	Unload Load
Specifies the format in which to unload or load data area sources. Possible input values are:												
N	Converts data areas to the new internal data area format.											
0	Converts data areas to the old internal data area format. If one or more data area sources cannot be converted to the old internal data area format, the Object Handler issues a corresponding message when unloading is complete. In addition, in the <b>Status</b> column of the unload report generated by the unload function, a corresponding remark appears next to the names of the data area sources affected.											
*	Does not convert data areas. This is the default.											

## Replace Options

The replace options described below only apply to the load function:

<b>Do not replace</b>	Does not replace any objects. This is the default.
<b>Replace all</b>	Replaces all objects.
<b>Replace obsolete</b>	Replaces objects with a date older than the date of the objects in the load file.
<b>Replace except newer</b>	Replaces all objects except those with a date newer than the date of the objects in the load file.

## Number to Process

Number to process only applies to the load and scan functions.

In the field **Number to process**, enter a value with a maximum of 5 digits. If a value greater than 0 is specified, the load or scan function stops after the specified number of objects has been processed.



**Note:** If a cataloged Natural object is processed directly after the source object of the same name, they are considered one object.

## FDIC Settings

FDIC settings are used to specify the Predict file (FDIC) to be used for processing XRef data (only applies if Predict is installed) or load DDMs:

<b>DBID</b>	The database ID where the FDIC file is located.
<b>FNR</b>	The file number where the FDIC file is located.
<b>Password</b>	Optional. The Adabas password of the Adabas file where the FDIC file is located.
<b>Cipher</b>	Optional. The cipher code of the Adabas file where the FDIC file is located.

## FSEC Settings

FSEC settings only apply if Natural Security is installed.

FSEC settings are used to specify the Natural Security data file (FSEC) to be used for security checks:

<b>DBID</b>	The database ID where the FSEC file is located.
<b>FNR</b>	The file number where the FSEC file is located.
<b>Password</b>	Optional. The Adabas password of the Adabas file where the FSEC file is located.
<b>Cipher</b>	Optional. The cipher code of the Adabas file where the FSEC file is located.

## Set Global Parameters

---

**Only applies to the load or unload function.**

The fields provided on the **Parameters** screen can be used to change global parameter settings for the objects to be processed with the load or unload function, and to change the target environment for the load function. For example, you can specify new names (or name ranges) under which the selected objects are unloaded to the work file, or you can specify a different library into which the selected objects are loaded from the work file.

If global parameters are specified during the unload function, the parameter settings affect the objects before they are written to the work file. If they are specified during the load function, the parameter settings affect the objects before they are written to the target environment.

The values that can be specified to change parameter settings, are entered next to the required parameters in the fields **Check Value** and **New Value**.

If no value has been entered in **Check Value**, the value entered in **New Value** affects all objects to which the specific parameter setting applies. If a value has been entered in **Check Value**, the value entered in **New Value** only affects objects to which the specific parameter setting and the value entered in **Check Value** apply. If a **Check Value** or **New Value** is not relevant to the type of object to be processed, any value entered in either field will be ignored. For example: Natural system error messages have no library name. Therefore, when processing Natural system error messages, a value entered in **Check Value** or **New Value** for the **Library** field will be ignored.

**Check Value** and **New Value** do not apply to the parameter **Error number difference** and the parameters contained in the section **System files for load** of the **Parameters** screen.

For valid parameter settings, see also [parameter-setting](#) in the section *Direct Commands*.

The following fields are contained in the **Parameters** screen:

Field/Section	Explanation
<b>Object name</b>	<p><b>Check Value/New Value:</b></p> <p>A single object name or a range of names: see <i>Name</i> in <i>Name, Date and Time Specification</i> and <i>Rules for New Values</i>.</p> <p><b>Note:</b> Not applicable to DDMs on mainframe platforms.</p>
<b>Library</b>	<p><b>Check Value/New Value:</b></p> <p>A single library name or a range of names: see <i>Name</i> and <i>Rules for New Values</i>.</p>
<b>Date</b>	<p><b>Check Value/New Value:</b></p> <p>A single date or a range of dates: see <i>Date</i> in <i>Name, Date and Time Specification</i> and <i>Rules for New Values</i>.</p>
<b>Time</b>	<p><b>Check Value/New Value:</b></p> <p>A time or a range of times: see <i>Time</i> in <i>Name, Date and Time Specification</i> and <i>Rules for New Values</i>.</p>
<b>User ID</b>	<p><b>Check Value/New Value:</b></p> <p>A single user ID or a range of user IDs: see <i>Name</i> and <i>Rules for New Values</i>.</p>
<b>Terminal ID</b>	<p><b>Check Value/New Value:</b></p> <p>A single terminal ID or a range of terminal IDs: see <i>Name</i> and <i>Rules for New Values</i>.</p>
<b>Lang. codes</b>	<p>Only applies when processing Natural system error messages or user-defined error messages.</p> <p><b>Check Value/New Value:</b></p> <p>Up to 8 valid language codes such as code 4 for Spanish. If more than one language code is specified, <b>Check Value</b> must contain the same number of language codes. In this case, the language code in <b>Check Value</b> is replaced by the language code in the corresponding <b>New Value</b>.</p> <p>Note: <b>New Value</b> does not apply to the long texts of Natural system error messages for which English (code 1) is the only valid language.</p>
<b>Error number difference</b>	<p>Only applies when processing Natural system error messages or user-defined error messages.</p> <p>A 4-digit positive or negative value (+/-nnnn) to be used as a new number range for error messages. Start and end values must be provided in the <b>Error number from/to</b> fields (see <i>Natural Library Objects</i>) to validate whether the new range can be applied to the selected error messages.</p> <p>Example:</p>

Field/Section	Explanation
	If <b>Error number from/to</b> selects message numbers 1 to 10 and <b>Error number difference</b> is set to 2000, the messages will be renumbered from 2001 to 2010. A value of -1000 in <b>Error number difference</b> would cause a validation error.
<b>FDT DBID/FNR</b>	<b>Check Value/New Value:</b> A valid database ID (DBID) and/or file number (FNR) for Adabas FDTs.
System files for load: Load FNAT  <b>DBID</b> <b>FNR</b> <b>Password</b> <b>Cipher</b>	Only applies to the load function.  The database ID ( <b>DBID</b> ) and file number ( <b>FNR</b> ) of the target FNAT system file. This system file is used for all library objects whose library name starts with <i>SYS</i> , but not <i>SYSTEM</i> . If required, enter the Adabas password ( <b>Password</b> ) and the cipher code ( <b>Cipher</b> ) for the system file.
System files for load: Load FUSER  <b>DBID</b> <b>FNR</b> <b>Password</b> <b>Cipher</b>	Only applies to the load function.  The database ID ( <b>DBID</b> ) and file number ( <b>FNR</b> ) of the target FUSER system file. This system file is used for all library objects whose library name does not start with <i>SYS</i> , and for the library <i>SYSTEM</i> . If required, enter the Adabas password ( <b>Password</b> ) and the cipher code ( <b>Cipher</b> ) for the system file.
System files for load: Load FNAT/FUSER  <b>Select</b>	Only applies to the load function.  Invokes the <b>Select System File</b> window with a list of all system files available in your Natural environment: see <a href="#">Select System File</a> .
System files for load: Load NCP  <b>DBID</b> <b>FNR</b> <b>Password</b> <b>Cipher</b>	Only applies to the load function.  The database ID ( <b>DBID</b> ) and file number ( <b>FNR</b> ) of the target Adabas file into which the Natural command processor sources are to be loaded. If required, enter the Adabas password ( <b>Password</b> ) and the cipher code ( <b>Cipher</b> ) for the system file.

This section covers the following topic:

---

- Rules for New Values

### Rules for New Values

The following applies to **New Value** for **Object name**, **Library**, **Date/Time**, **User ID** and **Terminal ID**.

If **New Value** contains a range with an asterisk (\*) such as ABC\*, the number of characters before the asterisk (\*) determines the number of characters to be replaced in **Check Value**. This is also valid if **Check Value** is shorter than the range specified in **New Value** (see the second example in *Examples* below).

#### Examples:

1. If **Object name** is ABCDEFG and **New Value** is set to ZYX\*, the resulting object name is ZYXDEFG.
2. If **Object name** is AB and **New Value** is set to ZYX\*, the resulting object name is ZYX.
3. If **Date/Time** is 2005-03-26 and **New Value** is set to 2006\*, the resulting object date is 2006-03-26.



# 37 Workplans

---

- Creating, Selecting and Modifying Workplans ..... 182
- Contents of Workplans ..... 182
- Examples of Workplans ..... 183
- Referencing Workplans ..... 184

Workplans define individual standard procedures for command execution, object selection and parameter or option settings which can be used to further automate function processing.

Workplans are Natural objects of the type Text. They are, by default, stored in the library WORKPLAN located in the current FUSER system file.

## Creating, Selecting and Modifying Workplans

You can use the **administration** function (see the relevant section) to create a Workplan, select a Workplan from a list, modify a Workplan, or change the default library for Workplans. The default library can also be changed by specifying the `Workplan-Library` parameter in your Object Handler profile (see *Profile Settings*).

## Contents of Workplans

A Workplan consists of a header (generated by the Object Handler) and an associated instructional or textual part. Instructional parts contain Object Handler commands and parameter and/or option settings. Textual parts contain plain text only. Header and instructional or textual parts can contain comments (for example, the short description of the Workplan) that must start with the delimiter characters `/*` and are restricted to one line.

There are six types of Workplan: PROCEDURE, SELECTION, LIST, PARAMETER, OPTION and TEXT.

The table below lists the valid headers (to be entered if creating a Workplan outside the Object Handler) for the corresponding types of Workplan and describes the contents of the instructional or textual part. Additionally, it provides cross references to the clauses that apply when specifying Object Handler direct commands. The Object Handler direct commands provided are explained in the section *Direct Commands*.

Valid Headers	Contents	Related Topic in <i>Direct Commands</i>
TYPE PROCEDURE	An Object Handler command procedure.  This Workplan can contain any combination of Object Handler commands available for PROCEDURE. Enter a sequence of commands separated by semicolons (;).	<i>Basic Command Syntax</i>
TYPE SELECTION	Selection criteria for objects.  This Workplan can be used in Object Handler Workplan commands.	<i>select-clause</i>

Valid Headers	Contents	Related Topic in <i>Direct Commands</i>
TYPE LIST	A list of objects.  This Workplan can be used in Object Handler Workplan commands.	<i>select-clause</i>  <i>Object List - LIST Workplan</i>
TYPE PARAMETER	Parameters for the unload or load function.  This Workplan can be used to change attributes for the objects to be processed such as the name of a new target library where objects are loaded.  TYPE PARAMETER can be used in Object Handler Workplan commands.	<i>parameter-setting</i>
TYPE OPTION	Options for the unload or load function, for example, report settings.  This Workplan can be used in Object Handler Workplan commands.	<i>option-setting</i>
TYPE TEXT	Comments or any other text that can be used for documentation purpose.	Not applicable

## Examples of Workplans

The following table lists examples of instructional parts contained in a Workplan.

Workplan Type	Instruction	Explanation
PROCEDURE	FINDLIB * LIB TEST	Check whether the library TEST exists.
PROCEDURE	UNLOAD A* LIB TEST	Unload from the library TEST into Work File 1 all Natural programming objects starting with A, and all user-defined error messages.
SELECTION	* LIB TEST	Process all objects from the library TEST.
TEXT	This is a Workplan comment.	Any text.

This section covers the following topic:

- [Example of Workplan Contents](#)

## Example of Workplan Contents

The following is an example listing of a PROCEDURE Workplan where the UNLOAD command is executed:

```
TYPE PROCEDURE /* VERSION=03.01 NATURAL VERSION=04.02.01 PL=0 AUTHOR=SAG
DATE=2006-07-20 09:40:12
/* unload from library TEST with target library PROD01
UNLOAD * LIB TEST OBJTYPE N
WITH NEWLIBRARY PROD01
WHERE REPORT MYREP01
```

## Referencing Workplans

---

You can reference a Workplan by using Object Handler menu functions or direct commands (see also the section [Direct Commands](#)).

The following syntax applies when referencing a Workplan with the Object Handler direct commands described in the section [Direct Commands](#).

```
( workplan-name
  [ LIBRARY library-name ]
  [ DBID dbid [ FNR fnr ] ] [ NAME vsam-name ]
  [ CIPHER cipher ]
  [
    {
      PASSWORD
      PSW
    } password
  ]
)
```

The syntactical options are explained in the following section:

- [Keyword Explanation](#)

## Keyword Explanation

The table below describes the keywords and values that apply to the syntax for referencing Workplans.

Keyword	Values	Default Value
<i>workplan-name</i>	The name of the Natural text member in the Workplan library to be used as the Workplan.	No default
LIBRARY	The name of the library where the Workplan is located.	WORKPLAN
DBID	The ID of the Adabas database where the Workplan library is located.	0 (current FNAT/FUSER)
FNR	The number of the Adabas file where the Workplan library is located.	0 (current FNAT/FUSER)
NAME	Only applies to objects on mainframes.  The name of a valid VSAM file where the Workplan library is located.	blank (current FNAT/FUSER)
CIPHER	Only applies to objects on mainframes.  An 8-digit cipher code.	blank (current FNAT/FUSER)
PASSWORD	Only applies to objects on mainframes.  An 8-character Adabas password.	blank (current FNAT/FUSER)



# 38

## Name, Date and Time Specification

---

▪ Name .....	188
▪ Date .....	189
▪ Time .....	190

You can use a name, a date, a time or a range of names, dates and times to select Natural library objects, Natural command processor sources, Natural-related objects or Natural DDMs (data definition modules).

## Name

You can specify a name or a range of names.

In the list of options below, *value* is any combination of one or more characters:

	Input	Items Selected
	<i>value</i>	All items with names equal to <i>value</i> .
	*	All items.
	>	
	?	All items with any single character for each question mark (?) entered.
Leading Characters	<i>value</i> *	All items with names that start with <i>value</i> .  Example: AB* Selected: AB, AB1, ABC, ABEZ Not selected: AA1, ACB
Wildcard	<i>value</i> ?	All items with names that start with <i>value</i> and end with any single character for each question mark (?) entered.  Example: ABC? Selected: ABCA, ABCZ Not selected: AXC, ABCAA
	<i>value</i> ? <i>value</i> ? <i>value</i> * <i>value</i> ? * <i>value</i> ? <i>value</i> *	All items that match <i>value</i> combined with asterisk (*) and question mark (?) in any order.  Example: A?C*Z Selected: ABCZ, AXCBBBZ, AN CZ Not selected: ACBZ, ABDEZ, AXCBBBZA
Start Value	<i>value</i> >	All items with names greater than or equal to <i>value</i> .  Example: AB> Selected: AB, AB1, BBB, ZZZZZZZ Not selected: AA1, AAB
End Value	<i>value</i> <	All items with names less than or equal to <i>value</i> .  Example: AX< Selected: AB, AWW, AX Not selected: AXA, AY



**Note:** The parameter specification option **New Value** only allows leading characters (asterisk (\*) notation). See [Rules for New Values](#) in *Set Global Parameters* in the section *Settings*.

## Date

All date values within the Object Handler are specified in international date format.

You can specify a date, a range of dates, a special date or a range of special dates. A date must be specified in the format *YYYY-MM-DD* (*YYYY* = year, *MM* = month, *DD* = day).

In the list of options below, the underlined portion of a keyword represents its valid abbreviation, and *value* is any combination of one or more digits:

	Input Value	Items Selected
Date	<i>YYYY-MM-DD</i>	All items with a date equal to <i>YYYY-MM-DD</i> .  Example: 2003-02-15
Leading characters	<i>value*</i>	All items with a date that starts with <i>value</i> .  Example: 2002* Selected: 2002-01-01, 2002-12-31 Not selected: 2001-12-31, 2003-01-01
Start value	<i>value&gt;</i>	All items with a date greater than <i>value</i> .  Example: 2002-05> Selected: 2002-05-01, 2002-12-31, 2003-01-01, 2003-12-31 Not selected: 2002-04-31, 2001-12-31  <b>Special dates</b> can be used as <i>value</i> (see below).
End value	<i>value&lt;</i>	All items with a date less than <i>value</i> .  Example: 2003-02< Selected: 2002-05-01, 2002-12-31, 2003-01-01, 2003-01-31 Not selected: 2003-02-01, 2003-05-18  <b>Special dates</b> can be used as <i>value</i> (see below).
<b>Special Dates</b>		
TODAY (+/- nnnn)		All items with the date of the current day.  The day can be followed by <i>+nnnn</i> or <i>-nnnn</i> where <i>nnnn</i> has a maximum of 4 digits.  The resulting date is computed as the date of the current day plus or minus <i>nnnn</i> days.  Example: If the current date is 2003-03-01, TODAY +5 results in 2003-03-06.

	Input Value	Items Selected
YESTERDAY		All items with the date of the day before the current day.
MONTH		All items with the date range of the current month.  Example: The current month is 2003-02. Selected: 2003-02-01, 2003-02-30 Not selected: 2003-03-01
		FMDATE: Starts with the first day of the current month.  TODATE: Ends with the last day of the current month.  If the values of FMDATE and TODATE are identical, the selection is restricted to one day.
YEAR		All items with the date range of the current year.  Example: The current year is 2003. Selected: 2003-01-01, 2002-12-31 Not selected: 2002-31-12
		FMDATE: Starts with the first day of the current year.  TODATE: Ends with the last day of the current year.  If the values of FMDATE and TODATE are identical, the selection is restricted to one year.



**Note:** The parameter specification option **New Value** only allows leading characters (asterisk (\*) notation). See [Rules for New Values](#) in *Set Global Parameters* in the section *Settings*.

## Time

You can specify a time or a range of times. The time must be specified in the format *HH:II:SS* (*HH* = hours, *II* = minutes, *SS* = seconds).

In the list of options below, *value* is any combination of one or more digits:

	Input Value	Items Selected
Time	<i>HH:II:SS</i>	All items with a time equal to <i>HH:II:SS</i> .  Example: 14:15:16
Leading characters	<i>value</i> *	All items with a time that starts with <i>value</i> .  Example: 13:* Selected: 13:00:00, 13:10:53, 13:59:59 Not selected: 12:59:59, 14:00:00

# 39 Work Files

---

■ Work File Assignment .....	192
■ Work File Format .....	192

This section describes work files and valid formats that apply to the unload, load and scan functions of the Object Handler.

See also [Work File Options](#) in the section *Settings*.

## Work File Assignment

---

The following table lists the work files used by the Object Handler.

File	Explanation
Work File 1	Used for the unload, load and scan functions.  Contains the data unloaded.
Work File 7	Only used if Entire Connection is installed and if <b>Use PC File</b> is selected on the <b>Options</b> screen (see also <a href="#">Set Additional Options</a> ).  Work File 7 must be defined as Entire Connection work file to be used for the unload, load and scan functions.  Contains the data unloaded.
Work File 8	Only used if Entire Connection is installed and if <b>Use PC File</b> is selected on the <b>Options</b> screen (see also <a href="#">Set Additional Options</a> ).  Work File 8 must be defined as Entire Connection work file to be used for the unload, load and scan functions.  Used as internal file for processing Entire Connection commands.  <b>Note:</b>  The number of the work file can be changed with user exit routine <b>OBJHEX03</b> (see <i>Batch Condition Codes and User Exit Routines</i> ) or with the options PCCOMMANDFILENUMBER, PCCOM and PCCFN (see <a href="#">option-setting</a> ) when using a Workplan of the type PROCEDURE.

## Work File Format

---

There are two file formats for unloading objects in the source environment into work files and for loading them from work files into the target environment: an internal format and the Transfer format. Work files must be of internal format to transfer binary data. Work files must be of Transfer format to transfer text data.

This section covers the following topics:

- [Internal Format](#)
- [Transfer Format](#)

## Internal Format

The internal format is an internal record layout for work files that are used to transfer Natural sources and cataloged objects, error messages, Natural command processor sources, Adabas FDTs (Field Definition Tables), Natural DDMs (data definition modules) and Natural-related objects from one environment to another.

Use work files of internal format to transfer objects between identical platforms.

With the internal format activated, Natural objects are read from the source environment and written to a Natural work file by using the unload function of the Object Handler. This work file can be transported to another environment with standard file transfer services. In the target environment, the objects can then be read from the work file and loaded into the local file or database system with the load function of the Object Handler.

We strongly recommend that you define Work File 1 with `RECFM=VB` (variable blocked) because the Object Handler writes records of variable length to the work file, which reduces the work file size.

If you define Work File 1 with `RECFM=FB` (fixed blocked), you must specify the option `FIXEDLENGTH` for the unload function. Otherwise, the records written to the work file are filled up with binary zeros.



**Note:** Work files created by the utility NATUNLD on the server, must be processed in internal format. The work files must be created on a server of the same platform where NATUNLD was applied.

## Transfer Format

See also [Transfer format](#) in the section *Settings*.

The Transfer format is a general record layout for work files that contain load or unload data. This format is platform-independent and can be used to transfer the sources of Natural objects, Natural command processor sources, error messages, DDMs and Adabas FDTs from one hardware platform to another and between UNIX, OpenVMS, mainframe and Windows platforms.

With the option **Transfer format** set, the unload function of the Object Handler reads Natural objects from a hardware platform and then restructures them.

Formatted records are written to a Natural work file that can be transported to another platform with standard file transfer services. On the target platform, the load function of the Object Handler then reads the objects from the work file and loads them into the local file or database system. The

objects read from the work file are restructured according to the structure of the new hardware platform.

### **Handling Sources in Unicode/UTF-8**

Transfer format is also used to unload or load sources of Natural objects in Unicode/UTF-8 (Universal Transformation Format, 8-bit form). If you specify the corresponding unload option (`WORKFILETYPE` set to `UTF-8` in command mode or **Unicode work file** in menu mode), all object sources will be unloaded into a work file in UTF-8. If you specify the corresponding load option (`LOAD-CODE-PAGE` in command mode or **Use load code page** in menu mode), all object sources in UTF-8 will be converted with the specified code page when they are loaded into a Natural system file.

### **Work Files from SYSTRANS**

Use Transfer format to process work files created by the utility SYSTRANS. Work files that contain object sources encoded in UTF-8 cannot be processed with SYSTRANS.

# 40 Direct Commands

---

The Object Handler provides direct commands for the following purposes:

- To execute an Object Handler function such as unloading or loading objects in batch mode or in direct command online mode without using Object Handler menus (see also *Batch or Direct Command Calls*).
- To execute or reference a Workplan (see also the section *Workplans*).
- To be used as an instruction in a Workplan.
- To navigate through screens.
- To perform special functions.

This section describes the basic command syntax and the individual clauses, parameter and option settings available to perform these tasks. In addition, you can view examples that illustrate the use of direct commands.

The symbols used in the syntax diagrams shown in this section are explained in *System Command Syntax* in the *System Commands* documentation.

This section covers the following topics:

- **Basic Command Syntax**
- *select-clause*
- **Object List - LIST Workplan**
- *parameter-setting*
- *option-setting*
- **Examples of Using Direct Commands**

 **Commands for Navigation and Special Functions**

# 41 Basic Command Syntax

---

This section describes the Object Handler direct commands provided for executing Object Handler functions and Workplans of the type PROCEDURE. It also describes the commands used for migrating from the old utilities NATUNLD/NATLOAD and SYSTRANS to the Object Handler.

For explanations of the variable values contained in the syntax diagrams shown in this section, refer to the relevant sections in the *Object Handler* documentation. For explanations of the symbols used in the syntax diagrams, see *System Command Syntax* in the *System Commands* documentation.

```
EXECUTE (procedure-workplan)
```

Executes a Workplan of the type PROCEDURE. See also the section [Workplans](#).

```
UNLOAD select-clause [parameter-setting] [option-setting]
```

Unloads the objects defined in the *select-clause* with the parameters defined in *parameter-setting* with the options defined in *option-setting*.

```
LOAD select-clause [parameter-setting] [option-setting]
```

Loads the objects defined in the *select-clause* with the parameters defined in *parameter-setting* with the options defined in *option-setting*.

```
LOADALL [parameter-setting] [option-setting]
```

Loads all objects from a work file with the parameters defined in *parameter-setting* with the options defined in *option-setting*.

```
SCAN select-clause [option-setting]
```

Scans a work file for the objects defined in the *select-clause* with the options defined in *option-setting*.

```
SCANALL [option-setting]
```

Scans a work file for all objects with the options defined in *option-setting*.

```
FIND select-clause [option-setting]
```

Finds the objects defined in the *select-clause* with the options defined in *option-setting* and writes a report of the objects found into a Natural text member stored in the Workplan library. In addition, a report of the objects found can be written to a specified report file as Natural text member in the Workplan library.

```
FINDLIB select-clause [option-setting]
```

Finds the libraries for Natural objects or Natural command processor sources defined in the *select-clause* with the options defined in *option-setting* and writes a report of the objects found into a Natural text member stored in the Workplan library. In addition, a report of the objects found can be written to a specified report file as Natural text member in the Workplan library.

```
DELETE select-clause [option-setting]
```

Deletes the objects defined in the *select-clause* with the options defined in *option-setting*.

**Restriction:** It is not possible to delete an FDT.

```
UNDELI select-clause [option-setting]
```

Unloads delete instructions for the objects defined in the *select-clause* with the options defined in *option-setting*.

```
RESTART [restart-text-member]
```

Continues an interrupted load function. This is only possible if information was written to a Natural text member in the Workplan library during the aborted load. See also **RESTART** in the section *option-setting (Direct Commands)* and **Restart Load**.

```
DISPLAY STATISTICS
```

Displays statistics information about the objects processed.

```
NATUNLD natunld-direct-command
```

Executes an Object Handler command in the syntax of the old utility NATUNLD. See also **Migration from NATUNLD/NATLOAD and SYSTRANS to the Object Handler**.

```
NATLOAD natload-direct-command
```

Executes an Object Handler direct command issued in the syntax of the old utility NATLOAD. See also **Migration from NATUNLD/NATLOAD and SYSTRANS to the Object Handler**.

```
SYSTRANS systrans-direct-command
```

Executes an Object Handler direct command issued in the syntax of the old utility SYSTRANS. See also **Migration from NATUNLD/NATLOAD and SYSTRANS to the Object Handler**.



# 42

## select-clause

---

▪ Syntax of select-clause .....	202
▪ SELECTION or LIST Workplan .....	202
▪ Natural Library Object and DDM Selection .....	203
▪ Natural-Related Debug Environment Selection .....	209
▪ Natural-Related Profile Selection .....	210
▪ Natural-Related DL/I Subfile Selection .....	212
▪ Natural System Error Message Selection .....	214
▪ Natural Command Processor Selection .....	215
▪ FDT Selection .....	217
▪ Application Selection .....	218
▪ Object Selection for Delete Instructions .....	221
▪ Help Text Selection .....	223

The *select-clause* comprises either a Workplan of the type SELECTION or LIST, or selection specifications for the objects, FDTs or applications to be processed.

This section describes the syntax that applies to the *select-clause*. The keywords and variable values contained in the syntax diagrams represent the parameters that can be used to specify object selection criteria. If indicated, a variable value must be supplied with a keyword.

## Syntax of select-clause

---

The *select-clause* consists of one of the following options:

```
{ (selection-workplan)
  (list-workplan)
  object-selection
  delete-instruction-selection
  help-text-selection }
```

The *selection-workplan* and *list-workplan* options are explained in *SELECTION or LIST Workplan* below.

The use of *object-selection* depends on the object type, DDM, FDT or application you want to process, for each of which the appropriate syntax and keywords are explained in the remainder of this section.

The *delete-instruction-selection* options are explained in *Delete Instructions for Selected Objects*.

The *help-text-selection* option is explained in *Help Text Selection*.

## SELECTION or LIST Workplan

---

A Workplan of the type SELECTION contains a header (`TYPE SELECTION`) and a selection from one of the following types of object or file: Natural library objects, Natural-related objects, Natural system error messages, Natural command processor sources, DDMs or Adabas FDTs (Field Definition Tables).

A Workplan of the type LIST contains a header (`TYPE LIST`) and a selection list of objects as described in the section *Object List - LIST Workplan*. Such an object list can be used for the UNLOAD, LOAD or FIND command only.

For further information on using Workplans, see the section *Workplans*.

## Natural Library Object and DDM Selection

This selection is used to select Natural objects for processing including Natural DDMs (data definition modules) and user-defined error messages.

The appropriate syntax is shown and explained in the following section.

- [Syntax of Natural Library Object and DDM Selection](#)

### Syntax of Natural Library Object and DDM Selection

```

object-name
LIBRARY library-name
[ DBID dbid FNR fnr
  [NAME vsam-name]
  [CIPHER cipher]
  [ { PASSWORD } password
    PSW ] ]
[OBJTYPE group-type]
[ SETNO set-number [SETUSER set-user] [SETLIBRARY set-library] ]
[NATTYPE object-type]
[SCKIND object-kind]
[MODE object-mode]
[FMNUM error-number-from]
[TONUM error-number-to]
[SLKIND message-type]
[LANGUAGE languages]
[DDMDBID dgm-dbid] [DDMFNR dgm-fnr]
[NATVERS natural-version]
[ DATE date
  [FMDATE date-from] [TODATE date-to] ]
[ [SIZE size]
  [FMSIZE size-from] [TOSIZE size-to] ]
[USERID user-id]
[TID terminal-id]
[except-clause]

```

*except-clause*

```

EXCEPT
( object-name
[LIBRARY library-name]
[OBJTYPE group-type]
[SCKIND object-kind]
[NATTYPE object-type]
[MODE object-mode]
[SLKIND message-type]
[FMNUM error-number-from] [TONUM error-number-to]
[LANGUAGE languages]
[DDMDBID dgm-dbid] [DDMFNR dgm-fnr]
[NATVERS natural-version]
[
    DATE date
    [FMDATE date-from] [TODATE date-to]
]
[
    SIZE size
    [FMSIZE size-from] [TOSIZE size-to]
]
[USERID user-id]
[TID terminal-id]
)

```

 **Notes:**

1. For the command `FINDLIB`, only the following keywords are processed: `LIBRARY`, `DBID`, `FNR`, `NAME`, `CIPHER` and `PASSWORD` or `PSW`.
2. When processing Natural DDMs, you must set `OBJTYPE` to `D`. In addition, some keywords do not apply to DDMs, which is described below.

**Keyword Explanation of Natural Library Object and DDM Selection**

The keywords and valid values for the objects to be processed are described in the following section.

Keyword	Valid Values	Default Value
<i>object-name</i>	<p>A valid object name or a range of names.</p> <p>If <i>object-name</i> contains blank characters, it must be enclosed in double quotation marks (" ").</p> <p>See also <i>Name</i> in <i>Name, Date and Time Specification</i>.</p>	none

Keyword	Valid Values	Default Value
LIBRARY	<p>A valid library name or a range of names.</p> <p>If <b>OBJTYPE</b> (see below) is set to D, the library name is ignored.</p> <p>If <b>SETNO</b> is specified, a range of names is not allowed.</p> <p>See also <i>Name</i>.</p>	none
DBID	<p>Not valid for DDMs on mainframes (<b>OBJTYPE</b> set to D; see below).</p> <p>A valid database ID.</p>	0 (current FNAT/FUSER)
FNR	<p>Not valid for DDMs on mainframes (<b>OBJTYPE</b> set to D; see below).</p> <p>A valid file number.</p>	0 (current FNAT/FUSER)
NAME	<p>Only applies to objects on mainframes. Not valid for DDMs on mainframes (<b>OBJTYPE</b> set to D; see below).</p> <p>A valid VSAM name.</p>	blank (current FNAT/FUSER)
CIPHER	<p>Only applies to objects on mainframes. Not valid for DDMs on mainframes (<b>OBJTYPE</b> set to D; see below).</p> <p>The 8-digit cipher code of the Adabas file where the objects are stored.</p>	blank (current FNAT/FUSER)
PASSWORD or PSW	<p>Only applies to objects on mainframes.</p> <p>Not valid for DDMs on mainframes (<b>OBJTYPE</b> set to D; see below).</p> <p>An 8-character Adabas password.</p>	blank (current FNAT/FUSER)
OBJTYPE	<p>Types of object are:</p> <ul style="list-style-type: none"> <li>D DDMs</li> <li>E Natural error messages</li> <li>N Natural programming objects</li> <li>* Asterisk (all)</li> </ul> <p>or a valid combination.</p> <p><b>Exception:</b> Object type D cannot be combined with any other type.</p>	*
SETNO	<p>Only applies to the unload and find functions and if Predict is installed. Not applicable to application objects (see also <i>Selecting Application Objects</i>).</p> <p>A one- or two-digit number that identifies the retained set to be used for the names of the objects to be processed. A retained set is created with the save set option of the LIST XREF command.</p> <p>If SETNO is specified, the value specified for <i>object-name</i> is ignored.</p>	none

Keyword	Valid Values	Default Value
	For detailed information on Predict sets, refer to the <i>Predict</i> documentation.	
SETUSER	<p>Only applies to the unload and find functions and if Predict is installed. Not applicable to application objects (see also <i>Selecting Application Objects</i>).</p> <p>The ID of the user who created the Predict set. If no ID is specified, the value of the system variable *USER (see also the <i>System Variables</i> documentation) is used.</p>	*USER
SETLIBRARY	<p>Only applies to the unload and find functions and if Predict is installed. Not applicable to application objects (see also <i>Selecting Application Objects</i>).</p> <p>The name of the library to be searched for a Predict set. If you do not specify SETLIBRARY, the library specified with LIBRARY is used instead.</p>	
NATTYPE	<p>Not applicable if OBJTYPE is set to D.</p> <p>One or more single-character codes for Natural object types:</p> <ul style="list-style-type: none"> <li>P Program</li> <li>N Subprogram</li> <li>S Subroutine</li> <li>C Copycode</li> <li>H Helproutine</li> <li>T Text</li> <li>7 Function</li> <li>8 Adapter</li> <li>G Global data area</li> <li>L Local data area</li> <li>A Parameter data area</li> <li>M Map</li> <li>4 Class</li> <li>3 Dialog</li> <li>5 Natural command processor</li> <li>9 Resource</li> <li>* All object types</li> </ul>	*
SCKIND	<p>Not applicable if OBJTYPE is set to D.</p> <p>The kind of Natural programming objects. Valid input values are:</p> <ul style="list-style-type: none"> <li>S Source objects: objects that are only stored in source form.</li> </ul>	A

Keyword	Valid Values	Default Value
	<p>C Cataloged objects: objects that are only stored in cataloged form.</p> <p>A All source and cataloged objects.</p> <p>W All STOWed objects: source and cataloged objects with identical date and time.</p> <p>B Source and cataloged objects if both exist.</p> <p><b>Note:</b> W and B are valid for the UNLOAD and FIND commands only. For LOAD and SCAN, W and B are valid entries, but they are treated like A (all objects). If data is processed in Transfer format, only S (source objects) or A applies.</p>	
MODE	<p>Not applicable if <b>OBJTYPE</b> is set to D.</p> <p>The programming mode of the Natural programming objects. Valid input values are:</p> <p>A Any.</p> <p>R All objects in reporting mode.</p> <p>S All objects in structured mode.</p>	
FMNUM	<p>A start number of Natural error messages.</p> <p>Valid range: 1 to 9999.</p>	1
TONUM	<p>An end number of Natural error messages.</p> <p>Valid range: 1 to 9999.</p> <p>The value must be greater than or equal to the value of FMNUM, if specified.</p>	9999 or value of FMNUM (if specified)
SLKIND	<p>The kind of Natural error message text. Valid input values are:</p> <p>S Short text. Cannot be applied to the DELETE command (see <i>Basic Command Syntax</i>).</p> <p>L Long text.</p> <p>A Short and/or long text.</p> <p>B Short and long text, if both exist.</p>	A
LANGUAGE	<p>Up to 8 valid language codes (for example, code 1 for English) of user-defined error messages. An asterisk (*) selects all language codes.</p>	*
DDMDBID	<p>The valid database ID (1 to 65535) of a DDM.</p> <p>UNLOAD, LOAD and SCAN: 0 denotes that no check is performed. DDMs are processed, regardless of their database ID (DBID).</p>	0

Keyword	Valid Values	Default Value
DDMFNR	The valid file number (1 to 65535) of a DDM.  UNLOAD, LOAD and SCAN: 0 denotes that no check is performed. DDMs are processed, regardless of their file number (FNR).	0
NATVERS	The Natural version of Natural programming objects.  Valid version format: <i>VRS</i> where <i>V</i> is the 1-digit version, <i>R</i> the 1-digit release, and <i>S</i> the 2-digit system maintenance level. You can also specify a range of versions: see <i>Name</i> .	blank (no check)
DATE	The save or catalog date of Natural programming objects.  You can add a time by inserting a blank between date and time. For the format and ranges allowed, see <i>Date</i> and <i>Time</i> in <i>Name, Date and Time Specification</i> .  Special terms allowed are YESTERDAY and TODAY. See <i>Special Dates</i> in <i>Date</i> .	blank (no check)
FMDATE	A start value:  The date on or after which Natural programming objects were cataloged or saved. The format is identical to DATE. See <i>Date</i> .  Special terms allowed are YEAR, MONTH, YESTERDAY and TODAY. See <i>Special Dates</i> in <i>Date</i> .	blank (no check)
TODATE	An end value:  The date on or before which Natural programming objects were cataloged or saved. The format is identical to DATE. See <i>Date</i> .  Special terms allowed are YEAR, MONTH, YESTERDAY and TODAY. See <i>Special Dates</i> in <i>Date</i> .	blank (no check) or high value (if FMDATE specified)
SIZE	The size of Natural programming objects (up to 7 digits).	0 (no check)
FMSIZE	A start value:  The minimum size of Natural programming objects (up to 7 digits).	0 (no check)
TOSIZE	An end value:  The maximum size of Natural programming objects (up to 7 digits).	0 (no check) or high value (if FMSIZE specified)
USERID	The ID of the user who saved or cataloged the Natural programming objects.  You can also specify a range of user IDs: see <i>Name</i> .	blank (no check)

Keyword	Valid Values	Default Value
TID	<p>Not applicable if <b>OBJTYPE</b> is set to D.</p> <p>The ID of the terminal where the Natural programming objects were saved or cataloged (provided by the Natural system variable *INIT-ID).</p> <p>You can also specify a range of terminal IDs: see also <i>Name</i>.</p>	blank (no check)
EXCEPT	All items that match the selection criteria entered before EXCEPT are checked against <i>all</i> parameters contained within the parentheses following the keyword EXCEPT. If they match all these parameters too, they are not processed.	not applicable



#### Notes:

- Parameters that are irrelevant for **OBJTYPE** are ignored. For example: **DATE**, **SIZE** and **USERID** have no meaning for Natural error messages.
- DBID**, **FNR**, **NAME**, **CIPHER** and **PASSWORD** or **PSW** are ignored by the **LOAD** or **SCAN** command. These parameters must instead be specified in the *parameter-setting* clause as described for **LOADFNAT...** and **LOADFUSER...** in *Keyword Explanation of parameter-clause*.

## Natural-Related Debug Environment Selection

This selection is used to select Natural-related debug environments for processing.

The appropriate syntax is shown and explained in the following section.

- [Syntax of Natural-Related Debug Environment Selection](#)

### Syntax of Natural-Related Debug Environment Selection

```

object-name
NATPATH DEBUG
[LIBRARY library-name]
[ DBID dbid [FNR fnr] ]
[NAME vsam-name]
[CIPHER cipher]
[ { PASSWORD } password ]
[ EXCEPT
  (object-name

```

```
[LIBRARY library-name
]
```

## Keyword Explanation of Natural-Related Debug Environment Selection

The keywords and valid input values for the debug environments to be processed are described in the following section.

Keyword	Valid Values	Default Value
<i>object-name</i>	A valid debug environment name or a range of names. See also <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .	none
LIBRARY	A valid library name or a range of names. See also <a href="#">Name</a> .	none
DBID	A valid database ID.	0 (current FUSER)
FNR	A valid file number.	0 (current FUSER)
NAME	A valid VSAM name.	blank (current FUSER)
CIPHER	The 8-digit cipher code of the Adabas file where the debug environments are stored.	blank (current FUSER)
PASSWORD or PSW	An 8-character Adabas password.	blank (current FUSER)
EXCEPT	See <a href="#">EXCEPT</a> in <i>Natural Library Object and DDM Selection</i> .	not applicable



**Note:** DBID, FNR, NAME, CIPHER and PASSWORD or PSW are ignored by the LOAD or SCAN command. These parameters must instead be specified in the *parameter-setting* clause as described for LOADFNAT... and LOADFUSER... in [Keyword Explanation of parameter-clause](#).

## Natural-Related Profile Selection

This selection is used to select Natural-related profiles for processing.

The appropriate syntax is shown and explained in the following section.

- Syntax of Natural-Related Profile Selection

## Syntax of Natural-Related Profile Selection

```

object-name
NATPATH PROFILE
[OBJTYPE profile-type]
[ DBID dbid [FNR fnr] ]
[NAME vsam-name]
[CIPHER cipher]
[ { PASSWORD } password ]
[ EXCEPT
  (object-name
  [OBJTYPE profile-type]
  )]

```

## Keyword Explanation of Natural-Related Profile Selection

The keywords and valid input values for the profiles to be processed are described in the following section.

Keyword	Valid Values	Default Value														
<i>object-name</i>	A valid profile name or a range of names. See also <i>Name</i> in <i>Name, Date and Time Specification</i> .	none														
OBJTYPE	The type of profile: <table border="1"> <tbody> <tr> <td></td> <td></td> </tr> <tr> <td>D</td> <td>Device profile</td> </tr> <tr> <td>E</td> <td>Editor profile</td> </tr> <tr> <td>M</td> <td>Map profile</td> </tr> <tr> <td>P</td> <td>Parameter profile</td> </tr> <tr> <td>*</td> <td>Asterisk (all profile types)</td> </tr> <tr> <td></td> <td></td> </tr> </tbody> </table> or any combination.			D	Device profile	E	Editor profile	M	Map profile	P	Parameter profile	*	Asterisk (all profile types)			*
D	Device profile															
E	Editor profile															
M	Map profile															
P	Parameter profile															
*	Asterisk (all profile types)															
DBID	A valid database ID.	0 (current FNAT)														
FNR	A valid file number.	0														

Keyword	Valid Values	Default Value
		(current FNAT)
NAME	A valid VSAM name.	blank (current FNAT)
CIPHER	The 8-digit cipher code of the Adabas file where the profiles are stored.	blank (current FNAT)
PASSWORD or PSW	An 8-character Adabas password.	blank (current FNAT)
EXCEPT	See <a href="#">EXCEPT</a> in <i>Natural Library Object and DDM Selection</i> .	not applicable



**Note:** DBID, FNR, NAME, CIPHER and PASSWORD or PSW are ignored by the LOAD or SCAN command. These parameters must instead be specified in the *parameter-setting* clause as described for LOADFNAT... and LOADFUSER... in [Keyword Explanation of parameter-clause](#).

## Natural-Related DL/I Subfile Selection

This selection is used to select Natural-Related DL/I subfiles for processing.

The appropriate syntax is shown and explained in the following section.

- [Syntax of Natural-Related DL/I Subfile Selection](#)

### Syntax of Natural-Related DL/I Subfile Selection

```

object-name
NATPATH SUBFILE
[OBJTYPE subfile-type]

[ DBID dbid [FNR fnr] ]
[NAME vsam-name]
[CIPHER cipher]
[ { PASSWORD } password ]
[ { PSW } ]
[ EXCEPT
  (object-name

```

```
[OBJTYPE subfile-type]
```

```
)]
```

### Keyword Explanation of Natural-Related DL/I Subfile Selection

The keywords and valid input values for the DL/I subfiles to be processed are described in the following section.

Keyword	Valid Values	Default Value								
<i>object-name</i>	A valid DL/I subfile name or a range of names. See also <i>Name</i> in <i>Name, Date and Time Specification</i> .	none								
OBJTYPE	The type of DL/I subfile: <table border="1" data-bbox="435 716 922 898"> <tr> <td></td> <td></td> </tr> <tr> <td>D</td> <td>NDB</td> </tr> <tr> <td>P</td> <td>NSB</td> </tr> <tr> <td>*</td> <td>Asterisk (both subfile types)</td> </tr> </table>			D	NDB	P	NSB	*	Asterisk (both subfile types)	*
D	NDB									
P	NSB									
*	Asterisk (both subfile types)									
DBID	A valid database ID.	0 (current FDIC)								
FNR	A valid file number.	0 (current FDIC)								
NAME	A valid VSAM name.	blank (current FDIC)								
CIPHER	The 8-digit cipher code of the Adabas file where the DL/I subfiles are stored.	blank (current FDIC)								
PASSWORD or PSW	An 8-character Adabas password.	blank (current FDIC)								
EXCEPT	See <b>EXCEPT</b> in <i>Natural Library Object and DDM Selection</i> .	not applicable								



**Note:** DBID, FNR, NAME, CIPHER and PASSWORD or PSW are ignored by the LOAD or SCAN command. These parameters must instead be specified in the *parameter-setting* clause as described for LOADFNAT... and LOADFUSER... in *Keyword Explanation of parameter-clause*.

## Natural System Error Message Selection

This selection is used to select Natural system error messages for processing.

The appropriate syntax is shown and explained in the following section.

- [Syntax of Natural System Error Message Selection](#)

### Syntax of Natural System Error Message Selection

```

ERROR NATERROR
  DBID dbid FNR
  [ fnr [NAME
    vsam-name] [ { PASSWORD
                    PSW } password ] ]
  [CIPHER
  cipher]
  [FMNUM error-number-from] [TONUM error-number-to]
  [SLKIND message-type]
  [LANGUAGE languages]
  [ EXCEPT
    (
      [FMNUM error-number-from] [TONUM error-number-to]
      [SLKIND message-type]
      [LANGUAGE languages]
    )
  ]

```

### Keyword Explanation of Natural System Error Message Selection

The keywords and valid input values for the Natural system error messages to be processed are described in the following section.

Keyword	Valid Values	Default Value
DBID	Only applies to system error messages on mainframes. A valid database ID.	0 (current FNAT)
FNR	Only applies to system error messages on mainframes. A valid file number.	0 (current FNAT)
NAME	Only applies to system error messages on mainframes. A valid VSAM name.	blank (current FNAT)

Keyword	Valid Values	Default Value
CIPHER	Only applies to system error messages on mainframes.  The 8-digit cipher code of the Adabas file where the system error messages are stored.	blank (current FNAT)
PASSWORD or PSW	Only applies to system error messages on mainframes.  An 8-character Adabas password.	blank (current FNAT)
FMNUM	A start number of system error messages.  Valid range: 1 to 9999.	1
TONUM	An end number of system error messages.  Valid range: 1 to 9999.  The value must be greater than or equal to the value of FMNUM if specified.	9999 or value of FMNUM (if specified)
SLKIND	See <a href="#">SLKIND</a> in <i>Natural Library Object and DDM Selection</i> .	A
LANGUAGE	Up to 8 valid language codes (for example, code 1 for English) of system error messages.  An asterisk (*) selects all language codes.	*
EXCEPT	See <a href="#">EXCEPT</a> in <i>Natural Library Object and DDM Selection</i> .	



**Note:** DBID, FNR, NAME, CIPHER and PASSWORD or PSW are ignored by the LOAD or SCAN command. These parameters must instead be specified in the *parameter-setting* clause as described for LOADFNAT . . . in [Keyword Explanation of parameter-clause](#).

## Natural Command Processor Selection

This selection is used to select Natural command processor sources for processing.

The appropriate syntax is shown and explained in the following section.

- Syntax of Natural Command Processor Source Selection

### Syntax of Natural Command Processor Source Selection

```

object-name PROCESSOR ncp-library-name
[
    DBID ncp-dbid FNR ncp-fnr [file-options] ]
[EXCEPT
    (object-name
    [LIBRARY ncp-library-name]
    )]

```

*file-options*

```

[NAME ncp-vsam-name]
[CIPHER ncp-cipher]
[ { PASSWORD } ncp-password ]
  { PSW

```

 **Note:** For the command FINDLIB, only the following keywords are processed: PROCESSOR, DBID, FNR, NAME, CIPHER and PASSWORD or PSW.

### Keyword Explanation of Natural Command Processor Source Selection

The keywords and valid input values for the Natural command processor sources to be processed are described in the following section.

Keyword	Valid Values	Default Value
<i>object-name</i>	The name of a valid Natural command processor source or a range of names.  See also <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .	none
PROCESSOR	A valid library name or a range of names.  See also <a href="#">Name</a> .	none
DBID	The valid database ID of the Adabas file where the Natural command processor sources are stored.	0 (current FNAT/FUSER)
FNR	The valid file number of the Adabas file where the Natural command processor sources are stored.	0 (current FNAT/FUSER)

Keyword	Valid Values	Default Value
NAME	Only applies to Natural command processor sources on mainframes. A valid VSAM name.	blank
CIPHER	The 8-digit cipher code of the Adabas file where the Natural command processor sources are stored.	blank
PASSWORD or PSW	The 8-character Adabas password of the Adabas file where the Natural command processor sources are stored.	blank
EXCEPT	See <a href="#">EXCEPT</a> in <i>Natural Library Object and DDM Selection</i> .	



**Note:** DBID, FNR, NAME, CIPHER and PASSWORD or PSW are ignored by the LOAD or SCAN command. These parameters must instead be specified in the *parameter-setting* clause as described for LOADNCP. . . in [Keyword Explanation of parameter-clause](#).

## FDT Selection

This selection is used to select Adabas FDTs (Field Definition Tables) for processing.

For loading FDTs, see also [FDTs](#) in the section *Object Specification*.

The appropriate syntax is shown and explained in the following section.

- [Syntax of FDT Selection](#)

### Syntax of FDT Selection

```
FDT
DBID dbid
{ FNR fnr [CIPHER cipher] [ { PASSWORD }
  PSW } password ] }
FMFNR fnr-start TOFNR fnr-end
```

## Keyword Explanation of FDT Selection

The keywords and valid input values for the FDTs to be processed are described in the following section.

Keyword	Valid Values	Default Value
DBID	The database ID of the FDT.	none
FNR	The file number of the FDT.	none
CIPHER	The 8-digit Adabas cipher code of the FDT.	none
PASSWORD or PSW	The 8-character Adabas password of the FDT.	none
FMFNR	Only applies to the FIND or UNLOAD command.  A start value: The file number (FNR) of an FDT.	none
TOFNR	Only applies to the FIND or UNLOAD command.  An end value: The file number (FNR) of an FDT.	none

## Application Selection

---

This selection applies to applications created and maintained in Natural Studio's application workspace and the libraries or objects that belong to these applications.

The appropriate syntax is shown and explained in the following section.

- [Selecting Base and Compound Applications](#)
- [Selecting Application Libraries](#)
- [Selecting Application Objects](#)

### Selecting Base and Compound Applications

This selection only applies to the find function.

**Syntax**

```

APPLICATION APNAME application-name
[APTYPE application-type]
[COMPAPPLICATION compound-application-name]
[
    EXCEPT
    (APNAME application-name
    [APTYPE application-type]
    )]

```

**Selecting Application Libraries**

This selection only applies to the find function.

**Syntax**

```

APPLICATION APLIBRARY application-library-name
[BASEAPPLICATION base-application-name]
[ COMPAPPLICATION compound-application-name]
[
    DBID dbid [FNR fnr] ]
[
    EXCEPT
    (APLIBRARY application-library-name
    [BASEAPPLICATION base-application-name]
    )]

```

**Selecting Application Objects**

This selection only applies to the find and unload functions.

**Syntax**

```

APPLICATION APOBJECTS application-object-name
[BASEAPPLICATION base-application-name]
[COMPAPPLICATION compound-application-name]
[LIBRARY library-name]
[object-specification]
[
    EXCEPT
    (APOBJECT application-object-name
    [LIBRARY library-name]
    )]

```

```
[BASEAPPLICATION base-application-name]  
[object-specification]  
)]
```

### Keyword Explanation of Application Selection

The keywords and valid input values for the applications, application libraries or application objects to be processed are described in the following section.

Keyword	Valid Values	Default Value								
APNAME	A valid name of a Natural application or a range of names.  See also <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .	*								
APTYPE	A valid application type: <table border="1" data-bbox="457 758 1094 940"> <tr> <td></td> <td></td> </tr> <tr> <td>B</td> <td>Base application</td> </tr> <tr> <td>0</td> <td>Compound application</td> </tr> <tr> <td>*</td> <td>All: base and/or compound applications</td> </tr> </table>			B	Base application	0	Compound application	*	All: base and/or compound applications	*
B	Base application									
0	Compound application									
*	All: base and/or compound applications									
COMPAPPLICATION	Only applies if APTYPE is set to * or B.  The name of a compound application to which the specified base application belongs or a range of names.  Only base applications that belong to the specified compound application(s) are selected; base applications that do not belong to a compound application are not selected.	none								
EXCEPT	See <a href="#">EXCEPT</a> in <i>Natural Library Object and DDM Selection</i> .	not applicable								
APLIBRARY	The valid name of a library that belongs to a Natural base or compound application or a range of names.  See also <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .	*								
BASEAPPLICATION	The valid name of a Natural base application to which an application library or application object belongs.  See also <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .	*								
DBID	The valid database ID of an application library.	0 (no check)								
FNR	The valid file number of an application library.	0 (no check)								
APOBJECT	The valid name of an application object that belongs to a base or compound application, or a range of names.  See also <a href="#">Name</a> in <i>Name, Date and Time Specification</i> .	*								

Keyword	Valid Values	Default Value
LIBRARY	A valid library name or a range of names.  If <b>OBJTYPE</b> is set to D (see <i>Natural Library Object and DDM Selection</i> ), the library name is ignored.  See also <i>Name</i> in <i>Name, Date and Time Specification</i> .	*
<i>object-specification</i>	Indicates that additional selection criteria can be specified for application objects as shown in the <b>syntax</b> diagram for Natural library objects and DDMs: all items listed below <b>LIBRARY</b> <i>library-name</i> can also be applied to application objects whereas <i>object-name</i> in the EXCEPT clause is irrelevant for application objects.	not applicable

## Object Selection for Delete Instructions

This selection is used to specify delete instructions for Natural library objects, DDMs, user-defined error messages and Natural system error messages. The delete instructions are executed when a work file of internal format is loaded in the target environment with the **DELETEALLOWED** option specified.

The appropriate syntax is shown and explained in the following section.

- [Syntax of Delete Instructions for Natural Library Objects and DDMs](#)
- [Syntax of Delete Instructions for User-Defined Error Messages](#)
- [Syntax of Delete Instructions for Natural System Error Messages](#)

### Syntax of Delete Instructions for Natural Library Objects and DDMs

```

object-name
LIBRARY library-name
[
  OBJTYPE { N }
  OBJTYPE { D }
]
[
  NATTYPE { * }
]
[SKIND object-kind]

```

## Keyword Explanation of Delete Instructions for Natural Library Objects and DDMs

The keywords and valid values for the objects to be processed are described in the following section.

Keyword	Valid Values	Default Value
<i>object-name</i>	A valid object name or a start value ( <i>value*</i> ) for a range of names such as ABC*.	none
LIBRARY	A valid library name. A range specification is <i>not</i> allowed.  If <b>OBJTYPE</b> (see below) is set to D, the library name is ignored.	none
OBJTYPE	A valid object-type code:  D    DDMs N    Natural programming objects  Object type D cannot be combined with object type N.	*
NATTYPE	Not applicable if <b>OBJTYPE</b> is set to D.  A Natural object type. Valid input values are:  * All object types	*
SCKIND	Not applicable if <b>OBJTYPE</b> is set to D.  The kind of Natural programming objects. Valid input values are:  S    Source objects. If used in the <i>except-clause</i> (see <i>Syntax of Natural Library Object and DDM Selection</i> ): objects that are stored only in source form. C    Cataloged objects. If used in the <i>except-clause</i> : objects that are stored only in cataloged form. A    All source and cataloged objects.	A

## Syntax of Delete Instructions for User-Defined Error Messages

```
*
LIBRARY library-name
OBJTYPE E
FMNUM error-number-from
[TONUM error-number-to]
[SLKIND message-type]
[LANGUAGE languages]
```

*library-name* denotes the name of a single library; a range specification is not allowed.

For explanations of the other elements used in this syntax, see [Keyword Explanation of Natural Library Object and DDM Selection](#).

### Syntax of Delete Instructions for Natural System Error Messages

```
ERROR NATERROR
FMNUM error-number-from
[TONUM error-number-to]
[SLKIND message-type]
[LANGUAGE languages]
```

For explanations of the elements used in this syntax, see [Keyword Explanation of Natural System Error Message Selection](#).

## Help Text Selection

This selection is used to specify that Natural help texts are processed during the unload, load, scan and find functions when internal format is used for the work files. These help texts are identical to the help information provided by the Natural Help utility, which is invoked with the HELP system command (see the *System Commands* documentation).

### Syntax of Help Text Selection

```
ERROR NATURAL HELP
[ DBID dbid FNR fnr [NAME vsam-name]
  [CIPHER cipher] [ { PASSWORD } password ] ]
```

For explanations of the elements used in this syntax, see [Keyword Explanation of Natural Library Object and DDM Selection](#).



# 43

## Object List - LIST Workplan

---

- Syntax of object-type-and-location ..... 226
- Syntax of object-name-description ..... 228
- Example of an Object List ..... 627

An object list is a Workplan of the type LIST, which specifies object selection criteria for the objects to be processed in the UNLOAD, LOAD or FIND command. An object list can be used as an alternative to the *select-clause* and the SELECTION Workplan.

The following syntax applies to an object list:

```
TYPE LIST
{ object-type-and-location (object-name-description ...) } ...
```

The syntactical options are explained in the following section. The keywords and variable values contained in the syntax diagrams shown in this section represent parameters that are used to specify object selection criteria. If indicated, a variable value must be supplied with a keyword. Each syntax element (except for the ones enclosed in parentheses) must start on a new line and end on the same line.

For explanations of the keywords contained in the syntax diagrams, refer to the section *select-clause*.

## Syntax of object-type-and-location

The syntax diagrams that apply to *object-type-and-location* are shown in the following section.

- [Natural Objects and DDMs](#)
- [Natural System Error Messages](#)
- [Natural Command Processor Sources](#)
- [Natural-Related Debug Environments](#)
- [Natural-Related Profiles](#)
- [Natural-Related DL/I Subfiles](#)
- [FDTs](#)

### Natural Objects and DDMs

```
LIBRARY library-name
[ DBID dbid FNR fnr [NAME vsam-name] [CIPHER [ { PASSWORD } password ] ] ]
[ OBJTYPE group-type ]
```



#### Notes:

1. No ranges are allowed for *library-name*.

2. For DDMs, OBJTYPE must be set to D.

## Natural System Error Messages

```
ERROR NATERROR
[ DBID dbid FNR fnr [NAME vsam-name] [CIPHER [ { PASSWORD } password ] ]
  cipher] [ { PSW } ] ]
```

## Natural Command Processor Sources

```
PROCESSOR ncp-library-name
[ DBID dbid FNR fnr [NAME vsam-name] [CIPHER [ { PASSWORD } password ] ]
  cipher] [ { PSW } ] ]
```



**Note:** No ranges are allowed for *ncp-library-name*.

## Natural-Related Debug Environments

```
NATPATH DEBUG
LIBRARY library-name
[ [ DBID dbid [FNR fnr] ] [NAME vsam-name] [CIPHER [ { PASSWORD } password ] ]
  [CIPHER cipher] [ { PSW } ] ] ]
```

## Natural-Related Profiles

```
NATPATH PROFILE
[ [ DBID dbid [FNR fnr] ] [NAME vsam-name] [CIPHER [ { PASSWORD } password ] ]
  [CIPHER cipher] [ { PSW } ] ] ]
```

## Natural-Related DL/I Subfiles

```
NATPATH SUBFILE
[ [ DBID
  [ dbid
  [FNR
  fnr] ] [NAME vsam-name] [CIPHER cipher] [ { PASSWORD } password ] ] ]
```

## FDTs

```
FDT
```

## Syntax of object-name-description

The syntax diagrams that apply to *object-name-description* are shown in the following section:

- [Natural Objects](#)
- [DDMs](#)
- [Natural System Error Messages](#)
- [Natural Command Processor Sources](#)
- [Natural-Related Debug Environments](#)
- [Natural-Related Profiles](#)
- [Natural-Related DL/I Subfiles](#)
- [FDTs](#)

## Natural Objects

```
{ object-name [SCKIND object-kind]
  error-number [SLKIND message-type] [LANGUAGE languages]
  FMNUM error-number-from TONUM error-number-to [SLKIND message-type] [LANGUAGE
  languages] }
```

**DDMs**`object-name`**Natural System Error Messages**

```

error-number [ SLKIND message-type ] [ LANGUAGE languages ]
{ FMNUM error-number-from TONUM error-number-to [ SLKIND message-type ] [ LANGUAGE
languages ] }

```

**Natural Command Processor Sources**`object-name`**Natural-Related Debug Environments**`object-name`**Natural-Related Profiles**`object-name [ OBJTYPE profile-type ]`**Natural-Related DL/I Subfiles**`object-name [ OBJTYPE subfile-type ]`**FDTs**

```

DBID dbid FNR fnr [ CIPHER cipher ] [ { PASSWORD } password ]
[ { PSW } password ]

```

## Example of an Object List

---

The following is an example of a Workplan of the type LIST:

```
TYPE LIST
  LIBRARY LIB-1 OBJTYPE N      /* process Natural objects from library 'LIB-1'
    ( A* SCKIND S              /* all sources objects whose names start with 'A'
      B1                        /* source and/or cataloged object of 'B1'
    CDE> SCKIND C )           /* all cataloged objects with names greater than/equal
to 'CDE'
  /*                            /* comment line
  LIBRARY LIB-2                /* process Natural objects from library 'LIB-2'
                                /* including error messages and shared resources
  ( *                           /* all source and/or cataloged objects
                                /* including shared resources
  FMNUM 1 TONUM 100           /* error messages from 1 to 100
  )
```

# 44 parameter-setting

---

- Syntax of parameter-clause ..... 232
- Keyword Explanation of parameter-clause ..... 233

The *parameter-setting* clause is used to change attributes for the LOAD or UNLOAD command for the objects to be processed and to define target destinations for the LOAD command (for example, FNAT).

The following syntax applies to the *parameter-setting* clause:

```
WITH
{
  (parameter-workplan)
  parameter-clause
}
```

For an explanation of the syntax that applies to *parameter-workplan*, refer to [Referencing Workplans](#) in the section *Workplans*.

This section covers the following topics:

## Syntax of parameter-clause

The syntax of the *parameter-clause* is shown in the following diagram. If indicated, a variable value must be supplied with a keyword.

```
[
  [NAME old-name] NEWNAME new-name ]
[
  [LIBRARY old-library-name]
  NEWLIBRARY new-library-name ]
[
  LOADFNATDBID fnat-dbid LOADFNATFNR fnat-fnr
  [LOADFNATNAME vsam-name]
  [LOADFNATCIPHER fnat-cipher]
  [
    {
      LOADFNATPASSWORD
      LOADFNATPSW
    }
    fnat-password
  ]
  [LOADFUSERDBID fuser-dbid LOADFUSERFNR fuser-fnr
  [LOADFUSERNAME fuser-vsam-name]
  [LOADFUSERCIPHER fuser-cipher]
  [
    {
      LOADFUSERPASSWORD
      LOADFUSERPSW
    }
    fuser-password
  ]
  [LOADNCPDBID ncp-file-dbid LOADNCPFNR ncp-file-fnr
  [LOADNCPNAME ncp-file-vsam-name]
  [LOADNCPCIPHER ncp-file-cipher]
  [
    {
      LOADNCPPASSWORD
      LOADNCPPSW
    }
    ncp-file-password
  ]
]
```

```

[ [FDTDBID old-fdt-dbid FDTFNR old-fdt-fnr] NEWFDTDBID new-fdt-dbid NEWFDTFNR
  new-fdt-fnr ]
[ERRNUMDIFF modification-of-error-message-range]
[ [LANGUAGE old-language]
  NEWLANGUAGE new-language ]
[ [DATE old-date] NEWDATE new-date ]
[ [USERID old-userid] NEWUSERID
  new-userid ]
[ [TID old-terminal-id] NEWTID
  new-terminal-id ]
[ [PATH old-external-path-name]
  NEWPATH new-external-path-name ]

```

## Keyword Explanation of parameter-clause

The keywords and variable values (if relevant) of the *parameter-clause* are explained in the following section.

Keyword	Values	Restricted to Command
NAME	The object name to be checked if NEWNAME is specified.	
NEWNAME	A new object name. <b>Note:</b> Not applicable to DDMs on mainframe platforms.	
LIBRARY	The library name to be checked if NEWLIBRARY is specified.	
NEWLIBRARY	A new library name. <b>Note for the LOAD function:</b> NEWLIBRARY does <i>not</i> affect the library name used in the delete instruction of a work file that is processed with the <b>DELETEALLOWED</b> option.	
LOADFNATDBID	The database ID (DBID) of FNAT libraries.	LOAD
LOADFNATFNR	The file number (FNR) of FNAT libraries.	LOAD
LOADFNATNAME	Only applies to objects on mainframes. An FNAT VSAM file name.	LOAD
LOADFNATCIPHER	Only applies to objects on mainframes. An FNAT cipher code.	LOAD

Keyword	Values	Restricted to Command
LOADFNATPASSWORD or LOADFNATPSW	Only applies to objects on mainframes. An FNAT Adabas password.	LOAD
LOADFUSERDBID	The DBID of FUSER libraries.	LOAD
LOADFUSERFNR	The FNR of FUSER libraries.	LOAD
LOADFUSERNAME	Only applies to objects on mainframes. An FUSER VSAM file name.	LOAD
LOADFUSERCIPHER	Only applies to objects on mainframes. An FUSER cipher code.	LOAD
LOADFUSERPASSWORD or LOADFUSERPSW	Only applies to objects on mainframes. An FUSER Adabas password.	LOAD
LOADNCPDBID	The DBID of the Adabas file for Natural command processor sources.	LOAD
LOADNCPFNR	The FNR of the Adabas file for Natural command processor sources.	LOAD
LOADNCPNAME	Only applies to objects on mainframes. The VSAM name of the Adabas file for Natural command processor sources.	LOAD
LOADNCPCIPHER	The cipher code of the Adabas file for Natural command processor sources.	LOAD
LOADNCPPASSWORD or LOADNCPPSW	Only applies to objects on mainframes. The Adabas password of the Adabas file for Natural command processor sources.	LOAD
FDTDBID	The DBID of the Adabas FDT (Field Definition Table) to be checked if NEWFDTDBID is specified.	
NEWFDTDBID	A new DBID of the FDT.	
FDTFNR	The DBID of the FDT to be checked if NEWFDTFNR is specified.	
NEWFDTFNR	A new FNR of the FDT.	
ERRNUMDIFF	A number (positive or negative) that is to be added to the Natural error messages during the UNLOAD or LOAD command.  ERRNUMDIFF can only be specified if FMNUM and TONUM (see <i>select-clause</i> ) have been specified as selection criteria. Otherwise, it is not possible to check for valid results.	
LANGUAGE	Up to 8 valid language codes (for example, code 1 for English) of Natural error messages to be checked if NEWLANGUAGE (see below) is specified.  If <i>language</i> contains more than one language code, <i>new-language</i> must contain the same numbers of language codes. Each <i>language</i>	

Keyword	Values	Restricted to Command
	language code is replaced by the language code in the corresponding position of <i>new-language</i> .  If <i>language</i> is not specified, <i>new-language</i> must not contain more than one language code.	
NEWLANGUAGE	Up to 8 valid language codes (for example, code 4 for Spanish) for new user-defined error messages. This option does not apply to the long texts of Natural system error messages for which English (language code 1) is the only valid language.  See also <a href="#">LANGUAGE</a> above.	
DATE	An object date.  You can add a time by inserting a blank between date and time. For the format and ranges allowed, see <a href="#">Date</a> and <a href="#">Time</a> in <i>Name, Date and Time Specification</i> .	
NEWDATE	A new object date.  NEWDATE can be a date followed by a time value. You can add a time by inserting a blank between date and time. See also <a href="#">Date</a> and <a href="#">Time</a> in <i>Name, Date and Time Specification</i> .	
USERID	The user ID to be checked if NEWUSERID is specified.	
NEWUSERID	A new user ID.	
TID	Only applies to objects on mainframes. The terminal ID to be checked if NEWTID is specified.	
NEWTID	Only applies to objects on mainframes. A new terminal ID.	
PATH	The path name to be checked if NEWPATH is specified.	
NEWPATH	A new path name.	



#### Notes:

1. Parameters not applicable to the selection criterion processed are ignored.
2. LOADFNAT . . . , LOADFUSER . . . and LOADNCP . . . are used for the LOAD command only, and ignored otherwise.
3. LOADFNAT . . . is used for libraries starting with SYS (except SYSTEM).
4. LOADFUSER . . . is used for libraries not starting with SYS (but including SYSTEM).
5. LOADNCP . . . is used for Natural command processor sources.



# 45 option-setting

---

▪ Syntax of option-setting .....	238
▪ Keyword Explanation of option-setting .....	240

The *option-setting* clause is used to change the default values of Object Handler command options.

The syntax that applies to the *option-setting* clause is shown and explained in the following section. The keywords and variable values contained in the syntax diagrams shown represent the parameters that are used to specify the default values. If indicated, a variable value must be supplied with a keyword.

## Syntax of option-setting

```
WHERE
{
  (option-workplan)
  option-clause
}
```

The syntax diagram that applies to *option-workplan* is shown and described in [Referencing Workplans](#) in the section *Workplans*.

The syntax of the *option-clause* is shown in the following section.

- [Syntax of option-clause](#)

### Syntax of option-clause

```
[ REPLACE { ALL
             OBSOLETE
             EXCEPT } ]
[ transfer-options
  internal-format-options ]
[ NOREPORT
  NEWREPORT [file-name]
  REPORT [file-name]
  BATCHREPORT ]
[ NORESTART
  RESTART [restart-text-member] ]
[NUMBERPROCESS number]
[FIXEDLENGTH]
[FDIC (dbid,fnr,password,cipher)]
[FSEC (dbid,fnr,password,cipher)]
[ { WORKFILETYPE
    WFTYPE } { DEFAULT
    UTF-8 } ]
```

[ { PC NEWPC } [file-name] ]
[ { PCCOMMANDFILENUMBER PCCOM PCCFN } command-file-number ]

## Separators

Commas must be used as separators between the values following the FDIC and FSEC keywords, or if a value is missing. For example: FDIC (10,21,,2a).

If the session parameter ID (see *ID - Input Delimiter Character* in the *Parameter Reference* documentation) has been set to a comma, use a slash (/) as the separator between values.

*transfer-options*

TRANSFER
[ <u>CONVERSION-TABLE</u> { <u>SYSTEM-TABLE</u> <u>USER-TABLE</u> } [conversion-program] ]
[ <u>SUBSTITUTE</u> ]
[ <u>INCLUDE-LINE-NUMBERS</u> ]
[ <u>UPPERCASE-TRANSLATION</u> ]
[ <u>INCORPORATE-FREE-RULES</u> ]
[ <u>LOAD-CODE-PAGE</u> code-page-name]
[ <u>DA-FORMAT</u> data-area-format]

*internal-format-options*

[ XREF { ON OFF DOC FORCE <u>SPECIAL</u> } ]
[ <u>DELETEALLOWED</u> ]
[ <u>NOSYMBOLTABLE</u> ]
[ <u>VERSIONCHECK</u> ]

## Keyword Explanation of option-setting

The keywords and the variable values (if relevant) of *option-setting* are explained in the following section:

Option	Explanation	Restricted to Command
REPLACE	Replaces existing objects according to the option specified:  ALL            All objects (default setting). OBSOLETE    All objects with a date older than the date of the object in the load file. EXCEPT     All objects except those with a date newer than the date of the object in the load file.	LOAD LOADALL
TRANSFER	Set Transfer mode.  The data is read and written in Transfer format. For valid options, see <a href="#">Keyword Explanation of transfer-options</a> .	UNLOAD LOAD SCAN
NOREPORT	Specifies the report file setting:  No data is recorded to a report file. This is the default setting for the FIND and FINDLIB commands.	
NEWREPORT	Specifies the report file setting:  Report data is recorded and written to a Natural text member stored in the Workplan library. An existing file will be overwritten.	
REPORT	Specifies the report file setting:  Report data is recorded and written to a Natural text member stored in the Workplan library. This is the default setting for the commands UNLOAD, LOAD, LOADALL, SCAN, SCANALL and DELETE.	
BATCHREPORT	Specifies the report setting for batch processing or when using the OBJHAPI Application Programming Interface:  Report data is either written to SYSOUT or output on the screen respectively (report data is <i>not</i> written to a file).	
NORESTART	No restart information is written to a file.	LOAD
RESTART	Restart information is written to a Natural text member stored in the Workplan library.	LOAD
NUMBERPROCESS	Specifies the number of objects to be processed.  The LOAD or SCAN command stops execution after the number specified.	LOAD SCAN

Option	Explanation	Restricted to Command
FIXEDLENGTH	<p>Sets the format of the unload work file to a maximum record length of fixed size.</p> <p>Every data record contains 256 bytes if written in internal format, or 100 bytes in Transfer format.</p>	UNLOAD
FDIC	<p>Specifies the system file FDIC to be used for processing:</p> <p>the database ID (<i>dbid</i>), file number (<i>fnr</i>), password (<i>password</i>) and cipher code (<i>cipher</i>) of the Adabas file.</p> <p>If no values (or 0) are specified, the current FDIC system file is used.</p>	UNLOAD LOAD DELETE
FSEC	<p>Specifies the system file FSEC to be used for processing:</p> <p>the database ID (<i>dbid</i>), file number (<i>fnr</i>), password (<i>password</i>) and cipher code (<i>cipher</i>) of the Adabas file.</p> <p>If no values (or 0) are specified, the current FSEC system file is used.</p>	UNLOAD LOAD DELETE
WORKFILETYPE or WFTYPE	<p>The work file type of Natural Work File 1 when data is read and written to the work file:</p> <p>DEFAULT Default binary work file.</p> <p>UTF-8 Unicode/UTF-F8 encoded binary work file.</p> <p>UTF-8 only applies to the unload function and if <b>TRANSFER</b> is specified.</p> <p>If UTF-8 is specified, you cannot use the options <b>CONVERSION-TABLE</b>, <b>SUBSTITUTE</b> and <b>INCORPORATE-FREE-RULES</b>.</p> <p>If <b>WORKFILETYPE</b> has not been specified, the current type is used.</p>	UNLOAD
PC NEWPC	<p>Only applies if Entire Connection is installed.</p> <p>Writes data to or reads data from an Entire Connection work file. <i>file-name</i> denotes the complete path name assigned to the Entire Connection work file. If your system environment does not accept a backslash (\) separator, use a slash (/) instead. If you do not specify <i>file-name</i>, Entire Connection prompts you for the name of a work file.</p> <p>If <b>NEWPC</b> is specified, the data unloaded overwrites the contents of the existing work file or fills a new work file from the top. Otherwise, the data is appended.</p> <p>See also <a href="#">Work File Assignment</a> in <i>Work Files</i>.</p>	UNLOAD LOAD SCAN

Option	Explanation	Restricted to Command
PCCOMMANDFILENUMBER or PCCOM or PCCFN	<p>Only applies if Entire Connection is installed.</p> <p>Specifies the number of the work file that is used for processing Entire Connection commands.</p> <p>The default value is 8 for Work File 8, which must be defined as Entire Connection work file.</p> <p>See also <a href="#">Work File Assignment</a> in <i>Work Files</i>.</p>	UNLOAD LOAD SCAN

The keywords and the variable values (if relevant) of *transfer-options* and *internal-format-options* are explained in the following section:

- [Keyword Explanation of transfer-options](#)
- [Keyword Explanation of internal-format-options](#)

### Keyword Explanation of transfer-options

When using the TRANSFER keyword, you can specify the following options:

Option	Explanation	Restricted to Command
CONVERSION-TABLE	<p>Converts data processed in Transfer format by using either of the following conversion tables:</p> <p>SYSTEM-TABLE: The internal Natural conversion table.</p> <p>USER-TABLE: A user-defined conversion table if <i>conversion-program</i> has been specified. This program must be stored in the library SYSOBJH or one of its steplib; see the example programs OTNCONAE and OTNCONEA in the library SYSOBJH.</p>	UNLOAD LOAD SCAN
SUBSTITUTE	<p>Replaces line references by labels during the unload in Transfer format.</p> <p>This option only applies if your source-code line numbers are used for statement references. If so, the line numbers of referenced lines and the line number references are replaced by labels. The sources are not modified in the database.</p>	UNLOAD
INCLUDE-LINE-NUMBERS	Transfers line numbers during the unload in Transfer format. By default, line numbers in Natural objects are <i>not</i> unloaded.	UNLOAD
UPPERCASE-TRANSLATION	Translates any source code into upper case during the load in Transfer format.	LOAD

Option	Explanation	Restricted to Command
	By default, source code in Natural objects is <i>not</i> translated.	
INCORPORATE - FREE - RULES	Incorporates source text of Predict free rules associated with a map into a map source during the unload in Transfer format if Predict is installed.	UNLOAD
LOAD - CODE - PAGE	<p>Specifies the code page to be used for converting object sources encoded in Unicode/UTF-8 (Universal Transformation Format, 8-bit form).</p> <p>If you use this option, all object sources unloaded into a work file in UTF-8, will be converted with the specified code page when they are loaded into a work file.</p> <p>If you specify *CODEPAGE as <i>code - page - name</i> or if <i>code - page - name</i> is not specified, the value assigned to the system variable *CODEPAGE is used (see the <i>System Variables</i> documentation).</p> <p>If LOAD - CODE - PAGE is specified, you cannot use the options CONVERSION - TABLE and UPPERCASE - TRANSLATION.</p>	LOAD LOADALL
DA - FORMAT	Specifies format conversion of data area sources: see <a href="#">Data area format</a> in <i>Transfer Options in Settings</i> .	UNLOAD LOAD

### Keyword Explanation of internal-format-options

When using *internal-format-options*, you can specify the following:

Option	Explanation	Restricted to Command				
XREF	<p>Only applies if Predict is installed.</p> <p>Loads or unloads XRef data of cataloged Natural objects. You can specify one of the following values:</p> <table border="1" data-bbox="446 1453 1323 1747"> <tbody> <tr> <td>ON</td> <td> <p>UNLOAD: Unloads cataloged objects and their cross-reference data (if any).</p> <p>LOAD: Loads cataloged objects and their cross-reference data if cross-references exist in the work file.</p> </td> </tr> <tr> <td>OFF</td> <td>No XRef data is processed. This is the default.</td> </tr> </tbody> </table>	ON	<p>UNLOAD: Unloads cataloged objects and their cross-reference data (if any).</p> <p>LOAD: Loads cataloged objects and their cross-reference data if cross-references exist in the work file.</p>	OFF	No XRef data is processed. This is the default.	LOAD UNLOAD
ON	<p>UNLOAD: Unloads cataloged objects and their cross-reference data (if any).</p> <p>LOAD: Loads cataloged objects and their cross-reference data if cross-references exist in the work file.</p>					
OFF	No XRef data is processed. This is the default.					

Option	Explanation	Restricted to Command
	<p>DOC</p> <p>Only applies to LOAD.</p> <p>Loads cataloged objects only if Predict entries exist for the objects in the FDIC system file.</p> <hr/> <p>FORCE</p> <p>Only applies to LOAD.</p> <p>Loads cataloged objects and their cross-reference data only if cross-references exist in the work file and if Predict entries exist for the objects in the FDIC system file.</p> <hr/> <p>SPECIAL</p> <p>Only applies to LOAD.</p> <p>Loads cataloged objects and their cross-reference data (if any).</p>	
DELETEALLOWED	Processes delete instructions from work files created by the utility NATUNLD when loading objects in internal format.	LOAD
NOSYMBOLTABLE	<p>Unloads cataloged Natural library objects without their corresponding internal Natural symbol tables.</p> <p>This will reduce the amount of disk storage required. However, this is only useful for a production environment, as several application development functions which require the symbol tables will then not be available; in addition, the profile parameter RECAT=ON (see the <i>Parameter Reference</i> documentation) will not apply.</p>	UNLOAD
VERSIONCHECK	<p>Checks the Natural version of the cataloged object to be loaded. The Natural version under which the objects were cataloged and written to the work file is compared with the current Natural version. Objects cataloged under a Natural version higher than the current one will be rejected.</p> <p>VERSIONCHECK can only be used if data is loaded in internal format, that is, if the <b>TRANSFER</b> option is <i>not</i> specified.</p>	LOAD

# 46

## Examples of Using Direct Commands

---

- Unloading Objects for the Same Platform ..... 246
- Unloading Objects for Different Platforms ..... 247
- Loading Objects in Internal Format ..... 247
- Loading Objects in Transfer Format ..... 248

This section provides examples for using Object Handler direct commands.



**Tip:** For additional examples, you can view the command generated for an Object Handler function. This command is automatically displayed when you use a wizard. In advanced-user mode, you can activate the display of the command by either entering the Object Handler command `SET ADVANCEDCMD ON` or setting the parameter `Display-Cmd-in-Advanced-Mode` to `Y (Yes)` in the Object Handler profile (see also [Profile Settings](#)).

## Unloading Objects for the Same Platform

---

This section contains examples of how to unload objects in internal format to a work file in order to load them on the same platform, within either a local mainframe, UNIX, OpenVMS or Windows environment:

- Unload all Natural programming objects (source objects only) from library ABC:

```
UNLOAD * LIB ABC OBJTYPE N SCKIND S
```

- Unload all Natural programming objects (cataloged objects only) from library ABC:

```
UNLOAD * LIB ABC OBJTYPE N SCKIND C
```

- Unload all Natural programming objects (cataloged objects and source objects) from library ABC:

```
UNLOAD * LIB ABC OBJTYPE N SCKIND A
```

- Unload all Natural programming objects (source objects only) from library ABC to load in library ABCNEW:

```
UNLOAD * LIB ABC OBJTYPE N SCKIND S WITH NEWLIBRARY ABCNEW
```

- On a mainframe: Unload all DDMs whose names start with EMP and which point to database 88:

```
UNLOAD EMP* LIB * OBJTYPE D DDMDBID 88
```

- On UNIX, OpenVMS or Windows: Unload all DDMs whose names start with EMP and which point to database 88:

```
UNLOAD EMP* LIB * OBJTYPE N NATTYPE V DDMDBID 88
```

- On UNIX, OpenVMS or Windows: Unload all DDMs whose names start with EMP from library VLIB to load in library VLIBNEW:

```
UNLOAD EMP* LIB VLIB OBJTYPE N NATTYPE V WITH NEWLIBRARY VLIBNEW
```

- Unload all user-defined error messages from library `ERRLIB` to load in library `NEWERR`:

```
UNLOAD * LIB ERRLIB OBJTYPE E SLKIND A WITH NEWLIBRARY NEWERR
```

- On Windows: Unload all Natural programming objects (cataloged objects and source objects) from library `ABC` to a portable work file on a PC:

```
UNLOAD * LIB ABC OBJTYPE N WHERE WORKFILE C:\WF1.SAG WORKFILETYPE PORTABLE
```

or

```
UNLOAD * LIB ABC OBJTYPE N WHERE WORK C:\WF1.SAG WFT P
```

## Unloading Objects for Different Platforms

This section contains command examples of how to unload objects in Transfer format to a work file in order to load them on a different platform such as unloading in a mainframe and loading in a UNIX, an OpenVMS or a Windows environment.

- Unload all Natural programming objects (source objects only) from library `ABC`:

```
UNLOAD * LIB ABC OBJTYPE N WHERE TRANSFER
```

- Unload all Natural programming objects (source objects only) and user-defined error messages from library `ABC`:

```
UNLOAD * LIB ABC WHERE TRANSFER
```

- Unload all Natural programming objects (source objects only) from library `ABC` with fixed record length:

```
UNLOAD * LIB ABC OBJTYPE N WHERE TRANSFER FIXEDLENGTH
```

## Loading Objects in Internal Format

This section contains command examples of how to load objects from a work file in internal format.

- Load all objects to library `LIBNEW` and replace any that already exist:

```
LOADALL WITH NEWL LIBNEW WHERE REPLACE ALL
```

- Load all object with target library `TGTLIB` to the new target library `NEWTGT`:

```
LOAD * LIB TGTLIB WITH NEWLIBRARY NEWTGT
```

- Load the user-defined error messages 1000 to 1500 from library ERRLIB only:

```
LOAD * LIB ERRLIB OBJTYPE E FMNUM 1000 TONUM 1500
```

## Loading Objects in Transfer Format

---

This section contains command examples of how to load objects from a work file in Transfer format.

- Load all objects to library LIBNEW and replace any that already exist:

```
LOADALL WITH NEWL LIBNEW WHERE TRANSFER REPLACE ALL
```

- Load all object with target library TGTLIB to new target library NEWTGT:

```
LOAD * LIB TGTLIB WITH NEWLIBRARY NEWTGT WHERE TRANSFER
```

# 47

## Commands for Navigation and Special Functions

The Object Handler commands in CUI (character user interface) environments are mainly provided for navigation purpose and special function settings such as specifying trace files.

An Object Handler command is entered in the Command line of any Object Handler screen. If you want to execute a Natural system command from an Object Handler screen, enter two slashes (//) before the command. Note that any Natural system command terminates the Object Handler.

### ▶ To invoke the Commands menu of the Object Handler

- Choose PF10 (Cmds).

Or:

On any Object Handler screen, in the Command line, enter the following:

```
CMDS
```

The Object Handler commands are listed below. An underlined portion of a keyword represents an acceptable abbreviation, *Sub* denotes subcommand.

Command	Sub 1	Sub 2	Explanation
CANCEL			Cancels the current function and displays the Object Handler <b>Main Menu</b> .
<u>C</u> HANGE	<u>W</u> ORKPLAN	<u>L</u> IBRARY	Invokes the administration function and displays a screen where you can change the Workplan library.
<u>C</u> LEAR			Resets the current contents of the input fields in the map to the default values.
<u>C</u> MDS			Invokes the <b>Commands</b> screen.
or			

Command	Sub 1	Sub 2	Explanation
COMMANDS			
BYE			Terminates the Object Handler.
EXIT			
QUIT			
.			
FIN			Terminates the Object Handler and ends the Natural session.
GO	HOME		Displays the Object Handler <b>Main Menu</b> .
GO	UNLOAD		Invokes the unload function.
GO	UNLOAD	END	Ends the current unload function.
		ERROR	Invokes the unload function for Natural system error messages.
		DDM	Invokes the unload function for DDMs.
		FDT	Invokes the unload function for FDTs.
		LIBRARY	Invokes the unload function for Natural library objects.
		NCP	Invokes the unload function for Natural command processor sources.
		RELATED	Invokes the unload function for Natural-related objects.
		SELECTION or LIST	Displays a screen where you can enter or select the SELECTION or LIST Workplan to be used for the unload function.
GO	LOAD		Invokes the load function.
GO	LOAD	ALL	Invokes the load function for all objects contained in the work file.
		END	Ends the current load function.
		ERROR	Invokes the load function for Natural system error messages.
		DDM	Invokes the load function for DDMs.
		FDT	Invokes the load function for FDTs.
		LIBRARY	Invokes the load function for Natural library objects.
		NCP	Invokes the load function for Natural command processor sources.
		RELATED	Invokes the load function for Natural-related objects.
SELECTION or LIST	Displays a screen where you can enter or select the SELECTION or LIST Workplan to be used for the load function.		
GO	RESTART		Displays a screen where you can specify the text member to be used for the restart load function.
GO	SCAN		Invokes the scan function.

Command	Sub 1	Sub 2	Explanation
G <u>O</u>	S <u>CAN</u>	A <u>LL</u>	Invokes the scan function for all objects contained in the work file.
		E <u>ND</u>	Ends the current scan function.
		E <u>RROR</u>	Invokes the scan function for Natural system error messages.
		D <u>DM</u>	Invokes the scan function for DDMs.
		F <u>DT</u>	Invokes the scan function for FDTs.
		L <u>IBRARY</u>	Invokes the scan function for Natural library objects.
		N <u>CP</u>	Invokes the scan function for Natural command processor sources.
		R <u>ELATED</u>	Invokes the scan function for Natural-related objects.
G <u>O</u>	S <u>ELECTION</u> or L <u>IST</u>		Displays a screen where you can enter or select the SELECTION or LIST Workplan to be used for the scan function.
G <u>O</u>	A <u>DMIN</u>		Invokes the administration function.
G <u>O</u>	A <u>DMIN</u>	C <u>HANGE</u>	Displays a screen where you can change the Workplan library.
		C <u>REATE</u>	Opens a menu with which you can create a Workplan.
		L <u>IST</u>	Generates a list of Workplans available in the Workplan library.
G <u>O</u>	V <u>IEW</u>		Invokes the view function.
G <u>O</u>	V <u>IEW</u>	D <u>DM</u>	Invokes the view function for DDMs.
		E <u>RROR</u>	Invokes the view function for Natural system error messages.
		F <u>DT</u>	Invokes the view function for FDTs.
		L <u>IBRARY</u>	Invokes the view function for Natural library objects.
		N <u>CP</u>	Invokes the view function for Natural command processor sources.
		R <u>ELATED</u>	Invokes the view function for Natural-related objects.
G <u>O</u>	F <u>IND</u>		Invokes the find function.
G <u>O</u>	F <u>IND</u>	D <u>DM</u>	Invokes the find function for DDMs.
		E <u>RROR</u>	Invokes the find function for Natural system error messages.
		F <u>DT</u>	Invokes the find function for FDTs.
		L <u>IBRARY</u>	Invokes the find function for Natural library objects.
		N <u>CP</u>	Invokes the find function for Natural command processor sources.
		R <u>ELATED</u>	Invokes the find function for Natural-related objects.
		S <u>ELECTION</u> or L <u>IST</u>	Displays a screen where you can enter or select the SELECTION or LIST Workplan to be used for the find function.
H <u>ELP</u>			Invokes the Object Handler help function.

Command	Sub 1	Sub 2	Explanation
INIT			Reinitializes the Object Handler utility.
READ	<u>PROFILE</u>		Updates Object Handler settings as defined in the text member PROFILE (see also <i>Profile Settings</i> ).
SET	<u>ADVANCEDCMD</u>	ON	Activates the display of commands generated by the Object Handler in advanced-user mode.
		OFF	Deactivates the display of commands generated by the Object Handler in advanced-user mode.
	<u>EXECUTIONMSG</u>	ON	Activates a window that displays the processing status.
		OFF	Deactivates a window that displays the processing status.
	<u>FREE</u>	ON	Activates free format editing.
		OFF	Deactivates free format editing.
	<u>TRACE</u>	ON	Activates trace mode: a trace of each Object Handler action is output to the screen.
		OFF	Deactivates trace mode.
		<u>WORKFILE</u>	Activates trace mode: a trace of each Object Handler action is output to the Natural text member stored in the Workplan library.
	<u>TRACEFILE</u>		Displays a screen where you can specify the Natural text member in the Workplan library to be used for the trace.
SETTINGS			Displays a screen where you can specify the unload, load or scan settings.
<u>SHOW</u> or <u>DISPLAY</u>	<u>LAST</u>	<u>MESSAGE</u>	Displays the last interface return code and message issued by the processing interface of the Object Handler.
		<u>RESULT</u>	Displays the last result issued by the processing interface of the Object Handler.
	<u>PROFILE</u>		Displays the text member PROFILES (if available).
	<u>REPORT</u>		Displays the report created last.
	<u>STATISTICS</u>		Displays statistics information about the objects processed.
	<u>STATUS</u>		Displays the current Object Handler status (contents of global variables).
	<u>TRACE</u>	<u>FILE</u>	Displays the Natural text member in the Workplan library that contains the trace.

# 48

## Batch Condition Codes and User Exit Routines

---

▪ Condition Codes Returned in Batch .....	254
▪ Applying User Exit Routines .....	254
▪ User Exit Routines Available .....	255

This section describes the condition codes returned for Object Handler functions in batch mode and the user exit routines available for function processing.

## Condition Codes Returned in Batch

---

Object Handler processing in batch mode terminates with one of the following condition codes:

Condition Code	Explanation
0	Object Handler process terminated successfully.
30	An internal Object Handler error occurred.
40	An error was detected in the Object Handler command.
50	An error occurred during Object Handler processing.
60	A Natural Security error occurred during Object Handler processing.
99	A Natural error occurred during Object Handler processing.

## Applying User Exit Routines

---

The Object Handler user exit routines are supplied as source objects in the Natural system library SYSOBJH. These source objects are named SRC-EX $nn$ , where  $nn$  denotes the number of the user exit routine.

### ▶ To activate a user exit routine

- CATALOG or STOW source object SRC-EX $nn$  under the name OBJHEX $nn$  in the Natural system library SYSOBJH.

Different names are used to guarantee that the source object (possibly modified according to your requirements) and the cataloged object of the user exit routine are not overwritten by an update installation.

For detailed descriptions of the user exit routines, see the source objects of SRC-EX $nn$  in the library SYSOBJH.

---

## User Exit Routines Available

---

The following user exit routines are available:

- [OBJHEX01 for Processing Failures](#)
- [OBJHEX02 for Object Rejection](#)
- [OBJHEX03 for Default Option Values](#)

### **OBJHEX01 for Processing Failures**

Whenever a condition code is set to a value greater than 0 (zero) in batch mode, the user exit routine OBJHEX01 (if available) will be invoked before the Object Handler stops processing. With this user exit routine, you can specify whether to continue or terminate Object Handler processing. In the case of termination, you can change the condition code. For further details, see the source of the user exit routine SRC-EX01 in the Natural system library SYSOBJH.

### **OBJHEX02 for Object Rejection**

If the Object Handler load function was executed successfully in batch mode (with Condition Code 0), but one or more objects were rejected during loading (for example, not replaced), before the Object Handler stops processing, the user exit routine OBJHEX02 (if available) is invoked. With OBJHEX02, you can specify whether to continue or terminate Object Handler processing. In the case of termination, you can set a condition code. For further details, see the source of the user exit routine SRC-EX02 in the Natural system library SYSOBJH.

### **OBJHEX03 for Default Option Values**

You can apply user exit routine OBJHEX03 to set default options for processing Object Handler commands. This user exit is invoked before an Object Handler command is processed. For further details, see the source object of the user exit routine SRC-EX03 in the Natural system library SYSOBJH.



# 49 Tools

---

▪ Status .....	258
▪ Last Result .....	258
▪ Traces .....	258
▪ Reports .....	259

The Object Handler provides special features to display status information and reports and to check or modify trace settings.

## Status

---

Displays the Object Handler functions currently used, the user environment, the Workplan library and the setting of the trace option described below.

### ▶ To display the status

- In the Command line of any Object Handler screen, enter the following:

```
SHOW STATUS
```

See also the [SHOW](#) command described in *Commands for Navigation and Special Functions* in the section *Direct Commands*.

## Last Result

---

Displays the last internal command issued by the processing interface of the Object Handler and possible return codes and messages.

### ▶ To display the last result

- In the Command line of any Object Handler screen, enter the following:

```
SHOW LAST RESULT
```

See also the [SHOW](#) command described in *Commands for Navigation and Special Functions* in the section *Direct Commands*.

## Traces

---

Activates or deactivates the trace function. Traces record internal Object Handler program flows to provide control information for error diagnoses. The trace option is set off by default.

### ▶ To change the setting

- Use the command [SET TRACE](#) as described in *Commands for Navigation and Special Functions* in the section *Direct Commands*.

---

## Reports

---

Lists the objects loaded, unloaded or scanned, and records errors that may interrupt processing. See also *Work File Options* in the section *Settings*. The report option is set on by default and is displayed after the unload, load or scan function has been executed.

▶ **To display the contents of the latest report file**

- In the Command line of any Object Handler screen, enter the following:

```
SHOW REPORT
```

See also the [SHOW](#) command described in *Commands for Navigation and Special Functions* in the section *Direct Commands*.



# 50

## Profile Settings

---

- Editing PROFILES ..... 262
- Parameter Specification ..... 263

Natural provides the option to customize the default settings of your current Object Handler utility environment. For this purpose, Natural provides the text member PROFILE in the Natural system library SYSOBJH. PROFILE is used to specify environment-specific default values for flags and options that appear when entering the corresponding Object Handler screens.

▶ **To activate individual profile settings**

- 1 Save the text object PROFILE under the name PROFILES in the library SYSOBJH.
- 2 In the text object PROFILES, set the parameters required and save the modifications.
- 3 Invoke the Object Handler to activate PROFILES.

If activated, the following Natural confirmation message appears: Defaults from SYSOBJH profile (PROFILES) read.

▶ **To deactivate individual profile settings**

- Delete the text object PROFILES from the library SYSOBJH.

## Editing PROFILES

---

The following section provides general instructions for editing the text object PROFILES.

Empty lines or lines that start with an asterisk (\*) or /\* are considered comments and therefore ignored.

Each line is limited to 90 bytes.

This section covers the following topic:

- [General and User-Specific Settings](#)

### General and User-Specific Settings

You can specify general profile settings that apply to all users, and individual profile settings that apply only to single users.

Parameters used for general settings are entered within the general block delimited by [General-Start] and [General-End].

Parameters used for individual settings are entered within any number of user-specific blocks, each of which is delimited by [User-Start *UID-EXAM*] and [User-End *UID-EXAM*] where *UID-EXAM* represents the user ID as assigned by the Natural system variable \*USER (see the *System Variables*

documentation). Each user-specific block only applies to the user whose ID is entered within the block delimiters.

Parameter settings made in a user-specific block override respective parameter settings made in the general block.

### Example of a User-Specific Block

```
[User-Start SAG]
  Advanced-Mode           Y
  Option-Write-Report     N
[User-End SAG]
```

In the example above, regardless of the corresponding settings made in the general block, for User SAG, the field **Advanced mode** appears activated (selected) in the **Main Menu** and the field **Write report** appears deactivated (not selected) on the Object Handler **Options** screen.

## Parameter Specification

The table below lists the parameters contained in PROFILE, the possible values that can be entered and the Object Handler functions to which the parameters apply. In addition, the table provides a brief description of the parameters or a reference to the corresponding Object Handler documentation section. Default parameter values are underlined.

Parameter	Possible Values	Function	Description/ Documentation Section
Advanced-Mode	<u>N</u> (No) or Y (Yes)	Unload Load Scan	<i>Advanced User</i> in <i>Functions</i>
Display-Cmd-in-Advanced-Mode	<u>N</u> or Y	Unload Load Scan	Displays the Object Handler command generated for a function executed in advanced-user mode.
Display-ExecutionMsg	<u>N</u> or Y	Unload Load Scan	Activates a window that displays the processing status.
Workplan-Library	<u>WORKPLAN</u> or any other Workplan library	Unload Load Scan Administration	<i>Workplans</i> and <i>Change the Workplan Library</i> in <i>Administration</i>
Workplan-Library-DBID	<u>Q</u> (current FNAT/FUSER) or	Unload Load Scan	<i>Change the Workplan Library</i>

Parameter	Possible Values	Function	Description/ Documentation Section
	any other Adabas database ID (DBID)	Administration	
Workplan-Library-FNR	Q (current FNAT/FUSER) or any other Adabas file number (FNR)	Unload Load Scan Administration	<a href="#">Change the Workplan Library</a>
TRACE	N or Y	Unload Load Scan	<a href="#">Traces in Tools</a>
TRACE-TARGET	S (Screen) or W (Work file)	Unload Load Scan	<a href="#">Traces</a>
Option-Replace	N or Y or O (Obsolete) E (Except)	Load	<a href="#">Replace Options in Settings</a>
Option-TRANSFER-FORMAT	N or Y	Unload Load Scan	<a href="#">Work File Format in Work Files</a>
Option-TR-INCLUDE-LINE-NUMBERS	N or Y	Unload	<a href="#">Include line numbers in Transfer Options</a>
Option-TR-SUBSTITUTE	N or Y	Unload	<a href="#">Substitute line numbers in Transfer Options</a>
Option-TR-TRANSLATE-TO-UPPER	N or Y	Load	<a href="#">Translate to upper case in Set Additional Options</a>
Option-TR-USE-CONVERSION-TABLE	N or S (System table) or U (User table)	Unload Load	<a href="#">Use conversion table in Transfer Options</a>
Option-TR-CONV-TABLE-NAME-LOAD	OTNCONEA or a user-written subprogram	Load	<a href="#">Use conversion table</a>
Option-TR-CONV-TABLE-NAME-UNLD	OTNCONAE or	Unload	<a href="#">Use conversion table</a>

Parameter	Possible Values	Function	Description/ Documentation Section
	a user-written subprogram		
Option-Write-Report	N or Y	Unload Load Scan	<a href="#">Reports</a> in <i>Tools</i>  <a href="#">Write report</a> in <i>Work File and Report Options</i>
Option-Write-Restart-Info	N or Y	Load	<a href="#">Write restart information</a> in <i>Work File and Report Options</i>
USE-OPTION-WORKPLAN	N or Y	Unload Load Scan	<a href="#">Workplans</a>
OPTION-WORKPLAN-Name	OPTIONWP or any other Workplan of the type OPTION	Unload Load Scan	<a href="#">Workplans</a>
USE-PARAMETER-WORKPLAN	N or Y	Unload Load	<a href="#">Workplans</a>
PARAMETER-WORKPLAN-Name	PARAWPLN or any other Workplan of the type PARAMETER	Unload Load	<a href="#">Workplans</a>
WORK-FILE-1-Name	The work file used for the PC file (Work File 7).	Unload Load Scan	<a href="#">Work Files</a>  Only applies if Entire Connection is installed.
Report-File-Name	The name of the text member in the Workplan library to be used for the report.	Unload Load Scan	<a href="#">Write report</a> in <i>Work File and Report Options</i>
Restart-File-Name	The name of the text member in the Workplan library to be used for the restart information.	Load	<a href="#">Write restart information</a> in <i>Work File and Report Options</i>
Trace-File-Name	The name of the text member in the Workplan library to be used for the trace.	All functions	<a href="#">Traces</a> in <i>Tools</i>



# 51 Migration from NATUNLD/NATLOAD and SYSTRANS to the Object Handler

---

- Converting Individual Commands ..... 268
- Processing Commands with a User Exit Routine ..... 270
- Processing SYSTRANS Commands with OBJHAPI ..... 270
- Unsupported SYSTRANS Options ..... 271

You can migrate from the old utilities NATUNLD/NATLOAD and SYSTRANS to the Object Handler by using the two methods described in this section.

## Converting Individual Commands

---

You can convert NATUNLD/NATLOAD or SYSTRANS direct commands to the corresponding Object Handler commands by using the Object Handler commands provided for migration. These migration commands automatically convert the command syntax used by the old utilities to the command syntax used by the Object Handler.

### ► To convert a single command

- 1 Use one of the following Object Handler direct commands:

```
NATUNLD
```

followed by a NATUNLD direct command.

Or:

```
NATLOAD
```

followed by a NATLOAD direct command.

Or:

```
SYSTRANS
```

followed by a SYSTRANS direct command.

The specified utility command is converted to the corresponding Object Handler command.

- 2 Specify any subsequent command for the Object Handler in the syntax that applies to the utility NATUNLD, NATLOAD or SYSTRANS respectively.

The syntax of this utility remains valid for the duration of the Object Handler session.

**Example of a NATUNLD Command:**

The following is an example of two consecutive NATUNLD utility commands and their corresponding Object Handler commands.

Old NATUNLD commands:	NATUNLD ALL * FM LIB1 TO LIB2
	ALL PG* FM LIB2
New Object Handler command:	SYSOBJH NATUNLD ALL * FM LIB1 TO LIB2
Subsequent Object Handler command in NATUNLD syntax:	ALL PG* FM LIB2

**Example of a SYSTRANS Command:**

The following is an example of two consecutive SYSTRANS utility commands and their corresponding Object Handler commands.

Old SYSTRANS commands:	TRANSCMD EXECUTE UNLOAD N FROM LIB1 NAME ETID
	END
New Object Handler command:	SYSOBJH SYSTRANS EXECUTE UNLOAD N FROM LIB1 NAME ETID END
Subsequent Object Handler command in SYSTRANS syntax:	END

**Example of SYSTRANS Batch Processing:**

The following is an example of processing a SYSTRANS utility command in batch by using map input data, and the corresponding Object Handler command and input data.

Old SYSTRANS batch sequence:

```
SYSTRANS
U
N,N,N,Y,N,N,N,N
N
SRCLIB1,PGM1,*,TGTLIB1
```

New Object Handler batch sequence:

```
SYSOBJH SYSTRANS
U
N,N,N,Y,N,N,N,N
N
SRCLIB1,PGM1,*,TGTLIB1
```

## Processing Commands with a User Exit Routine

---

As an alternative to redefining each single utility command, you can specify in a user exit routine that utility commands call `SYSOBJH` for function execution. This routine determines whether or not a direct commands issued to `NATUNLD/NATLOD` or `SYSTRANS` is forwarded as an `SYSOBJH` command to the Object Handler.

### ▶ To activate the user exit routine

#### 1 For `NATUNLD/NATLOAD`:

`SAVE` source object `U-S-EX03` under the name `UNLDEX03` in the Natural system library `SYSUNLD`. The source object is supplied in `SYSUNLD`.

For `SYSTRANS`:

`SAVE` source object `TRA-E2-S` under the name `TRA-EX-2` in the Natural system library `SYSTRANS`. The source object is supplied in `SYSTRANS`.

#### 2 Open the source of `UNLDEX03` or `TRA-EX-2` respectively and set `USE-SYSOBJH` to `Y` (this is the default value entered). `N` will deactivate the user exit routine.

You can define conditions for use such as when the Object Handler is to be used instead of an old utility or who is authorized to use the Object Handler.

#### 3 `CATALOG` or `STOW` the source object.

## Processing `SYSTRANS` Commands with `OBJHAPI`

---

You can use the `OBJHAPI` Application Programming Interface (supplied in the Natural system library `SYSOBJH`) to execute an Object Handler command in the syntax of the `SYSTRANS` utility.

If you use `OBJHAPI` for this purpose, you have to specify the parameter `P-EXTENSIONS-EXEC-SYSTRANS-CMD` in the program that invokes `OBJHAPI`. For details, see the example program `DOC-API` supplied in the library `SYSOBJH`.

## Unsupported SYSTRANS Options

---

The Object Handler does not support the following SYSTRANS direct command options:  
WORK-FILE-INPUT, SPECIAL-CONVERSION, RULE-LOAD **and** UNLOAD-RULES.



# 52

## Recording Utility

---

- Purpose of Recording ..... 274
- Data and Functions Recorded ..... 274
- Recording a Session ..... 275
- Playing Back a Recording ..... 276
- Manipulating a Recording ..... 278

With the Recording utility, you can record a Natural session and later play back the recorded session.

**Related Documentation:**

*Terminal Commands*

## Purpose of Recording

---

The Recording utility can be used for the following purposes:

■ **Demonstration**

Instead of having to type in several commands, such as input data by hand, you can play back a recorded sequence of keyboard actions to demonstrate a standard procedure.

■ **Application development**

When applying the same modifications to several objects (for example, programs or maps), you can use a recording to reduce the amount of work involved and at the same time ensure that the modifications are actually the same for all objects affected.

■ **Testing**

You can execute a standard testing procedure by simply playing back a recording.

■ **Quality control**

Before and after making changes to an application, you can play back a recording and compare the results of the two runs to make sure that certain things were not affected by the changes.

■ **User training**

You can incorporate the playback of recordings into training programs for users, to show them specific procedures. Also, you can record user keyboard actions in a session and then inform them of any errors they make or of ways to carry out actions more efficiently. The recording of user actions can also help you to detect any flaws in an application's user interface.

## Data and Functions Recorded

---

The Recording utility records the following:

- All input data and commands (including terminal commands) entered on the screen.
- Any function keys (PF keys) pressed.
- The current cursor position as contained in the system variable `*CURSOR` (see the *System Variables* documentation).

## Recording a Session

---

This section describes the steps required to activate and deactivate a recording.

- [Specifying Libraries](#)
- [Activating a Recording](#)
- [Deactivating a Recording](#)

### Specifying Libraries

▶ **To specify the library in which all subsequent recordings are to be stored**

- Enter the following terminal command:

```
%B=library-name
```

If you activate the recording process without having specified *library-name*, the name of the library in which the recording is stored is the same as the value of the system variable `*INIT-USER` (see the *System Variables* documentation) at the time when the recording process is activated.

When you log on to another library during a session being recorded, the library in which the recording is being stored remains the same (that is, either the one specified with `%B=` or the `*INIT-USER` library); this means that one recording can record keyboard actions across multiple applications.

### Activating a Recording

▶ **To activate a recording**

- Enter the following terminal command:

```
%Bname
```

All subsequent keyboard actions are recorded.

*name* denotes the name under which the data recorded are saved in source form as a Natural object of the type Recording. You can treat this source as any other Natural source (for example, delete it, copy it), except that you must not edit it: recordings contain binary data an editor will destroy.

*name* can only be specified once. If a recording object of the same name already exists in the library specified for recording, Natural returns the message `Error in recording activation`.

- ⚠ **Caution:** Any situation that leads to a backout transaction or rollback (for example, a non-activity timeout) while a recording is in progress, will delete part of the recording thus making the entire recording useless.

Terminal command `%Aname` included in a recording should be followed by terminal command `%B` as described in [Recording %A](#).

## Deactivating a Recording

### ▶ To deactivate a recording

- Enter the following terminal command:

```
%B
```

The recording has terminated.

## Playing Back a Recording

---

When a recording is played back, the sequence of, for example, commands and function keys is actually executed again.

The recording is independent of the terminal type; that is, a session recorded on one terminal can be played back on a terminal of another type. You can also play back a recording in batch mode; a recorded online session may, of course, react differently when played back in batch.

This section covers the following topics:

- [Step Mode and Background Mode](#)
- [Activating a Playback](#)
- [Interrupting a Playback](#)

### Step Mode and Background Mode

A recording can be played back in two modes: background mode and step mode.

In background mode, the entire recording is played back invisibly; that is, all keyboard actions of the recording are carried out without anything being displayed to you on the terminal screen during the execution of the recording. You cannot interrupt a recording that is played back in background mode, unless the recording contains the terminal command `%R` as explained in [Manipulating a Recording](#).

In step mode, a recording is played back step by step and all keyboard actions are displayed on the screen. By choosing any function key, you proceed from one step to the next. In step mode, it is also possible for you to interrupt the recording by pressing `CLEAR` as explained in [Interrupting a Playback](#).

By default, a recording is played back in background mode.

---

▶ **To set modes**

- 1 To activate step mode, enter the following terminal command:

```
%GON
```

- 2 To deactivate step mode and activate background mode, enter the following terminal command:

```
%GOFF
```

- 3 To toggle between step and background mode, enter the following terminal command:

```
%G
```

### Activating a Playback

▶ **To play back a recording**

- Enter the following terminal command:

```
%Aname
```

The recording saved under the specified name is executed again.

### Recording %Aname

If you issue the command `%Aname` while a session is being recorded, the recording specified with `%Aname` is not executed but the command `%Aname` is included into the object source that is being recorded. Thus, you can execute a recording from within another recording and concatenate a series of recording to one another. However, you cannot have nested recordings; the execution of the recording that contains the `%Aname` command stops after this command and is not resumed when the execution of `name` finishes. As a result, the data recorded after `%Aname` will never be played back. To avoid this, you should enter `%B` immediately after you have entered `%Aname` in a recording.

## Interrupting a Playback

### ▶ To interrupt a recording that is played back in step mode

- Press `CLEAR`.

Once you have interrupted a recording, you have the following options:

- You can continue your session normally from the point where you stopped the recording.
- You can insert additional keyboard actions into the recording: after you have pressed `CLEAR`, enter the command `%B` and all actions you perform are inserted into the source of the recording until you enter `%B` again. Then, the execution of the recording is resumed.
- You can alter the next step in the recording: after you have pressed `CLEAR`, enter the command `%R`, then enter the input data for the next step. The newly entered input data overwrite the input data for this step in the recorded source. When you press `ENTER`, this step is executed with the new input data and subsequently the execution of the recording is resumed.
- You can execute any helproutine: after you have pressed `CLEAR`, enter the command `%J` directly followed by the name of the desired helproutine. The helproutine is invoked and the execution of the recording is continued as soon as the execution of the helproutine ends.

## Manipulating a Recording

---

By recording the terminal command `%R`, you can manipulate a single step in a recording when it is played back. This applies in step mode and in background mode. In background mode, `%R` is the only way to interact with a recording that is being played back. Interaction, for example, may be required to provide an input option for sensitive data, such as passwords which are unknown at the time of the recording.

If the terminal command `%R` (redisplay last screen) has been recorded, the subsequent screen is open for user input when the recording is played back; that is, the input data for this screen are not taken from the recording but from what the user enters. Subsequently, the execution of the recording is continued.

# 53

## SYSAPI Utility - APIs of Natural Add-On Products

---

▪ Prerequisites .....	280
▪ Invoking and Terminating SYSAPI .....	280
▪ Listing API Example Programs .....	281
▪ Performing a Keyword Search .....	282
▪ Using an API Example Program .....	283

The utility SYSAPI is used to locate and test Application Programming Interfaces (APIs) provided by Natural add-on products such as Entire Output Management.

The API of a Natural add-on product is a Natural subprogram (cataloged object) that is used for accessing and possibly modifying data or performing services that are specific to an add-on product or a subcomponent.

The API of a Natural add-on product is supplied in the Natural library and/or system file provided for objects that are specific to a particular Natural add-on product. For instructions on using the API of a Natural add-on product, refer to the documentation of the respective add-on product.

For each API of a Natural add-on product, the utility SYSAPI provides a functional description, one example program and API-specific keywords.

### Related Topics:

- *Natural User Exits - Operations* documentation
- [SYSEXT - Natural Application Programming Interfaces - Utilities](#) documentation

## Prerequisites

---

- The appropriate Natural add-on product must be installed at your site.
- The version of the Natural add-on product installed must support the SYSAPI utility features.

## Invoking and Terminating SYSAPI

---

This section provides instructions for invoking and terminating the SYSAPI utility.

### ▶ To invoke the SYSAPI utility

- Enter the following system command:

```
SYSAPI
```

The SYSAPI menu appears which lists one or more Natural add-on products, each followed by one or more associated API groups.

Each API group represents a particular API feature provided for the Natural add-on product. An API group contains all example programs that relate to this feature.

You can select an API group for displaying all example programs that belong to this group as described in [Listing API Example Programs](#).



**Note:** In the **Command** line of any SYSAPI utility screen, you can enter any Natural system command.

▶ **To terminate SYSAPI**

- Press PF3 or PF12.

## Listing API Example Programs

You can obtain a list of all example programs available for a selected API group. You can shorten this list by specifying the name of an API example program or a particular range of names.

▶ **To list all API example programs**

- In the SYSAPI menu, in the input field next to the API group desired, enter any character or use cursor selection.

The SYSAPI list screen appears for the selected API group. The screen contains the **Example** column with the names of the API example programs, and the **Function** column with a brief functional description of the corresponding API.

The list is sorted in alphabetical order by name of example program. If required, press PF8 to scroll down one page in the list. Press PF6 to go to the beginning of the list.

▶ **To list a single example program or a particular range**

- 1 On the SYSAPI list screen, in the **Example** input field, enter any of the input values listed in the table below where *value* is any combination of one or more characters:

Input	Example Programs Selected
<i>value</i>	The example program that matches <i>value</i> .
*	All example programs. This is the default setting.
<i>value</i> * <i>value</i> ?	All example programs that match <i>value</i> combined with one or more asterisks (*) and/or one or more question marks (?) entered in any position and any order where:  asterisk (*) denotes any string of characters, question mark (?) denotes a single character.  Examples:  ABC* A?C*Z

- 2 Press ENTER.

The SYSAPI list screen displays the specified example program(s).

## Performing a Keyword Search

---

Keywords help you find the APIs relevant to your current task. You can use the keyword search function to list all keywords relevant to the selected API group or list API example programs by keyword(s).

### ▶ To list all keywords relevant to an API group

- In the **Command** line, enter KEYWORDS.

Or:

Press PF5.

The **List Keywords** window appears. It contains all keywords in alphabetical order that are specific to the API group.

If required, press PF8 to scroll down the list of keywords or press PF6 to go to the beginning of the list.

### ▶ To list example programs by keyword

- 1 On the SYSAPI list screen, in one or more of the input fields next to **Keywords**, enter a valid keyword. If required, press PF10 to clear the contents of all **Keywords** fields.

Or:

From the **List Keywords** window, choose a maximum of three keywords:

- Next to the keyword desired, enter any character to mark the keyword for selection and press ENTER.

The keywords selected are displayed in the **List Keywords** window under **Selected keywords**.

If required, press PF10 to clear the keywords listed under **Selected keywords** and repeat the keyword selection.

- Press PF3 to confirm the keyword selection.

The SYSAPI list screen appears with the selected keywords entered in the **Keywords** fields. Additionally, the example programs that contain the specified keywords are listed on the screen.

- 2 If required, in the **And/Or** field, enter an A or an O (the default setting is A) to combine keywords with a logical condition.
- 3 Press ENTER to perform the keyword search.

The example programs that contain the specified keyword(s) are listed on the screen.

## Using an API Example Program

The SYSAPI utility provides line commands that can be used to display, modify or execute an API example program. You enter a line command on the SYSAPI list screen, in the **Cmd** column next to the example program required. For a list of valid line commands, enter a question mark (?) in this column.

The following line commands are available on the SYSAPI list screen:

Line Command	Function
L	List source code.
E	Edit source code.
X	Execute program.



# 54

## SYSBPM Utility - Buffer Pool Management

---

The utility SYSBPM is used for managing buffer pools of the type Natural, DL/I or SORT.

SYSBPM provides statistical information on the current status of the buffer pool including the buffer pool cache and on the Natural objects loaded in the buffer pool and the BP cache. SYSBPM also provides administration functions.

For a general description of the Natural buffer pool, refer to the *Operations* documentation.

The buffer pool is defined with the macro NTBPI in the Natural parameter module, or with the corresponding dynamic profile parameter BPI, both of which are described in the *Parameter Reference* documentation. The type of buffer pool can be determined with the TYPE subparameter of the NTBPI macro as described in *Buffer Pool Initialization*.

- **Invoking and Operating SYSBPM**
- **List Objects**
- **Delete Objects**
- **Directory Information**
- **Hexadecimal Display**
- **Write to Work File**
- **Display Sorted Extract**
- **Buffer Pool Statistics**
- **BP Cache Statistics**
- **Select Buffer Pool**

- [Blacklist Maintenance](#)
- [Preload List Maintenance](#)
- [Performance Considerations](#)
- [SYSBPM Direct Commands](#)
- [Batch Processing](#)
- [Application Programming Interfaces](#)

 **Note:** In this documentation, the buffer pool is also referred to as “BP”.

# 55

## Invoking and Operating SYSBPM

---

▪ Invoking SYSBPM .....	288
▪ Online Help .....	289
▪ SYSBPM Main Menu - Fields, Functions and Commands .....	289
▪ SYSBPM in a z/OS Parallel Sysplex Environment .....	293

The functions of the SYSBPM utility always refer to Natural objects currently loaded in the Natural buffer pool and/or BP cache. Note that the buffer pool or BP cache only contains executed Natural objects that have been stowed or cataloged. Objects executed with the RUN command are not loaded into the buffer pool or BP cache.

You can choose a function code or a PF key from a SYSBPM menu to execute a SYSBPM function. Alternatively, you can use a **SYSBPM direct command** as described in the relevant section.

This section describes how to invoke the SYSBPM utility, obtain online help text and select functions by using the SYSBPM **Main Menu**. The functions are described in detail in the relevant sections of the SYSBPM documentation.

In addition, information is provided on using SYSBPM in a z/OS Parallel Sysplex environment.

## Invoking SYSBPM

### ▶ To invoke the SYSBPM utility

- Enter the following Natural system command:

```
SYSBPM
```

A SYSBPM **Main Menu** similar to the example below appears:

```

16:12:23          ***** NATURAL SYSBPM UTILITY *****          2002-08-27
BPNAME QA41GBP          - Main Menu -          Type Global Nat
BPPROP OFF          Loc DAEF QA41
          Preload QA41GBPL

          Object Functions          Object Pool Statistics

          L List Objects          A Buffer Pool
          D Delete Object          C BP Cache
          I Directory Information
          H Hexadecimal Display          Other Functions
          W Write to work file          S Select Buffer Pool
          X Display sorted extract          B Blacklist Maintenance
          ? Help          P Preload List Maintenance
          . Exit

Code .. _   Library ... * _____
Object .... * _____
DBID ..... 0____ FNR .. 0____ Object Pool ... B (B,C,*)
    
```

```

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help      Exit  Last      Flip      Canc

```

- In the SYSBPM **Main Menu**, specify the executed object(s) loaded in the buffer pool or BP cache by choosing either of the following options:
    - Complete the input fields as described in *SYSBPM Main Menu - Fields, Functions and Commands*.
- Or:
- In the Command line, enter a SYSBPM direct command as described in *SYSBPM Direct Commands*.

## Online Help

The online help function of SYSBPM provides information on **SYSBPM direct commands** (see the relevant section) or valid input values for fields that appear on SYSBPM screens.

### ▶ To invoke the online help function for SYSBPM direct commands

- On any SYSBPM screen, position the cursor in the Command line and press PF1 or enter a question mark (?).

The **Help** window appears with a list of all SYSBPM direct commands available.

### ▶ To invoke the online help function for a SYSBPM input field

- On any SYSBPM screen, position the cursor in any input field and press PF1 or enter a question mark (?).

The **Help** window appears for the relevant field with a list of all valid input values.

## SYSBPM Main Menu - Fields, Functions and Commands

This section covers the following topics:

- [Fields](#)
- [Functions](#)

▪ PF Keys and Direct Commands

**Fields**

The fields contained in the SYSBPM **Main Menu** are described in the following table:

Fields	Explanation
<b>BPNAME</b>	The name of the global buffer pool as specified with the profile parameter BPNAME. For a local buffer pool, no name but a blank field is displayed for BPNAME.  See also <i>BPNAME - Name of Natural Global Buffer Pool</i> in the <i>Parameter Reference</i> documentation.
<b>BPPROP</b>	The setting of the profile parameter BPPROP to control the propagation of changes to an object in a buffer pool.  See also <i>BPPROP - Global Buffer Pool Propagation</i> in the <i>Parameter Reference</i> documentation.
<b>Type</b>	The type of buffer pool, such as Global Nat, Local Nat, Global Sort or Global DL/I.
<b>Loc</b>	The location. Displays the host ID (in the example screen above: DAEF) and the subsystem ID (in the example screen above: QA41).
<b>Preload</b>	The name of a preload list if loaded.  See also <i>Preload List Maintenance</i> .
<b>Library</b>	The name of the library where the executed object is stored. You can specify a name or use asterisk (*) notation.  The default asterisk (*) selects all libraries.
<b>Object</b>	The name of the executed object loaded in the buffer pool. You can specify a name or use asterisk (*) notation.  The default asterisk (*) selects all objects.
<b>DBID</b>	The database ID (DBID) of the system file FNAT or FUSER where the executed object is stored and from where it is loaded.  If you specify 0 (zero; this is the default) as DBID, the specified object(s) will be selected regardless of their DBID. Any value other than 0 represents a particular DBID specification.
<b>FNR</b>	The file number (FNR) of the system file FNAT or FUSER where the executed object is stored and from where it is loaded.  If you specify 0 (zero; this is the default) as FNR, the specified object(s) will be selected regardless of their FNR. Any value other than 0 represents a particular FNR specification.
<b>Object Pool</b>	Applies to the functions <b>List Objects</b> , <b>Directory Information</b> , <b>Write to Work File</b> and <b>Display Sorted Extract</b> described in the following section.  Selects the type of object pool(s) to be used for these functions:  B Buffer pool.

Fields	Explanation	
	C	BP cache.
	*	Both buffer pool and BP cache. This is the default.
	The function <b>List Objects</b> generates a list of all objects that are loaded either in the buffer pool or in the BP cache. Objects loaded in the buffer pool are displayed first, and then objects in the BP cache.	
	The type of object pool entered determines the point at which the list starts. For example, if you enter a C the list starts with objects loaded in the BP cache. Scrolling up the list, you can display the objects loaded in the buffer pool. If no object is found in the BP cache, the list contains the objects of the buffer pool.	

**Functions**

The functions provided in the SYSBPM **Main Menu** are organized in three sections:

- The **Object Functions** section contains functions for displaying or manipulating objects in the buffer pool or BP cache.
- The **Object Pool Statistics** section contains functions for obtaining object-independent statistical data on the buffer pool or BP cache including hash tables. Object-independent data do not include any individual information on the object such as object name, size or addresses.
- The **Other Functions** section contains functions for selecting a buffer pool and for specifying the objects to be loaded into the buffer pool or BP cache.

The individual functions are listed below. You invoke a function by entering the one-letter code that corresponds to the function required in the **Code** field, for example, L for **List Objects**.

Code	Function	Explanation
L	<b>List Objects</b>	Displays information on the objects loaded in the buffer pool and/or the BP cache (if used). Each list item can be accessed individually and various functions can be performed for each object.
D	<b>Delete Objects</b>	Deletes one or more objects from the buffer pool and the BP cache.
I	<b>Directory Information</b>	Displays the full directory information of a specified object loaded in the buffer pool or the BP cache.
H	<b>Hexadecimal Display</b>	Displays in hexadecimal format a specified object loaded in the buffer pool.
W	<b>Write to Work File</b>	Writes to a local file or a PC text file the object directory information located in the buffer pool and/or BP cache.

Code	Function	Explanation
X	<b>Display Sorted Extract</b>	Displays a sorted list of 50 object directories located in the buffer pool or BP cache. The list items can be arranged by using any of the sort criteria provided.
A	<b>Buffer Pool</b>	Invokes the <b>Buffer Pool Statistics</b> menu. From this menu, you can invoke object-independent statistics functions for the buffer pool including hash table statistics.
C	<b>BP Cache</b>	BP cache required.  Invokes the <b>BP Cache Statistics</b> menu. From this menu, you can invoke object-independent statistics functions for the BP cache including hash table statistics.
S	<b>Select Buffer Pool</b>	Displays a selection list of all available buffer pools.
B	<b>Blacklist Maintenance</b>	Invokes the <b>Blacklist Maintenance</b> menu which is used to maintain a blacklist of objects, which are <i>not</i> to be executed.
P	<b>Preload List Maintenance</b>	Invokes the <b>Preload List Maintenance</b> menu. In a preload list, you can specify the names of objects, which are to be loaded into the buffer pool when the buffer pool is initialized.

### PF Keys and Direct Commands

In the SYSBPM **Main Menu**, you can use the PF keys or SYSBPM direct commands listed in the table below. An underlined portion of a command represents its minimum abbreviation. For further commands, see *[SYSBPM Direct Commands](#)*.

PF Key	Command	Function
PF1		Provides SYSBPM help information: see also <i><a href="#">Online Help</a></i> .
PF3	<u>EXIT</u>	Leaves the current function/screen and displays the previous screen.
PF4	LAST	Displays the SYSBPM direct command entered most recently.
PF6	FLIP	Switches the PF-key line: toggles between the display of PF1 to PF12 and PF13 to PF24.
PF12	<u>CANCEL</u>	Same as <u>EXIT</u> .
PF15	MENU	Invokes the SYSBPM <b>Main Menu</b> .

## SYSBPM in a z/OS Parallel Sysplex Environment

---

Whenever Natural switches to another operating system image (host), Natural also switches buffer pools. A switch of buffer pools is indicated by a different host ID, which is displayed in the **Loc** field of a SYSBPM screen.

Switching can take place after each terminal I/O, that is, after choosing any function key or by choosing ENTER. After switching buffer pools, browsing and positioning commands will not be executed (TOP, BOTTOM, +, -, LEFT, RIGHT). Instead, the list starts from the top of the new buffer pool.

If the BPPROP profile parameter (see *BPPROP - Global Buffer Pool Propagation* in the *Parameter Reference* documentation) is set to PLEX or to GPLEX, SYSBPM commands that manipulate blacklists, delete objects or initialize the buffer pool are first executed as usual, and then propagated to other buffer pools available on the same subsystem. If a BP switch caused a function to be aborted or propagated, an appropriate message appears. An appropriate message also appears if Natural has successfully switched to another host and changed buffer pools.



# 56 List Objects

---

- Invoking List Objects ..... 296
- Columns and Selection Options ..... 297
- PF Keys and Direct Commands ..... 306
- Line Commands ..... 307

This function invokes the **List Objects** screen where you can obtain statistical data on the directories of Natural objects currently loaded in the buffer pool (BP) or the BP cache (if used).

This section provides information on the statistical data displayed on the **List Objects** screen and the commands and functions available for selecting an object or a range of objects, manipulating their current status or navigating in the **List Objects** screen.

See also the function **Display Sorted Extract** for a sorted display of objects.

## Invoking List Objects

### ▶ To invoke the List Objects screen

- In the SYSBPM **Main Menu**, in the **Code** field, enter function code L and specify the object(s): see the valid **field input values** described in the section *Invoking and Operating SYSBPM*.

Or:

Go directly to the list of objects in the buffer pool by entering the following SYSBPM direct command:

```
DISPLAY LIST library-name object-name dbid fnr
```

Or:

Go directly to the list of objects in the BP cache by entering the following SYSBPM direct command:

```
DISPLAY CLIST library-name object-name dbid fnr
```

A **List Objects** screen similar to the example below appears:

```
17:13:17          ***** NATURAL SYSBPM UTILITY *****          2002-09-16
BPNAME QA41GBP          - List Objects -          Type Global Nat
BPPROP OFF          Loc DAEF QA41
C Library Object DBID FNR Loc RLD Use Max Reuse TotalUC ObjSize Sto
* * * * *
__ SYSBPM BPMCALL 10 1640 B 1 1 5 8,516 12
__ SYSBPM BPMNSC 10 410 B 1 4 3,380 4
__ SYSDLINP PCNDL02 255 253 B R 1 19 292 4
__ SYSLIBS NAT00017 10 410 B 1 1 5,000 8
__ SYSLIB ATEST 10 1640 B R 1 340 16,148 16
__ SYSLIB CATALL10 10 410 B 1 1 4,256 8
__ SYSBPM BPM141-M 10 1640 B 1 1 5,944 8
__ SYSDLINS U246005 255 253 B R 1 14 52 4
```

___	SYSBPM	MENU	10	1640	B		1	5	10,392	12
___	SYSLIBS	NAT00040	10	410	B		1	1	2,816	4
___	SYSLIBS	NAT00034	10	410	B		1	1	2,672	4
___	SYSDLIND	DNDL01	255	253	B	R	3	42	552	4
___	SYSLIB	ACATALL	10	410	B		1	3	55,728	56
___	SYSDLINS	U246004	255	253	B	R	2	28	172	4

Top of List  
 Command ==>  
 Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---  
 Help Exit Last Cache -- - + ++ > Canc

The **List Objects** screen lists all individual objects

1. currently loaded in the buffer pool (first part of the display) and
2. currently loaded in the BP cache (second part of the display).

The statistics displayed are snapshots of the contents of the buffer pool which are refreshed every time you press ENTER.

#### Note for GDA Objects Loaded in the Buffer Pool:

On the **List Objects** screen, two entries may be displayed for a GDA (global data area): one entry contains data in the GDA itself and the other entry contains the internal Natural symbol table for this GDA. This can happen if a program has been cataloged that references a GDA.

## Columns and Selection Options

This section describes the columns and statistical data displayed on the **List Objects** screen, and the input field and values provided for specifying object selection criteria.

### ▶ To specify selection criteria for the objects to be displayed

- In the input fields underneath the column titles, enter a valid value or range as described for the relevant fields.

The default value is a blank character or asterisk (\*) which selects all objects.

Column	Explanation	
<b>C</b>	In this column, you can enter a line command to perform a function for the object. See also <a href="#">Line Commands</a> .	
<b>Library</b>	The library from which the object was loaded. To specify selection criteria, see <a href="#">Name and Range Specification</a> .	
<b>Object</b>	The name of the object. To specify selection criteria, see <a href="#">Name and Range Specification</a> .	
<b>DBID</b>	The database ID of the Natural system file from which the object was loaded. To select objects of a specific database, enter a valid numeric value.	
<b>FNR</b>	The file number of the Natural system file from which the object was loaded. To select objects of a specific file, enter a valid numeric value.	
<b>Loc</b>	Location of the object:	
	B	Buffer pool.
	B/C	Buffer pool and BP cache.
	C	BP cache.
	C/B	BP cache and buffer pool.
	If B is listed in the first position, the statistical data derive from the buffer pool. If C is listed first, the data derive from the BP cache. Additionally, depending on this positioning, different line commands apply to the fields on the statistics screen (see also <a href="#">Line Commands</a> ).	
	To specify the object location(s), enter one of the values below:	
	B	Selects all objects loaded in the buffer pool only.
	B/C	Selects all objects loaded in the buffer pool as well as in the BP cache.
	B*	Selects all objects loaded in the buffer pool or in both the buffer pool and BP cache ( $B^* = B + B/C$ ).
C	Selects all objects loaded in the BP cache only.	
C/B	Selects all objects loaded in the BP cache as well as in the buffer pool.	
C*	Selects all objects loaded in the BP cache or in both the BP cache and buffer pool ( $C^* = C + C/B$ ).	

Column	Explanation	
RLD	Current status of the object in the buffer pool or the BP cache. A BP cache status only refers to object locking and is therefore only indicated underneath the L (Locked) of the RLD column.	
	Buffer pool:	
	R	Marked as resident. Resident means that the object is not deleted from the buffer pool, not even if the relevant value in the <b>Use</b> column changes to 0 (zero) denoting that the object is no longer used.
	L	Locked while load function is ongoing.
	D	A Delete call for the object is pending. The object will be deleted from the buffer pool as soon as the value in the <b>Use</b> column changes to 0 (zero).
	BP cache:	
	L	Locked while load function is ongoing.
	D	Locked for delete.
	To select all objects of a specific status, as described above, enter the code R, L or D.	

Column	Explanation	
<b>Use</b>	Buffer pool only.	
	The number of Natural applications that are currently executing the object.	
	To select objects, you can specify one of the following:	
	<i>value</i>	A numeric value. Selects all objects with this number. Example: 10 Selected: 10
	> <i>value</i>	A numeric start value (>). Selects all objects with a number greater than <i>value</i> . Example: >10 Selected: 11, 21 Not selected: 10
< <i>value</i>	A numeric end value (<). Selects all objects with a number less than <i>value</i> . Example: <10 Selected: 9, 8 Not selected: 10	
<b>Max</b>	Buffer pool only.  The maximum number of applications that have executed the object since it was loaded into the buffer pool.  To select objects, see the valid input values in <b>Use</b> above.	
<b>Reuse</b>	BP cache only.  Indicates how many time the object has been loaded (reused) from the BP cache into the buffer pool.  To select objects, see the valid input values in <b>Use</b> above.	

Column	Explanation				
<b>TotalUC</b>	<p>Total Use Count: The total number of Locate calls of the object since it was loaded into the buffer pool.</p> <hr/> <p>If a BP cache is used, this value is not lost if the object is removed from the buffer pool and saved to the BP cache. Therefore, this value indicates the number of times the object has been used since it was loaded from the system file.</p> <hr/> <p>For buffer pool objects, this value is updated regularly. For BP cache objects, this value is only updated after the object was removed from the buffer pool and saved in the BP cache.</p> <hr/> <p>To select objects, you can specify one of the following:</p> <hr/> <table border="0" data-bbox="423 779 1474 999"> <tr> <td data-bbox="423 779 862 999"><i>value</i> or &gt;<i>value</i></td> <td data-bbox="867 779 1474 999">           A numeric value or a numeric start value (&gt;).            Selects all objects with a number greater than or equal to <i>value</i>.            Example: &gt;10            Selected: 10, 11, 21            Not selected: 9         </td> </tr> <tr> <td data-bbox="423 1045 862 1266">&lt;<i>value</i></td> <td data-bbox="867 1045 1474 1266">           A numeric end value(&lt;).            Selects all objects with a number less than <i>value</i> or equal to <i>value</i>.            Example: &lt;10            Selected: 10, 9, 8            Not selected: 11         </td> </tr> </table>	<i>value</i> or > <i>value</i>	A numeric value or a numeric start value (>). Selects all objects with a number greater than or equal to <i>value</i> . Example: >10 Selected: 10, 11, 21 Not selected: 9	< <i>value</i>	A numeric end value(<). Selects all objects with a number less than <i>value</i> or equal to <i>value</i> . Example: <10 Selected: 10, 9, 8 Not selected: 11
<i>value</i> or > <i>value</i>	A numeric value or a numeric start value (>). Selects all objects with a number greater than or equal to <i>value</i> . Example: >10 Selected: 10, 11, 21 Not selected: 9				
< <i>value</i>	A numeric end value(<). Selects all objects with a number less than <i>value</i> or equal to <i>value</i> . Example: <10 Selected: 10, 9, 8 Not selected: 11				
<b>ObjSize</b>	<p>The size of the object.</p> <p>To select objects, see the valid input values in <b>TotalUC</b> above.</p>				
<b>Sto</b>	<p>Storage that has to be allocated for the object in the buffer pool or BP cache. The text record size of the buffer pool is defined at buffer pool initialization.</p> <p>To select objects, see the valid input values in <b>Use</b> above.</p>				
<b>BP Load Time*</b>	<p>The date and time when the object was first loaded into the buffer pool.</p> <p>This date and time will be kept until the object has been removed from both the buffer pool and BP cache (deletion from the BP cache only will not remove the display of date and time).</p> <p>To select objects, see <i>Date Specification</i> and <i>Time Specification</i>.</p>				

Column	Explanation
<b>BP Last Action*</b>	<p>Buffer pool only.</p> <p>The date and time when the object was last used by an application.</p> <p>To select objects, see <i>Date Specification</i> and <i>Time Specification</i>.</p>
<b>BPC Load Time*</b>	<p>BP cache (BPC) only.</p> <p>The date and time when the object was first loaded into the BP cache.</p> <p>This date and time will be kept until the object has been removed from the BP cache.</p> <p>To select objects, see <i>Date Specification</i> and <i>Time Specification</i>.</p>
<b>BPC Last Get*</b>	<p>BP cache (BPC) only.</p> <p>The date and time when the object was last swapped from the buffer pool into the BP cache. This time stamp is also updated if the object was already available in the BP cache and had therefore not been written to the BP cache again.</p> <p>To select objects, see <i>Date Specification</i> and <i>Time Specification</i>.</p>
<b>BPC Last Put*</b>	<p>BP cache (BPC) only.</p> <p>The date and time when the object was last loaded from the BP cache into the buffer pool.</p> <p>To select objects, see <i>Date Specification</i> and <i>Time Specification</i>.</p>
<b>1.BPperiod*</b>	<p>BP cache (BPC) only.</p> <p>The time frame the object has been available in the buffer pool starting with the time the object was first loaded and ending with the time the object was first swapped from the buffer pool into the BP cache. After 24 hours, the display of the time frame is canceled and replaced by this sign: **: **: **: *</p>
<b>NatVers*</b>	The Natural version number an object is cataloged with.

\* press **PF11** to display these columns as described in *PF Keys and Direct Commands*

This section covers the following topics:

- [Name and Range Specification](#)
- [Date Specification](#)

- Time Specification

## Name and Range Specification

You can shorten the list of objects displayed on the **List Objects** screen by entering a name or a range of names in the input fields for **Library** and/or **Object**.

In the list of options below, *value* is any combination of one or more characters:

Input Value	Selected Libraries/Objects
*	All libraries/objects.  This is the default.
<i>value</i>	All libraries/objects with a name equal to <i>value</i> .
<i>value</i> *	All libraries/objects with a name that starts with <i>value</i> .  Example: AB* Selected: AB, AB1, ABC, ABEZ Not selected: AA1, ACB
<i>value</i> ?	All libraries/objects with a name that starts with <i>value</i> and ends with any single character for each question mark (?) entered.  Example: ABC? Selected: ABCA, ABCZ Not selected: AXC, ABCAA
<i>value</i> ? <i>value</i> ?	All items that match <i>value</i> combined with asterisk (*) and question mark (?) in any order.  Example: A?C*Z Selected: ABCZ, AXCBBBZ, ANCZ Not selected: ACBZ, ABDEZ, AXCBBBZA
<i>value</i> * <i>value</i> ?	
* <i>value</i> ? <i>value</i> *	
<i>value</i> >	All libraries/objects with a name greater than or equal to <i>value</i> .  Example: AB> Selected: AB, AB1, BBB, ZZZZZZZ Not selected: AA1, AAB
<i>value</i> <	All libraries/objects with a name less than or equal to <i>value</i> .  Example: AX< Selected: AB, AWW, AX Not selected: AXA, AY

## Date Specification

You can shorten the list of objects displayed on the **List Objects** screen by entering a date, a range of dates, a special date or a range of special dates in the input fields for dates.

A date must be specified in the format *YYYYMMDD* (*YYYY* = year, *MM* = month, *DD* = day).

In the list of options below, *value* is any combination of one or more digits:

Input Value	Selected Objects
<i>YYYYMMDD</i>	All objects with a date equal to <i>YYYYMMDD</i> . Example: 20070630
<i>value</i> *	All objects with a date that starts with <i>value</i> . Example: 2007* Selected: 20070101 to 20071231 Not selected: 20061231, 20080101
<i>value</i> >	All objects with a date greater than or equal to <i>value</i> . Example: 2007> Selected: 20070101 to 20070101 Not selected: 20061231
<i>value</i> <	All objects with a date less than <i>value</i> . Example: 2007< Selected: 20060101 to 20061231 Not selected: 20070101, 20071231
<b>Special Dates</b>	
<u>T</u> ODAY	All objects with the date of the current day or a day before or after the current day:
or	
<u>T</u> 0+/- <i>n</i>	TODAY All objects with the date of the current day.
	T0+/- <i>n</i> All objects with the date of the current day plus or minus <i>n</i> days. Example: The current date is June 30th of 2007. T0-5 selects 20070625.
<u>Y</u> ESTERDAY	All objects with the date of the day before the current day. Example: The current date is June 30th of 2007. YESTERDAY selects 20070629.
<u>M</u> ONTH	All objects with the date range of the current month. Example: The current month is June of 2007. MONTH selects 20070601 to 20070630.

Input Value	Selected Objects
YEAR	All objects with the date range of the current year.  Example: The current year is 2007. YEAR selects 20070101 to 20071231.

## Time Specification

You can shorten the list of objects displayed on the **List Objects** screen by entering a time or a range of times in the input fields for times.

The time must be specified in the format *HH:II:SS* (*HH* = hours, *II* = minutes, *SS* = seconds).

In the list of options below, *value* can be any combination of one or more digits:

Input Value	Selected Objects
<i>HH:II:SS</i>	All objects with a time equal to <i>HH:II:SS</i> .  Example: 14:15:16
<i>value</i> *	All objects with a time that starts with <i>value</i> .  Example: 13* Selected: 13:00:00, 13:10:53, 13:59:59 Not selected: 12:59:59, 14:00:00
<i>value</i> >	All objects with a time greater than or equal to <i>value</i> .  Example: 12:30> Selected: 12:30:00, 12:30:01, 16:34:01 Not selected: 12:29:59
<i>value</i> <	All objects with a time less than <i>value</i> .  Example: 12:30< Selected: 12:29:59 Not selected: 12:30:00

## PF Keys and Direct Commands

On the **List Objects** screen, you can use the PF keys or SYSBPM direct commands listed in the table below. An underlined portion of a command represents its minimum abbreviation. For further commands, see *[SYSBPM Direct Commands](#)*.

PF Key	Command	Function
PF1		Provides SYSBPM help information. If chosen with the cursor on column C: lists all commands and functions available.  If chosen with the cursor on the input fields underneath the column titles: lists all possible input values for object selection.
PF3	<u>EXIT</u>	Leaves the current function/screen and displays the previous screen.
PF4	LAST	Displays the SYSBPM direct command entered most recently.
PF5	<u>CACHE</u>	Only applicable if BP cache data exists.  Scrolls to the top of the list with statistical data on BP cache objects.
PF6	-	Scrolls to the top of the list with statistical data on buffer pool objects.
PF7	-	Scrolls one page up in a list.
PF8	+	Scrolls one page down in a list.
PF9	++	Scrolls to the end of the list.
PF10	<	Scrolls left in the list.
	LEFT	Press PF11 to scroll to the right.
PF11	>	Scrolls right in the list and displays the additional screen columns: <b>BP Load Time, BP Last Action, BPC Load Time, BPC Last Get, BPC Last Put and 1.BPperiod.</b>
	RIGHT	Press PF10 to scroll to the left.
PF12	<u>CANCEL</u>	Same as <u>EXIT</u> .
PF15	MENU	Invokes the SYSBPM <b>Main Menu</b> .

## Line Commands

On the **List Objects** screen, in column **C**, for each object displayed, you can enter one of the line commands listed below:

Command	Function
CL	Buffer pool only. Releases an object marked as resident.
DE	Marks an object to be deleted from the buffer pool or BP cache.  The object is deleted as soon as the relevant <b>Use</b> count changes to 0 (object no longer used). If issued for a buffer pool object, the object will be deleted from both the buffer pool and the BP cache. If issued for a BP cache object, the object will be deleted from the BP cache only.
HD	Buffer pool only.  Displays in hexadecimal format the directory information of an object.
HE	Buffer pool only.  Corresponds to the function <b>Hexadecimal Display</b> as described in the relevant section.
F0	Buffer pool only.  Deletes an object immediately from the buffer pool, regardless of the relevant <b>Use</b> count.
LD	Corresponds to the function <b>Directory Information</b> as described in the relevant section.
RE	Buffer pool only.  Marks an object as resident.
Z0	Zooms in the fields <b>Object</b> , <b>Use</b> , <b>Max</b> , <b>Reuse</b> , <b>TotalUC</b> , <b>ObjSize</b> and <b>Sto</b> and displays them in full length. To zoom out, press ENTER.

For each command entered, a confirmation message appears for the relevant line overwriting text of rows displayed on the screen. Possible messages are:

- Failed (in response to any function that has not been executed successfully),
- Deleted (in response to the command DE or F0),
- Released (in response to the command CL) and
- Resident (in response to the command RE).



# 57 Delete Objects

---

This function deletes one object or more from the buffer pool (BP) and/or the BP cache. To specify the object pool from which the object(s) is to be deleted, use the **DELETE** command as described in *SYSBPM Direct Commands*.

An object that has a **Current Use Count** (see *Directory Information*) of 0 (zero) is deleted immediately. 0 denotes that such an object is no longer used. An object with a **Current Use Count** greater than 0 is marked for deletion and deleted as soon as its **Current Use Count** changes to 0.

## ▶ To invoke Delete Objects

- In the **SYSBPM Main Menu**, enter function code **D** and specify the object(s) to be deleted: see the valid **field input values** described in *Invoking and Operating SYSBPM*.

Or:

Enter the following SYSBPM direct command:

```
DELETE library-name object-name dbid fnr
```



# 58

## Directory Information

---

- Fields for Buffer Pool Objects ..... 313
- Fields for BP Cache Objects ..... 314
- PF Keys and Direct Commands ..... 315

This function is used to display the full directory of a Natural object currently loaded in the buffer pool or BP cache.

▶ **To invoke Directory Information**

- In the SYSBPM **Main Menu**, specify the following:

Enter function code I.

In the fields **Object**, **Library**, **DBID** and **FNR**, specify the object for which to display the directory: the valid **input values** are described in *Invoking and Operating SYSBPM*.

In the **Object Pool** field, enter the code that corresponds to the object pool from which to read the object directory information:

B or * (asterisk)	Buffer pool
C	BP cache

Or:

Use either of the following SYSBPM direct commands:

- `DISPLAY DIRECTORY library-name object-name dbid fnr`

(For the directory of the specified object loaded in the buffer pool.)

- `DISPLAY CDIRECTORY library-name object-name dbid fnr`

(For the directory of the specified object loaded in the BP cache.)

The **Directory Information** screen appears. Depending on the object pool (buffer pool or BP cache) selected, the screen provides different fields as described in the following section.

## Fields for Buffer Pool Objects

For directories of objects loaded in the buffer pool, the **Directory Information** screen provides the following fields and information on a specified object:

Field	Explanation	
<b>Directory of</b>	The type (for example, map) and name of the object.	
<b>Loaded</b>	<b>from Library</b>	The name of the library from which the object was loaded into the buffer pool.
	<b>on DBID/FNR</b>	The database ID (DBID) and file number (FNR) of the system file FNAT or FUSER from which the object was loaded into the buffer pool.
	<b>on</b>	The date and time when the object was loaded into the buffer pool.
	<b>by User</b>	The ID of the user who executed the object.
<b>Last Action on</b>	The date and time when the object was last used by an application.	
<b>BP Directory at Address</b>	The address of the directory of the object in the buffer pool.	
<b>Object at Address</b>	The address of the object in the buffer pool.	
<b>Allocated Size (KB)</b>	The size that has been allocated in the buffer pool for the object.	
<b>Object Size</b>	The size of the object.	
<b>Status (RLD)</b>	The status of the object:	
	R	The object is resident in the buffer pool. Resident means that the object is not deleted from the buffer pool, not even if its <b>Current Use Count</b> (see below) changes to 0 (zero).
	L	The object is currently locked.
	D	A Delete call for the object is pending. The object will be deleted from the buffer pool as soon as its <b>Current Use Count</b> (see below) changes to 0 (zero).
<b>Current Use Count</b>	The number of applications currently executing the object.  A value of 0 (zero) denotes that the object is no longer used.	
<b>Maximum Use Count</b>	The maximum number of applications that have executed the object since it was loaded into the buffer pool.	
<b>BP Total Use</b>	The total number of times an object has been executed since it was loaded from the system file into the buffer pool.  If a BP cache is used, this value is not lost if the object is removed from the buffer pool and saved to the BP cache. Therefore, this value indicates the number of times the object has been used since it was loaded from the system file.	

Field	Explanation
<b>Cataloged</b>	The information displayed in the <b>Cataloged</b> section of the <b>Directory Information</b> screen is identical to the information provided with the Natural system command <code>LIST DIRECTORY</code> described in the <i>System Commands</i> documentation.

## Fields for BP Cache Objects

For directories of objects loaded in the BP cache, the **Directory Information** screen provides the following fields and information on a specified object:

Field	Explanation								
<i>object-type</i>	The type (for example, map) and name of the object.								
<b>Library</b>	The name of the library from which the object was loaded into the buffer pool.								
<b>DBID</b>	The database ID of the system file FNAT or FUSER from which the object was initially loaded into the buffer pool.								
<b>FNR</b>	The file number of the system file FNAT or FUSER from which the object was initially loaded into the buffer pool.								
<b>Last Put</b>	The date and time when the object was last loaded from the BP cache into the buffer pool.								
<b>BP Load Time</b>	The date and time when the object was first loaded into the buffer pool.								
<b>Cache Load Time</b>	The date and time when the object was first loaded into the BP cache.								
<b>Last Get</b>	The date and time when the object was last swapped from the buffer pool into the BP cache.								
<b>Position Index</b>	Serially numbered internal Natural position index of the objects in the BP cache.								
<b>First Data Block Offset</b>	The address of the directory of the object in the BP cache.								
<b>Allocated Size (KB)</b>	The size that has been allocated in the BP cache for the object.								
<b>Object Size</b>	The size of the object.								
<b>Status</b>	The status of the object: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 20%;"></td> <td></td> </tr> <tr> <td>L</td> <td>Locked while load function is ongoing.</td> </tr> <tr> <td></td> <td></td> </tr> <tr> <td>D</td> <td>Locked for delete.</td> </tr> </table>			L	Locked while load function is ongoing.			D	Locked for delete.
L	Locked while load function is ongoing.								
D	Locked for delete.								
<b>Reuse</b>	Indicates how many times the object has been returned from the BP cache to the buffer pool.								
<b>BP Total Use</b>	The total number of times an object has been executed since it was initially loaded from the system file into the buffer pool and then into the BP cache.								

Field	Explanation
Cataloged	The information displayed in the <b>Cataloged</b> section of the <b>Directory Information</b> screen is identical to the information provided with the Natural system command LIST DIRECTORY described in the <i>System Commands</i> documentation.

## PF Keys and Direct Commands

On the **Directory Information** screen, you can use the PF keys or SYSBPM direct commands listed in the table below. An underlined portion of a command represents its minimum abbreviation. For further commands, see [SYSBPM Direct Commands](#).

PF Key	Command	Function
PF1		Provides help information on SYSBPM direct commands.
PF2	NEXT	Only applies if a range of objects was selected. Displays one object after the other and then redisplay the screen on which NEXT was entered.
PF3	<u>EXIT</u>	Leaves the current function/screen and displays the previous screen.
PF4	LAST	Displays the SYSBPM direct command entered most recently.
PF6	FLIP	Switches the PF-key line: toggles between the display of PF1 to PF12 and PF13 to PF24.
PF12	<u>CANCEL</u>	Same as <u>EXIT</u> .
PF15	MENU	Invokes the SYSBPM <b>Main Menu</b> .
	<u>FDELETE</u>	Buffer pool only. Deletes an object immediately from the buffer pool, regardless of its <b>Use</b> count.
	<u>RESIDENT</u>	Buffer pool only. Marks an object as resident. Resident means that the object is not deleted from the buffer pool, not even if its <b>Use</b> count is 0 (object no longer used).
	<u>CLEAR</u>	Buffer pool only. Releases an object marked as resident.
	<u>DELETE</u>	Marks an object for deletion. See Status D of the <b>buffer pool</b> or <b>BP cache</b> mentioned in a previous section.



# 59 Hexadecimal Display

---

- PF Keys and Direct Commands ..... 318

This function is used to display in hexadecimal format the code of a Natural object currently loaded in the buffer pool.

### ▶ To invoke Hexadecimal Display

- In the SYSBPM **Main Menu**, enter function code H and specify an object: see the valid **field input values** as described in the section *Invoking and Operating SYSBPM*.

Or:

Enter the following SYSPBM direct command:

```
DISPLAY HEX library-name object-name dbid fnr
```

The **Hexadecimal Display** screen appears with the object code displayed in hexadecimal format.

## PF Keys and Direct Commands

Within the object displayed on the screen, you can move to a specific location by entering either an absolute hexadecimal address or a hexadecimal offset relative to your current position.

On the **Hexadecimal Display** screen, you can use the PF keys or SYSBPM direct commands listed in the table below. An underlined portion of a command represents its minimum abbreviation. For further commands, see *SYSBPM Direct Commands*.

PF Key	Command	Function
PF1		Provides help information on SYSBPM direct commands.
PF2	NEXT	Only applies if a range of objects was selected. Displays one object after the other and then redisplay the screen on which NEXT was entered.
PF3	<u>EXIT</u>	Leaves the current function/screen and displays the previous screen.
PF4	LAST	Displays the SYSBPM direct command entered most recently.
PF6	-	Scrolls to the top of the display.
PF7	-	Scrolls up one page.
PF8	+	Scrolls down one page.
PF9	++	Scrolls to the end of the display.
PF12	<u>CANCEL</u>	Same as <u>EXIT</u> .
PF15	MENU	Invokes the SYSBPM <b>Main Menu</b> .
	<u>NEXT</u>	Only applies if a range of objects was selected.

---

PF Key	Command	Function
		Displays one object after the other and then redisplay the screen on which NEXT was entered.



# 60 Write to Work File

---

This function writes to a work file the directory information of Natural objects currently loaded in the buffer pool and/or BP cache.

▶ **To invoke Write to Work File**

1 In the **SYSBPM Main Menu**, enter the following:

- In the **Code** field, enter function code W.
- In the **Object Pool** field, enter the code that corresponds to the object pool from which to read the object directory information:

B	Buffer pool.
C	BP cache.
* (asterisk)	Both buffer pool and BP cache.

Or:

Enter one of the following **SYSBPM** direct commands:

■ **WRITE ALL**

(saves buffer pool and BP cache data)

■ **WRITE BP**

(saves buffer pool data only)

■ **WRITE BPC**

(saves buffer pool data only)

The **Work File Selection** window appears.

2 Specify the following:

- Select the target work file:

Enter N (No; this is the default setting) to output the data on Natural Work File 1.

Or:

If Entire Connection is installed, enter Y (Yes) to output the data on a PC text file by using Natural Work File 7.

- Enter the delimiter character, for example a semi-colon (;), to be used for separating the columns in the work file. The default is a blank character.

The statistical data written to the work file are snapshots of the list generated by the **List Objects** function. Refer to [List Objects](#) for explanations of the columns.

The PC text file can be used as the basis for spreadsheet calculation.

# 61 Display Sorted Extract

---

This function generates a sorted excerpt of 50 directory entries of Natural objects currently loaded in the buffer pool. This list can be used for evaluation purposes, such as determining the objects to be marked as resident or to be included in a preload list as described in the relevant section.

▶ **To invoke Display Sorted Extract**

- In the SYSBPM **Main Menu**, enter function code X and, in the **Object Pool** field, specify the type of pool by choosing B (buffer pool) or C (BP cache).

The **Specify Sort Criteria** window appears. In the input fields, enter any of the codes below to choose a column and the order by which to sort the statistics columns:

<b>Sort Value</b>	0	<b>ObjSize</b> = Object Size. This is the default.
	T	<b>TotalUC</b> = Total Use Count. See the description of <b>BP Total Use</b> in <i>Directory Information</i> .
	L	<b>BP Last Action</b> (only applicable to the buffer pool). The date and time when the objects were last used by an application.
<b>Sort Order</b>	D	Descending order. This is the default.
	A	Ascending order.

Or:

Use the SYSBPM direct command **SORT** or **SORT BPC** as described in the section *SYSBPM Direct Commands*.

The **BP Extract** screen appears which indicates the sort criteria specified.

The **BP Extract** screen is similar to the **List Objects** screen. For explanations of the columns and the commands that apply, refer to *List Objects*.



# 62

## Buffer Pool Statistics

---

- General Buffer Pool Statistics ..... 326
- Buffer Pool Load/Locate Statistics ..... 329
- Buffer Pool Fragmentation ..... 333
- Internal Function Usage ..... 335
- Buffer Pool Hash Table Statistics ..... 335
- Performance Hints ..... 338
- PF Keys and Direct Commands ..... 342

The **Buffer Pool** function invokes the **Buffer Pool Statistics** menu, which is used to obtain buffer-pool-related statistics (including hash table statistics) that are independent of Natural objects.

▶ **To invoke Buffer Pools Statistics**

- In the SYSBPM **Main Menu**, enter the following function code:

```
A
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY STATISTICS
```

The **Buffer Pool Statistics** menu appears.

The functions available in the **Buffer Pool Statistics** menu and the commands provided on the screens that are invoked by these functions are described in this section.

## General Buffer Pool Statistics

---

This function is used to monitor the performance of the buffer pool, and displays statistics regarding the activity of the buffer pool.

▶ **To invoke General Buffer Pool Statistics**

- In the **Buffer Pool Statistics** menu, enter the following function code:

```
G
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY GENERAL
```

The **General Buffer Pool Statistics** screen appears.

The statistics displayed on the **General Buffer Pool Statistics** screen are snapshots of the buffer pool which are refreshed each time you press ENTER. The following information is displayed:

Field	Explanation
<b>Buffer Pool Address</b>	The address of the buffer pool.
<b>Directory Section</b>	<p>The address of the buffer pool directory section relative to the beginning of the buffer pool.</p> <p>Each object loaded in the buffer pool requires a directory entry that contains information on this object. The space for these directory entries is allocated in the buffer pool.</p>
<b>Text Record Section</b>	<p>The address of the text record section relative to the beginning of the buffer pool.</p> <p>After the space used by the directory entries has been allocated, the remaining space is divided into blocks called text records (whose size, by default, is 4 KB). An object can occupy one or more text records, depending on its size.</p>
<b>Dataspace attached</b>	The name of the dataspace (BP cache) attached to the buffer pool.
<b>Buffer Pool Size (MB)</b>	<p>The size of the whole buffer pool in MB.</p> <p>The buffer pool size can be specified with the NTBPI macro in the parameter module or with the BPI profile parameter described in the <i>Parameter Reference</i> documentation.</p>
<b>Directory Entry Size</b>	The size of a directory entry in bytes.
<b>Text Record Size (KB)</b>	<p>The size of a text record in KB. The text record size can be specified with the NTBPI macro in the parameter module or with the BPI profile parameter described in the <i>Parameter Reference</i> documentation.</p> <p>You can change the text record size of an existing buffer pool if you reinitialize the buffer pool by using the INITIALIZE command.</p> <p>The default text record size is set to 4 KB. However, if you use applications that consist of many rather small objects, we recommend that you reduce it to 2 KB. This reduces the percentage of unused space in the buffer pool, although it can lead to Algorithm 2 (see <i>METHOD=S</i> in the <i>Operations</i> documentation) being invoked more frequently.</p>
<b>Buffer Pool Start</b>	The date and time when the buffer pool was originally started.
<b>Last Initialization</b>	<p>The date and time when the buffer pool was most recently initialized, and the ID of the user who performed the initialization.</p> <p>The buffer pool is initialized when:</p> <ul style="list-style-type: none"> <li>■ originally starting the buffer pool,</li> <li>■ executing the INITIALIZE SYSBPM direct command, or</li> </ul>

Field	Explanation
	<ul style="list-style-type: none"> <li>■ executing the REFRESH command of the GBP operating program; see also <i>Global Buffer Pool Operating Functions</i> in the <i>Operations</i> documentation.</li> </ul>
<b>Text Records - Total</b>	The total number of text records.
<b>Text Records - Used</b>	The number of text records currently used.
<b>Text Records - Used in %</b>	The percentage of text records currently used.
<b>Text Records - Max Used</b>	The maximum number of text records used.
<b>Text Records - Total Size</b>	<p>The total space used by all text records used, which is <b>Text Records - Used</b> multiplied by the size of a single text record.</p> <p>The difference between the total text record size and the total object size shows the amount of unused size in the text record section and can also be an indicator for the system administrator of whether to modify the text record size.</p>
<b>Text Records - Avg Usage %</b>	<p>The average usage in percent of all text records used, which is <b>Objects - Total Size</b> divided by <b>Text Records - Total Size</b>.</p> <p>This value should not be significantly less than 75%. If the buffer pool is almost full, any value above 75% indicates good usage of the buffer pool. If the usage is significantly less than 75%, the text record size should be reduced.</p>
<b>Space Used %</b>	<p>The actual usage in percent of the text record section, which is <b>Objects - Total Size</b> divided by the total size of the <b>Text Records</b> section.</p> <p><b>Tip:</b> If the buffer pool is almost full (that is, the value in the field <b>Text Records - Used</b> is almost 100%), any value above 75% indicates good usage of the buffer pool. If the usage is significantly less than 75%, the text record size should be reduced.</p>
<b>Objects - Loaded</b>	The number of objects currently loaded in the buffer pool.
<b>Objects - Max Loaded</b>	The maximum number of objects ever loaded simultaneously in the buffer pool since the buffer pool was started.
<b>Objects - Total Size</b>	The total size in bytes of the objects currently loaded.
<b>Objects - Avg TR Used</b>	The average number of text records used by one object.
<b>Objects - SumOfUseCounts</b>	<p>Totals the use counts of all objects currently loaded in the buffer pool.</p> <p>The use count counts all applications currently executing an object. If an object is currently not in use, its use count changes to 0 (zero).</p>
<b>Objects - AvgLifetimeUsed(min)</b>	The average life time (in minutes) of objects currently loaded in the buffer pool.
<b>Objects - AvgLifetimeReplace(min)</b>	The average life time (in minutes) of objects, which have already been replaced in the buffer pool.

## Buffer Pool Load/Locate Statistics

This function provides statistical information on the loading of objects into the buffer pool and the locating of objects in the buffer pool. This information also serves as an indicator of buffer pool performance.

### ▶ To invoke Buffer Pool Load/Locate Statistics

- In the **Buffer Pool Statistics** menu, enter the following function code:

```
L
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY LOAD
```

The **Buffer Pool Load/Locate Statistics** screen appears.

The statistics displayed on the **Buffer Pool Load/Locate Statistics** screen are snapshots of the buffer pool which are refreshed every time you press ENTER.

The following information is displayed on the screen:

Field	Explanation
<b>Total Locate Calls</b>	The total number of object location calls; that is, the total number of times the Natural buffer pool manager was requested to search the buffer pool for an object.  If the location is successful, the object has been found in the buffer pool or the BP cache and need not be loaded from a Natural system file thereby saving calls and I/Os.
<b>Total Locate Calls - successful</b>	The total number of successful Locate calls as an absolute number.
<b>Total Locate Calls - failed</b>	The total number of Locate calls that failed.
<b>Quick Locate Calls</b>	The total number of Quick Locate calls.  A Quick Locate call is the attempt to locate an object in the buffer pool by using the internal fast locate table. For further information, see <a href="#">Internal Fast Locate Table</a> .  In this case, the library (name, database ID and file number) and the position of the object in the buffer pool of the last successful locate call for this object is used again to locate the object.  This is the most efficient way to locate an object.

Field	Explanation
<b>Quick Locate Calls - successful</b>	The number of Quick Locate calls that have been successfully performed.
<b>Quick Locate Calls - failed</b>	<p>The number of Quick Locate calls that failed.</p> <p>A failed Quick Locate call indicates that an object could not be located at its previous position in the buffer pool by using the <b>internal fast locate table</b> (see the relevant section in <i>Performance Considerations</i>).</p> <p>This occurs if the object has either been deleted from the buffer pool or was moved to another position. A Quick Locate call that failed results in a Normal Locate call but without a steplib search since the Natural runtime remembers the library that contains the object.</p>
<b>Normal after Quick</b>	<p>The number of Normal Locate calls that result from <b>Quick Locate Calls - failed</b>.</p> <p>The value of <b>Normal after Quick</b> is always identical to the value of <b>Quick Locate Calls - failed</b>.</p>
<b>Normal after Quick - successful</b>	<p>The number of successful Normal Locate calls that result from <b>Quick Locate Calls - failed</b>.</p> <p>A Normal after Quick call is successful if the requested object is still in the buffer pool but at another position.</p>
<b>Normal after Quick - failed</b>	<p>The number of failed Normal Locate calls that result from <b>Quick Locate Calls - failed</b>.</p> <p>A failed Normal after Quick call indicates that the requested object is no longer available in the buffer pool. As a result, the object is reloaded from the system file by using the library entry in the <b>internal fast locate table</b> (see the relevant section in <i>Performance Considerations</i>).</p>
<b>Normal Locate Calls</b>	<p>The total number of Normal Locate calls.</p> <p>A Normal Locate call is the attempt to locate an object in the buffer pool without using the <b>internal fast locate table</b> (see the relevant section in <i>Performance Considerations</i>).</p> <p>A Normal Locate call always occurs if an object is referenced for the first time within a Natural session after a LOGON system command has been performed. The Natural runtime does not yet know the library where the object resides, compared with a Quick Locate call.</p>
<b>Normal Locate Calls - successful</b>	The number of Normal Locate calls that were successful in locating the required object in the buffer pool.
<b>Normal Locate Calls - failed</b>	<p>The number of Normal Locate calls that failed.</p> <p>A failed Normal Locate call indicates that an object (identified by its name and the library where it resides) could not be located in the buffer pool.</p>

Field	Explanation
	A failed Normal Locate will either result in a load from the BP cache or a system file, or a Normal Locate call for the next library in the steplib chain.
<b>STEPLIB Searches</b>	<p>The number of <b>Normal Locate Calls</b> that occurred from failed attempts to find an object in a steplib library.</p> <p>Normal Locate calls that perform unsuccessful steplib searches do not result in the load of an object from the BP cache or the system file.</p> <p><b>STEPLIB Searches</b> does <i>not</i> count Locate calls for objects that are neither contained in the current library nor any steplib or system file (error message <code>Invalid command, or program does not exist in library</code>). Locate calls that fail due to incorrect programming add to the counter of <b>Normal Locate Calls - failed</b>.</p> <p>The number of <b>STEPLIB Searches</b> is calculated by using the following formula:</p> <p><b>Normal Locate Calls - failed - (Number Loads into BP - Normal after Quick - failed)</b></p> <p>The fewer the number of <b>STEPLIB Searches</b>, the better the buffer pool is performing.</p> <p>See also <a href="#">Searching in Steplibs</a>.</p>
<b>Number Loads into BP</b>	<p>The number of times a load into the buffer pool was performed successfully.</p> <p>The load into the buffer pool (storage allocation request) can be triggered either by a load from the database or by a load from the BP cache.</p>
<b>Loads from Cache</b>	The total number of successful Locate calls of objects that resided in the BP cache. This information is counted only if the previous Locate call ( <b>Normal after Quick - failed</b> or <b>Normal Locate Calls - failed</b> ) failed. It indicates the number of database loads saved. This means, that, without the BP cache, the object would have to be loaded from the database.
<b>Loads from DB</b>	<p>The number of times an object was loaded from a Natural system file into the buffer pool.</p> <p>As several load calls may be necessary to load a single object, this value provides the actual number of object loads made since the most recent buffer pool refresh.</p> <p>When loading an object, the buffer pool manager uses different search algorithms: see <i>METHOD=S</i> and <i>METHOD=N</i> in the <i>Operations</i> documentation.</p>

Field	Explanation
<b>Loads from DB - finished</b>	<p>The number of object loads that finished successfully.</p> <p>An object load cannot finish if the load operation is canceled due to any of the following reasons:</p> <ul style="list-style-type: none"> <li>■ A concurrent object load occurred: see <b>Loads from DB - concurrent</b>.</li> <li>■ During the object load, an Adabas response code occurs.</li> <li>■ During the object load, a SYSBPM delete operation is executed for this object.</li> </ul>
<b>Loads from DB - concurrent</b>	<p>The number of object loads that have been performed simultaneously for the same object:</p> <p>Concurrent object loads occur if two or more Natural sessions that run simultaneously request the same object. While an object is being loaded by one session, other sessions request the same object and start loading it before a session has finished loading. In this case, the same object is loaded more than once.</p> <p>The first session that finishes loading the object will mark the object of the other sessions to be deleted from the buffer pool. The other sessions will then stop loading the object, remove the object marked for deletion from the buffer pool and use the object loaded successfully by the first session.</p> <p>The numbers of objects calculated by the counters <b>Loads from DB - finished</b> and <b>Loads from DB - concurrent</b> are usually identical. The numbers only differ if the concurrent load is only detected after both sessions have finished the load.</p>
<b>Load Calls</b>	<p>The total number of load calls made since the buffer pool has been refreshed. The load calls are correlated with the access to the system file from which the objects are read.</p> <p>The number of system file accesses is calculated as follows:</p> <ul style="list-style-type: none"> <li>■ Adabas system file: The number of <b>Load Calls</b> plus the number of <b>Loads from DB</b> (see above). The total number does not include Adabas RC calls.</li> <li>■ VSAM system file: The number of <b>Load Calls</b>.</li> </ul>
<b>Number Loads BP 2nd</b>	<p>This field is displayed if METHOD=S (selection process) is used as the search method for allocating storage.</p> <p>This field shows the number of times a storage allocation request satisfied the search criteria of Algorithm 2 as described in <i>METHOD=S</i> in the <i>Operations</i> documentation.</p>

Field	Explanation
<b>Number Load Cycles</b>	<p>This field is displayed if METHOD=N (next available) is used as the search method for allocating storage as described in the <i>Operations</i> documentation.</p> <p>This field indicates the number of times a search has been performed starting from the top of the buffer pool. This number gives an estimate of the frequency of cycling through the buffer pool in a wrap-around fashion.</p>
<b>Last Cycle Start</b>	<p>This field is displayed if METHOD=N (next available) is used as the search method for allocating storage as described in the <i>Operations</i> documentation.</p> <p>The time and date when <b>Number Load Cycles</b> was last increased.</p>
<b>Number Lock Retries</b>	<p>This field is displayed if METHOD=N (next available) is used as the search method for allocating storage as described in the <i>Operations</i> documentation.</p> <p>This field indicates the number of times a chain of locked buffer pool entries had to be unlocked, because they could not satisfy the allocation request.</p>
<b>Largest Alloc (TR)</b>	The largest single allocation size so far requested, specified in number of text records.
<b>Number Load Failure</b>	The total number of times an object load failed. The reason for a failure is that either all directory entries are in use at the time of the load request or not enough storage is available in the text record section to perform the load.
<b>Number Load Failure - Sizes failing last</b>	The number of text records that would have been required by the three most recent storage allocation requests that failed.

For detailed information on the search methods used for allocating space in the buffer pool, see *Buffer Pool Search Methods* in the *Operations* documentation.

## Buffer Pool Fragmentation

This function provides an overview of the buffer pool fragmentation; that is, an overview of how many different Natural objects occupy how many text records, and how the object locations are spread over the buffer pool.

### ▶ To invoke Buffer Pool Fragmentation

- In the **Buffer Pool Statistics** menu, enter the following function code:

```
F
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY FRAGMENTATION
```

The **Buffer Pool Fragmentation** screen appears.

Some of the fields provided on the **Buffer Pool Fragmentation** screen are identical to the items explained in *General Buffer Pool Statistics*:

**Buffer Pool Size**

**Buffer Pool Address**

**Text Record Section**

**Text Record Size**

**Number of Text Records** (corresponds to **Text Records - Total**)

In addition, the screen displays a diagram which shows how many different individual objects occupy how much text record size.

For example:

```

                                1-----10-----20-----30-----40-----50
005F0480  ._.+***_____++  ....**_.+**_.+_ *..+***+_++++XX
    
```

Each symbol in the diagram represents one text record, and each sequence of equal symbols represents a different individual object occupying one or more text records. The symbols have the following meaning:

_ and .	Objects with a <b>Current Use Count</b> (see <i>Directory Information</i> ) of 0 (zero).
+ and *	Objects with a <b>Current Use Count</b> greater than 0 (zero).
blank character	An unused text record.
XX	The end of the buffer pool, which means that no further text records are available.

In the example above, the buffer pool contains 48 text records. Three of them are not in use; the rest is occupied by 24 different objects, 12 of them with a **Current Use Count** of 0 (zero), and 12 with a **Current Use Count** greater than 0.

---

## Internal Function Usage

---

This function provides statistical information on the calls made to the Natural buffer pool manager.

▶ **To invoke Internal Function Usage**

- In the **Buffer Pool Statistics** menu, enter the following function code:

```
I
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY FUNCTION
```

The **Internal Function Usage** screen appears.

The statistics displayed on the **Internal Function Usage** screen are snapshots of the buffer pool which are refreshed every time you press ENTER.

The field **Total Calls** shows the overall number of all internal calls that have been made to the buffer pool manager.

Internally, the buffer pool manager can be invoked for various different functions. For each function, the number of times it has been invoked is displayed, both as an absolute number and as percentage. In addition, these numbers are represented in a horizontal bar chart.

---

## Buffer Pool Hash Table Statistics

---

This function only applies to buffer pools of the type Natural.

**Buffer Pool Hash Table Statistics** displays statistics about hash table slots and collisions per slot. The statistics determine the efficiency of the hash algorithm used.

For further information on hash tables, refer to *Buffer Pool Hash Table* in the *Operations* documentation.

The statistics are primarily intended for internal use by Software AG personnel only.

▶ **To invoke Buffer Pool Hash Table Statistics**

- In the **Buffer Pool Statistics** menu, enter the following function code:

H

Or:

Enter the following SYSBPM direct command:

DISPLAY HASH

The **Hash Table Collisions** screen appears.

The statistics displayed on the **Hash Table Collisions** screen are snapshots of the hash table which are taken every time you press ENTER. The following information is displayed:

Field	Explanation																				
<b>Total Number of Slots</b>	<p>The total number of hash table slots; that is, the total possible entries in the hash table that link the object names with the location of the objects.</p> <p>The number of slots, that is, the size of the hash table will be calculated internally depending on the number of text records.</p>																				
<b>Number of Slots used</b>	The number of slots in the hash table that have at least one object name mapped to them.																				
<b>Number of Slots free</b>	The number of slots in the hash table that have no object name mapped to them.																				
<b>Max. Collisions per Slot</b>	<p>The maximum number of collisions of any slot. The maximum number of collisions is the longest possible search path for an object.</p> <p>A collision is caused if the name of two different objects is mapped to the same slot by the hash algorithm. In this case, a collision resolution is used in order to find another slot.</p>																				
<b>Collisions</b>	<p>The number of current collisions. Depending on the collisions that occur, the table contains up to 10 rows:</p> <table border="1"> <tbody> <tr> <td>0</td> <td>No collision.</td> </tr> <tr> <td>1</td> <td>1 collision.</td> </tr> <tr> <td>2</td> <td>2 collisions.</td> </tr> <tr> <td>3</td> <td>3 collisions.</td> </tr> <tr> <td>4</td> <td>4 collisions.</td> </tr> <tr> <td>5</td> <td>5 collisions.</td> </tr> <tr> <td>6 - 10</td> <td>Between 6 and 10 collisions.</td> </tr> <tr> <td>11 - 15</td> <td>Between 11 and 15 collisions.</td> </tr> <tr> <td>16 - 20</td> <td>Between 16 and 20 collisions.</td> </tr> <tr> <td>21</td> <td>More than 21 collisions.</td> </tr> </tbody> </table>	0	No collision.	1	1 collision.	2	2 collisions.	3	3 collisions.	4	4 collisions.	5	5 collisions.	6 - 10	Between 6 and 10 collisions.	11 - 15	Between 11 and 15 collisions.	16 - 20	Between 16 and 20 collisions.	21	More than 21 collisions.
0	No collision.																				
1	1 collision.																				
2	2 collisions.																				
3	3 collisions.																				
4	4 collisions.																				
5	5 collisions.																				
6 - 10	Between 6 and 10 collisions.																				
11 - 15	Between 11 and 15 collisions.																				
16 - 20	Between 16 and 20 collisions.																				
21	More than 21 collisions.																				

Field	Explanation
	No collision means that only one object name is mapped per slot. To locate this object, you need to access the hash table once only.
	If the number of collisions is greater than 0 (zero), for example, $x$ , $x+1$ object names are mapped to the same slot. To locate one of these objects, you need to access the hash table up to $x+1$ .
<b>Number of Slots</b>	The number of slots related to the number of collisions.  In addition, the percentage of these slots related to all slots used is displayed.
<b>Number of Slots Totalled</b>	The same values as <b>Number of Slots</b> , but the values are totaled.

### Example of Hash Table Statistics

```

14:36:26          ***** NATURAL SYSBPM UTILITY *****          2003-08-13
BPNAME NATGBP    - Buffer Pool Hash Table Statistics -          Type Global Nat
BPPROP OFF                                             Loc DAEF QA41

Total Number of Slots ..          523
Number of Slots used ..          475 ( 90.8 %)      Max. Collisions
Number of Slots free ..          48 ( 9.1 %)          per Slot ..... 7

Collisions      Number of Slots      Number of Slots Totalled
    0              0 ( 0.0 %)          0 ( 0.0 %)
    1             164 ( 34.5 %)         164 ( 34.5 %)
    2             194 ( 40.8 %)         358 ( 75.3 %)
    3              96 ( 20.2 %)         454 ( 95.5 %)
    4              16 ( 3.3 %)          470 ( 98.9 %)
    5               4 ( 0.8 %)          474 ( 99.7 %)
    6 - 10        1 ( 0.2 %)          475 ( 100.0 %)

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help      Exit  Last      Flip      Canc

```

## Performance Hints

This function only applies to a buffer pool and a BP cache of the type Nat (Natural).

The **Performance Hints** function provides statistical information about workloads on a Natural buffer pool and a BP cache including ratings of the overall performance.

### Related Topic:

- *Performance Considerations*

This section covers the following topics:

- [Invoking Performance Hints](#)
- [Evaluating Performance Hints](#)

### Invoking Performance Hints

This section describes how to invoke the **Performance Hints** function and the statistics field displayed on the **Performance Hints** screen.

#### ▶ To invoke Performance Hints

- In the **Buffer Pool Statistics** menu, enter the following function code:

```
P
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY PERFORMANCE
```

A **Performance Hints** screen similar to the example below appears:

```

13:27:16          ***** NATURAL SYSBPM UTILITY *****          2005-05-19
BPNAME QA41GBP          - Performance Hints -          Type Global Nat
BPPROP OFF          Loc DAEF QA41
          Preload QA41GBPL

          Rating
          (1=best - 6=worst)

Buffer Pool
```

```

Locates / Loads Ratio .....      162.85      3
Wrap Time Last (hh:mm:ss) ..      00:06:29      4
Wrap Time Avg (hh:mm:ss) ...      00:01:22      5

BP Cache

Object Reuse Factor .....          3.11         4
Wrap Time Last (hh:mm:ss) ..      02:02:29      1
Wrap Time Avg (hh:mm:ss) ...      00:32:06      3
Get / Search % .....              74.48 %

Command ==>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help       Exit  Last       Flip                               Canc

```

The fields on the **Performance Hints** screen provide the following information:

Field	Explanation
<b>Buffer Pool - Locates / Loads Ratio</b>	<p>The ratio of <b>Total Locate Calls - successful</b> to <b>Loads from DB</b>. A value greater than 1 indicates that Natural located more objects in the buffer pool than it loaded from the system file.</p> <p>This ratio serves as a buffer pool efficiency indicator. The larger the number, the better the buffer pool is performing. This is the primary indicator of performance from one buffer pool session to the next.</p>
<b>Buffer Pool - Wrap Time Last</b>	<p>This field is displayed if METHOD=N (next available) is used as the search method for allocating storage as described in the <i>Operations</i> documentation.</p> <p>The time in hours, minutes and seconds (<i>hh:mm:ss</i>) since the buffer pool was last completely reused (wrapped around).</p>

Field	Explanation
	<p>Objects are loaded into the buffer pool one after the other (in sequential order) filling the top of the buffer pool first and the bottom last. When the bottom of the buffer pool has been reached, the buffer end is wrapped around and the next object is loaded again at the top of the buffer pool.</p> <p>When the buffer pool has been filled completely for the first time, a new object loaded into the buffer overwrites an object that was loaded in the previous wrap-around cycle and that is not currently locked (marked as resident or in use).</p> <p>Every object loaded into the buffer pool is assigned a directory entry that contains information such as the name of the object, the library where it is stored and the <b>BP Load Time</b> time stamp of the loading into the buffer pool.</p> <p><b>Wrap Time Last</b> is evaluated each time an object is loaded into the buffer pool. <b>Wrap Time Last</b> indicates the time period between the last loading of an object (<b>BP Load Time</b>) and the last overwriting of an object.</p> <p>The longer the period of a wrap-around cycle, the better the buffer pool is performing. This period can vary considerably at night or over a weekend when user traffic is low compared with normal working hours.</p>
<b>Buffer Pool - Wrap Time Avg</b>	<p>This field is displayed if METHOD=N (next available) is used as the search method for allocating storage as described in the <i>Operations</i> documentation.</p> <p>The average time in hours, minutes and seconds (<i>hh:mm:ss</i>) of one wrap-around cycle since the buffer pool was initialized or refreshed.</p> <p><b>Wrap Time Avg</b> is calculated by dividing the life time of the buffer pool by the number of wrap-around cycles.</p> <p><b>Wrap Time Last</b> compared with <b>Wrap Time Avg</b> shows whether the buffer pool is currently being used more frequently than on average.</p>
<b>BP Cache - Object Reuse Factor</b>	<p>The ratio of objects swapped out of the BP cache (Get calls) to objects swapped into the BP cache (<b>Put calls</b>).</p> <p>The value is calculated by dividing successful Get calls by successful Put calls. It shows the overall reuse factor; that is, how often an object loaded once into the BP cache could be successfully reloaded into the buffer pool. The higher the value, the better the BP cache efficiency.</p> <p>Example of an <b>Object Reuse Factor</b>: A ratio of 5.70 indicates that an object loaded in the BP cache was swapped into the buffer pool 5.7 times on average.</p>
<b>BP Cache - Wrap Time Last</b>	<p>The time in hours, minutes and seconds (<i>hh:mm:ss</i>) since the BP cache was last completely reused (wrapped around).</p> <p>Objects are loaded into the BP cache one after the other (in sequential order) filling the top of the BP cache first and the bottom last. When the bottom of the BP cache has been reached, the end of the BP cache is wrapped around and the next object is loaded again at the top of the BP cache. When the BP cache has been filled completely for the first time,</p>

Field	Explanation
	<p>a new object loaded into the buffer overwrites the object that was loaded in the previous wrap-around cycle.</p> <p>Each object loaded into the BP cache is assigned a directory entry that contains information such as the name of the object, the library where it is stored and the <b>BPC Load Time</b> time stamp when it was loaded into the BP cache.</p> <p><b>Wrap Time Last</b> is evaluated every time an object is loaded into the BP cache. <b>Wrap Time Last</b> indicates the time period between the last loading of an object (<b>BPC Load Time</b>) and the last overwriting of an object.</p> <p>The longer the period of a wrap-around cycle, the better the BP cache is performing. This period can vary considerably at night or over a weekend when user traffic is low compared with normal working hours.</p>
<b>BP Cache - Wrap Time Avg</b>	<p>The average time in hours, minutes and seconds (<i>hh:mm:ss</i>) of one wrap-around cycle since the BP cache was started.</p> <p><b>Wrap Time Avg</b> is calculated by dividing the life time of the BP cache by the number of wrap-around cycles.</p> <p><b>Wrap Time Last</b> compared with <b>Wrap Time Avg</b> shows whether the BP cache is currently being used more frequently than on average.</p>
<b>BP Cache - Get / Search %</b>	<p>The percentage of objects returned from the BP cache successfully (Get calls) compared with the total number of search requests (Search calls) the buffer pool sent to the BP cache.</p> <p>This value indicates the percentage of objects the buffer pool could load from the BP cache, instead of the Natural system file FNAT or FUSER. The higher the value, the better the cache efficiency.</p> <p>Keep in mind that a search for an object in a chain of steplib may increase the number of Search calls the buffer pool sends to the BP cache. However, these calls will not result in a successful Get call or a load from the Natural system file, because an object may not be found in a steplib. <b>Get / Search %</b> does not consider the search through a long chain of steplib that is frequently used. We recommend that you use as few steplib as possible to increase overall performance.</p> <p>Example of a <b>Get / Search %</b> value: A value of 70% indicates that 70 % of all objects loaded into the buffer pool were retrieved from the BP cache and 30% were loaded from the system file.</p>

## Evaluating Performance Hints

The statistical values provided on the **Performance Hints** screen are the basis for a performance rating system where 1 denotes best (highest) and 6 denotes worst (lowest) performance.

The ratings shall give you an idea of how a buffer pool or a BP cache performs with the sized specified for them. The rating values should suit the requirements of most system environments.

There are environments where it has proven to be worth having a BP cache that is five times as big as the buffer pool. If ratings tend to be bad, the size of the buffer pool or BP cache should be increased. However, the size of a buffer pool or a BP cache can guarantee good performance even though ratings are bad. If the ratings of a BP cache are good, bad ratings for the buffer pool do not carry so much weight. For environments with extreme workloads, the ratings can, however, be useful indicators for when the size of the buffer pool or the BP cache should be changed.

The statistical values displayed on the **Performance Hints** screen are snapshots of the Natural buffer pool and BP cache which are refreshed each time a wrap-around occurs. Buffer pool and the BP cache should run for some time and the statistics values reach a certain extent to obtain meaningful results from these values. For example, if the size of a buffer pool is so large that only very few BP cache calls are required, the statistics regarding the BP cache will not be meaningful.

When evaluating the statistics you should also consider the type of system environment (for example, production versus test), the type of applications using the buffer pool (batch, online, user-defined or system), user traffic (peak hours or normal hours) and extraordinary operational factors.

## PF Keys and Direct Commands

On the buffer pool statistics screens, you can use the PF keys or SYSBPM direct commands listed in the table below. An underlined portion of a command represents its minimum abbreviation. For further commands, see [SYSBPM Direct Commands](#).

PF Key	Command	Function
PF1		Provides SYSBPM help information: see also <a href="#">Online Help</a> .
PF3	<u>EXIT</u>	Leaves the current function/screen and displays the previous screen.
PF4	LAST	Displays the SYSBPM direct command entered most recently.
PF6	FLIP	Switches the PF-key line: toggles between the display of PF1 to PF12 and PF13 to PF24.
PF8 (Load)	<u>DISPLAY</u> <u>L</u> OAD	Only applies to the screen <b>General Buffer Pool Statistics</b> . Displays the <b>Buffer Pool Load/Locate Statistics</b> screen.
PF8 (Gen)	<u>DISPLAY</u> <u>G</u> ENERAL	Only applies to the <b>Buffer Pool Load/Locate Statistics</b> screen. Displays the <b>General Buffer Pool Statistics</b> screen.

---

PF Key	Command	Function
PF12	CANCEL	Same as <a href="#">EXIT</a> .
PF15	MENU	Invokes the SYSBPM <b>Main Menu</b> .



# 63

## BP Cache Statistics

---

- General BP Cache Statistics ..... 346
- BP Cache Call Statistics ..... 348
- BP Cache Hash Table Statistics ..... 350
- Performance Hints ..... 351
- PF Keys and Direct Commands ..... 351

The **BP Cache** function only applies to a buffer pool of the type Natural.

This function invokes the **BP Cache Statistics** menu which is used to obtain statistical information on the BP cache.

Note that the **BP Cache** function can only be executed if a BP cache has been installed when initializing a global buffer pool (no BP cache support for local buffer pools).

▶ **To invoke BP Cache Statistics**

- In the **SYSBPM Main Menu**, enter the following function code:

```
C
```

Or:

Enter the following **SYSBPM** direct command:

```
DISPLAY CSTATISTICS
```

The **BP Cache Statistics** menu appears.

The functions available in the **BP Cache Statistics** menu and the commands provided on the screens that are invoked by these functions are explained in this section.

## General BP Cache Statistics

---

This function displays addresses and statistics regarding the activity of the BP cache.

▶ **To invoke General BP Cache Statistics**

- In the **BP Cache Statistics** menu, enter the following function code:

```
G
```

Or:

Enter the following **SYSBPM** direct command:

```
DISPLAY CGENERAL
```

The **General BP Cache Statistics** screen appears.

The statistics displayed on the **General BP Cache Statistics** screen are snapshots of the buffer pool, which are refreshed each time you press ENTER. The following information is displayed:

Field	Explanation
<b>Dataspace - Name</b>	The name of the dataspace where the BP cache resides.
<b>Dataspace - SToken</b>	The term SToken (for Space Token) identifies a dataspace.
<b>Dataspace - ALET</b>	The term ALET (for Address List Entry Token) identifies an index for accessing the dataspace.
<b>Dataspace - Size (MB)</b>	The size of the BP cache in MB.
<b>Dataspace - Current state</b>	The status of the BP cache:  not initialized locked for init closed free for operation undefined
<b>Dataspace - Initialization</b>	The date and time when the BP cache was initialized.
<b>Internal buffer offsets - Header buffer</b>	The header of the BP cache which contains general BP cache information.
<b>Internal buffer offsets - Hash buffer</b>	Contains the hash table (see also <a href="#">BP Cache Hash Table Statistics</a> ).
<b>Internal buffer offsets - Directory buffer</b>	The address of the BP cache directory section relative to the beginning of the BP cache.  Each Natural object loaded in the BP cache requires a directory entry that contains information on this object. The space for these directory entries is acquired from the BP cache itself.
<b>Internal buffer offsets - Text buffer</b>	The address of the text buffer relative to the beginning of the BP cache.  After allocating the space for all other buffers, the remaining space is divided into text records with a size of 4 KB.  An object can occupy one or more text records, depending on its size.
<b>Tot. Text Records</b>	The total number of text records in the BP cache.  The number of text records depends on the BP cache size. The text record size for the BP cache is 4 KB.
<b>Insert position</b>	The index number of the text record into which the next object will be inserted.  Objects will be inserted into the BP cache when they have to be removed from the buffer pool.

Field	Explanation
Reuse cycles	The number of times the BP cache has been completely reused.  Every time the BP cache is full, the BP cache manager reuses the BP cache from the start and overwrites the object(s) from there. The objects will remain in the BP cache until the BP cache is used again.
Objects - Max Loaded	The maximum number of objects currently loaded in the BP cache.
Objects - Loaded	The number of objects currently loaded in the BP cache.

## BP Cache Call Statistics

This function provides statistical information on the loading (put), retrieving (get) and deleting of objects into/from the BP cache. This information also serves as an indicator of BP cache performance.

### ▶ To invoke BP Cache Call Statistics

- In the **BP Cache Statistics** menu, enter the following function code:

```
L
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY CLOAD
```

The **BP Cache Call Statistics** screen appears.

The statistics displayed on the **BP Cache Call Statistics** screen are snapshots of the buffer pool which are refreshed each time you press ENTER. The following information is displayed:

Field	Explanation
Search calls	The number of Search calls the buffer pool sent to the BP cache while attempting to find an object in the BP cache.  If an object is found, a Search call results in a Get call.
Get calls (from BP cache)	The number of Get calls the buffer pool sent to the BP cache while attempting to load an object from the BP cache into the buffer pool.
Get calls - successful	The number of successful Get calls the BP cache performed, that is, the number of objects the BP cache swapped into the buffer pool.  A Get call is successful if an object the buffer pool attempted to load is actually loaded from the BP cache into the buffer pool.

Field	Explanation
	A Get call is unsuccessful, for example, if an object was deleted after it was found by the Search call.
<b>Put calls (to BP cache)</b>	The number of Put calls the buffer pool sent to the BP cache while attempting to swap an object from the buffer pool into the BP cache.
<b>Put calls - successful</b>	The number of Put calls that resulted in an object to swapped from the buffer pool into the BP cache.
<b>Put calls - obj. already cached</b>	The number of Put calls the buffer pool sent to the BP cache for objects that were already loaded in the BP cache.
<b>Delete calls</b>	<p>The number of Delete calls the buffer pool sent to the BP cache while attempting to delete an object from the BP cache.</p> <p>A Delete call requests either a single object or a range of objects (see also the section <a href="#">Delete Objects</a>).</p>
<b>Delete calls - successful</b>	<p>The number of successful Delete calls the buffer pool sent to the BP cache.</p> <p>A Delete call is successful if at least one object is actually deleted from the BP cache.</p> <p>A Delete call is unsuccessful if the object requested was not loaded in the BP cache and hence could not be deleted.</p> <p>Compared with the total number of Delete calls, the number of successful Delete calls can be very low. This happens, for example, if several Natural objects are cataloged with the CATALOG command. In this case, for every object cataloged successfully, Natural sends a Delete call to the BP cache. However, at the time the Delete call is sent, most of the cataloged objects are usually not loaded in the BP cache and the delete attempt fails.</p>
<b>Initialization</b>	The date and time when the BP cache was initialized.
<b>Last reuse cycle</b>	<p>The load date and time of the object that was most recently overwritten.</p> <p>An object is overwritten in the BP cache when its space has to be reused in order to load another object. The object that was loaded first in the BP cache will be swapped first. This means the load date and time of the object that has been in the BP cache longest corresponds to the date and time in <b>Last reuse cycle</b>.</p>
<b>Last access</b>	The date and time when the buffer pool last accessed the BP cache.
<b>Last Put (to BP cache)</b>	The date and time when the buffer pool last sent a Put call to the BP cache.
<b>Last Get (from BP cache)</b>	The date and time when the buffer pool last sent a Get call to the BP cache.
<b>Last Delete</b>	The date and time when the buffer pool last sent a Delete call to the BP cache.

## BP Cache Hash Table Statistics

This function displays statistics about hash table slots and collisions per slot. The statistics determine the efficiency of the hash algorithm used.

### ▶ To invoke BP Cache Hash Table Statistics

- In the **BP Cache Statistics** menu, enter the following function code:

```
H
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY CHASH
```

The **Cache Hash Table Collisions** screen appears.

The statistics displayed on the **Cache Hash Table Collisions** screen are snapshots of the hash table which are refreshed every time you press **ENTER**. The following information is displayed:

Field	Explanation
<b>Total Number of Slots</b>	The total number of hash table slots; that is, the total possible entries that link the object name with the location of the object.  The number of slots, that is, the size of the hash table will be calculated internally depending on the number of text records.
<b>Number of Slots used</b>	The number of slots that have one or more entries.
<b>Number of Slots free</b>	The number of slots that have no entry.
<b>Max. Collisions per Slot</b>	The maximum number of collisions of all slots.  The maximum number of collisions is the longest possible search path for an object.
<b>Collisions</b>	The number of possible collisions.  0 (zero) means no collision or one entry. When there are more than 5 collisions, the number of collisions will be specified in ranges (for example, 6 - 10).
<b>Number of Slots</b>	The number of slots grouped by their number of collisions.  For example, if the number of collisions is 3, the search algorithm must side step a maximum of 3 times to find an object. In addition, the percentage of these slots related to all slots used is displayed.
<b>Number of Slots Totaled</b>	The same values as <b>Number of Slots</b> , but the values are totaled.

## Performance Hints

See [Performance Hints](#) in the section *Buffer Pool Statistics*.

## PF Keys and Direct Commands

On BP cache statistics screens, you can use the PF keys or SYSBPM direct commands listed in the table below. An underlined portion of a command represents its minimum abbreviation. For further commands, see [SYSBPM Direct Commands](#).

PF Key	Command	Function
PF1		Provides SYSBPM help information: see also <a href="#">Online Help</a> .
PF3	<u>EXIT</u>	Leaves the current function/screen and displays the previous screen.
PF4	LAST	Displays the SYSBPM direct command entered most recently.
PF6	FLIP	Switches the PF-key line: toggles between the display of PF1 to PF12 and PF13 to PF24.
PF8 (CLoad)	<u>DISPLAY</u> <u>CLOAD</u>	Only applies to the <b>General BP Cache Statistics</b> screen. Displays the <b>BP Cache Call Statistics</b> screen.
PF8 (CGen)	<u>DISPLAY</u> <u>CGENERAL</u>	Only applies to the <b>BP Cache Call Statistics</b> screen. Displays the <b>General BP Cache Statistics</b> screen.
PF12	<u>CANCEL</u>	Same as <a href="#">EXIT</a> .
PF15	MENU	Invokes the SYSBPM <b>Main Menu</b> .



# 64

## Select Buffer Pool

---

- Invoking Select Buffer Pool ..... 354
- Display Buffer Pools ..... 355
- Reset Buffer Pool ..... 355

SYSBPM provides functions for displaying the buffer pools defined for your Natural system environment and selecting a buffer pool other than the buffer pool used by your current Natural session at startup.

This section provides information on the **Select Buffer Pool** function and the SYSBPM direct commands provided for displaying and resetting buffer pools.

## Invoking Select Buffer Pool

---

▶ **To invoke Select Buffer Pool**

- 1 In the SYSBPM **Main Menu**, enter the following function code:

```
S
```

Or:

Enter the following SYSBPM direct command:

```
SELECT BP
```

The **Select Buffer Pool** window appears with the following information on your current buffer pool (global or local) and on all further global buffer pools currently available in your Natural system environment:

<b>BPNAME</b>	The name of the buffer pool.
<b>Type</b>	The type of the buffer pool such as Global Nat, Local Nat, Global Sort, Global DL/I, Edit (Editor) or Mon (Monitor).
<b>Status</b>	The current status.
<b>Preload</b>	The name of the preload list (if loaded).
<b>Address</b>	The address of the buffer pool.
<b>Loc</b>	The location of the buffer pool indicated by <b>HostID</b> and <b>SubsID</b> (subsystem ID).

For further information on the fields, see [SYSBPM Main Menu - Fields, Functions and Commands](#).

- 2 In column **C**, enter any character to select a buffer pool of the type Natural, DL/I or Sort (all other buffer pools can only be displayed) and press ENTER. The specified buffer pool is now defined in your current session and the contents of the buffer pool-related fields (as mentioned above) displayed on top of the SYSBPM **Main Menu** has changed accordingly.

Once you have selected a buffer pool from the **Select Buffer Pool** window, all SYSBPM functions apply to this buffer pool. Your Natural session itself, however, will continue to run with the startup buffer pool.

---

## Display Buffer Pools

---

▶ To display the buffer pools available in your Natural system

- Enter the following SYSBPM direct command:

```
DISPLAY BUFFERPOOL
```

The **Display Buffer Pools** window appears which provides the same information as described earlier for the [Select Buffer Pool window](#).

---

## Reset Buffer Pool

---

▶ To reset the buffer pool

- Enter the following SYSBPM direct command:

```
RESET BUFFERPOOL
```

SYSBPM switches back to the buffer pool originally defined for your current Natural session and the contents of the buffer pool-related fields (as mentioned in [Invoking Select Buffer Pool](#)) displayed on top of the SYSBPM **Main Menu** change accordingly.

Or:

Invoke the [Select Buffer Pool function](#) (described earlier) and select the startup buffer pool.



# 65 Blacklist Maintenance

---

▪ Maintain Blacklist .....	358
▪ List Object Sets .....	362
▪ Edit Object Set .....	362
▪ Add Object Set to Blacklist .....	365
▪ Delete Object Set from Blacklist .....	366
▪ Delete Object Set Source Object .....	367
▪ Additional Object Set Maintenance with Utilities .....	367
▪ Blacklist Maintenance in Batch Mode .....	368

This function is used to maintain a blacklist of Natural objects. In a blacklist, you can specify the Natural objects that are not to be loaded into the buffer pool; objects that are already loaded in the buffer pool are then deleted. If the BP cache is enabled, the Natural objects will also be deleted from the BP cache. The blacklist always applies to the buffer pool currently active.

In a blacklist, you can maintain individual Natural objects and/or libraries and object sets which contain multiple Natural objects. In an object set, you specify the objects not to be executed and add a single set (instead of multiple individual objects) to the blacklist. You can also combine both: maintain objects individually or by sets.

The source code of an object set is stored as a Natural source object of the type Text in the current Natural library and system file. The first line of an object set source contains the comment `**BBL**` (buffer pool blacklist).

For further information on the blacklist, see the relevant section in *Natural Buffer Pool* in the *Operations* documentation.

### ▶ To invoke Blacklist Maintenance

- In the SYSBPM **Main Menu**, enter the following function code:

```
B
```

Or:

Enter the following SYSBPM direct command:

```
BLACKLIST
```

The **Blacklist Maintenance** menu appears.

The functions provided in the **Blacklist Maintenance** menu are explained in the following section. This section also provides information on additional blacklist maintenance functions.

## Maintain Blacklist

---

This function invokes the **Maintain Blacklist** screen where you can display and maintain all Natural objects currently available in the blacklist.

### ▶ To invoke the Maintain Blacklist screen

- In the **Blacklist Maintenance** menu, enter the following function code:

```
M
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY BLACKLIST
```

The **Maintain Blacklist** screen appears with the current blacklist. Press PF7 to scroll one page backward and PF8 to scroll one page forward.

Depending on the mode set when calling the **Maintain Blacklist** function earlier during a SYSBPM session, the **Maintain Blacklist** screen appears in **Display Mode** (default when initializing SYSBPM) or **Add Mode**. Use PF9 to switch from one mode to the other.

This section covers the following topics:

- [Adding Objects](#)
- [Modifying Objects](#)
- [Deleting Objects](#)

## Adding Objects

### ▶ To add objects to the blacklist

- 1 In the **Blacklist Maintenance** menu, enter the following function code:

```
M
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY BLACKLIST
```

The **Maintain Blacklist** screen appears.

- 2 If required, press PF9 to switch to **Add Mode**.

A screen with empty input fields appears.

- 3 In the relevant input fields, enter the name of the library where the objects are stored, the names of the objects and the corresponding database IDs (DBID) and file numbers (FNR). If DBID and FNR are left blank, they will be taken from the current system file FUSER or FNAT in libraries whose names start with SYS (except the library SYSTEM).

If you need to clear the **Add Mode** screen, in the Command line, enter either of the following:

```
CLE
```

or

```
CLEAR
```

- 4 Press PF5 to confirm the addition.

Or:

In the Command line, enter either of the following:

```
UP
```

or

```
UPDATE
```

An appropriate message appears.

## Modifying Objects

### ▶ To modify objects on the blacklist

- 1 In the **Blacklist Maintenance** menu, enter the following function code:

```
M
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY BLACKLIST
```

The **Maintain Blacklist** screen appears.

- 2 If required, press PF9 to switch to **Display Mode** and obtain the list of all objects currently contained in the blacklist.
- 3 In the relevant input field(s), replace the existing entries with new values.
- 4 Press PF5 to confirm the modification.

Or:

In the Command line, enter either of the following:

```
UP
```

or

```
UPDATE
```

An appropriate message appears.

## Deleting Objects

### ▶ To delete individual objects from the blacklist

- 1 In the **Blacklist Maintenance** menu, enter the following function code:

```
M
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY BLACKLIST
```

The **Maintain Blacklist** screen appears.

- 2 If required, press PF9 to switch to **Display Mode** and obtain a list of all objects currently on the blacklist.
- 3 In column **C**, next to the object(s) required, enter the following line command:

```
DE
```

- 4 Press ENTER to confirm the deletion.

An appropriate message appears.

### ▶ To delete all objects from the blacklist

- 1 In the **Blacklist Maintenance** menu, enter the following function code:

```
M
```

Or:

Enter the following SYSBPM direct command:

```
DISPLAY BLACKLIST
```

The **Maintain Blacklist** screen appears.

- 2 Press PF2.

The **Confirm Delete** window appears.

- 3 Confirm the deletion, by entering Y (Yes).

Or:

Cancel the deletion, by entering N (No; this is the default) or by pressing PF3 without entering anything in the window.

4 Press ENTER to confirm the action.

An appropriate message appears.

## List Object Sets

---

This function invokes the **List Object Sets** screen which displays a list of all existing object sets.

### ▶ To invoke the List Object Sets screen

- In the **Blacklist Maintenance** menu, enter function code L, a library name and an object set name. Asterisk (\*) notation is also allowed for an object set name.

Or:

Enter the following SYSBPM direct command:

```
LIST SET library-name set-name
```

Asterisk (\*) notation is also allowed for *set-name*.

The **List Object Sets** screen appears with the specified set(s).

You can manipulate an object set from the **List Object Sets** screen by using any of the line commands provided to modify a set and add it to or delete it from the blacklist. For a list of possible line commands, enter a question mark (?) in the leftmost screen column (prefix information).

For a list of commands that can be entered in the command line of the **List Object Sets** screen, invoke the **Help** window by entering a question mark (?) in the command line. If required, press PF7 to scroll backward and PF8 to scroll forward in the window.

## Edit Object Set

---

This function invokes the **Edit Object Set** screen where you can create a new object set, add objects to an existing set or modify them, or delete objects from a set.

The editing functions provided on the **Edit Object Set** screen are a subset of the functions provided by the Software AG Editor described in the *Editors* documentation.

For a list of available line commands, enter a question mark (?) in the leftmost screen column (prefix information).

For a list of commands that can be entered in the command line of the **Edit Object Set** screen, invoke the **Help** window by entering a question mark (?) in the command line. If required, press PF7 to scroll backward and PF8 to scroll forward in the window.

This section covers the following topics:

- [Creating Object Sets](#)
- [Modifying Object Sets](#)

## Creating Object Sets

### ▶ To create an object set

- 1 In the **Blacklist Maintenance** menu, enter function code E and a library name. Do *not* enter the name of an object set but clear the contents (if any) of the corresponding field.

The **Edit Object Set** screen appears.

- 2 In the relevant input fields, enter the name of the library where the objects are stored, the names of the objects and the corresponding database IDs (DBID) and file numbers (FNR). If DBID and FNR are left blank, they will be taken from the current system file FUSER or FNAT in libraries whose names start with SYS (except the library SYSTEM).
- 3 To save the object set as a source object of the type Text, in the command line of the **Edit Object Set** screen, enter the following:

```
SA set-name
```

## Modifying Object Sets

This section provides instructions for adding objects to an object set, modifying existing objects and deleting objects from an object set. Note that any of these object set modifications will *not* update the current blacklist.

### ▶ To add a new object to an object set

- 1 In the **Blacklist Maintenance** menu, enter function code E, a library name and an object set name.

Or:

On the **List Object Sets** screen, in the leftmost column, next to the object set required, enter the following line command:

```
E
```

Or:

Enter the following SYSBPM direct command:

```
EDIT SET library-name set-name
```

The **Edit Object Set** screen appears with the specified object set.

- 2 Complete the input fields by entering the name of the library where the objects are stored, the names of the objects and the corresponding database IDs (DBID) and file numbers (FNR). If DBID and FNR are left blank, they will be taken from the current system file FUSER or FNAT in libraries whose names start with SYS (except the library SYSTEM).
- 3 To save the modifications, in the command line of the **Edit Object Set** screen, enter the following:

```
SA
```

▶ **To modify an object of an object set**

- 1 In the **Blacklist Maintenance** menu, enter function code E, a library name and an object set name.

Or:

On the **List Object Sets** screen, in the leftmost column, next to the object set(s) required, enter the following line command:

```
E
```

Or:

Enter the following SYSBPM direct command:

```
EDIT SET library-name set-name
```

The **Edit Object Set** screen appears with the specified object set.

- 2 In the relevant input field(s), replace the existing entries with new values.
- 3 To save the modifications, in the command line of the **Edit Object Set** screen, enter the following:

```
SA
```

▶ **To delete an object from an object set**

- 1 In the **Blacklist Maintenance** menu, enter function code E, a library name and an object set name.

Or:

On the **List Object Sets** screen, in the leftmost column, next to the object set(s) required, enter the following line command:

```
E
```

Or:

Enter the following SYSBPM direct command:

```
EDIT SET library-name set-name
```

The **Edit Object Set** screen appears with the specified object set.

- 2 In the leftmost column, next to the object required, enter the following line command:

```
D
```

The specified object is removed from the object set.

- 3 To save the modification, in the command line of the **Edit Object Set** screen, enter the following:

```
SA
```

## Add Object Set to Blacklist

This function is used to add all objects of an object set to the blacklist.

### ▶ To add an object set to the blacklist

- In the **Blacklist Maintenance** menu, enter function code *A*, a library name and an object set name.

Or:

On the **List Object Sets** screen, in the leftmost column, next to the object set(s) required, enter the following line command:

```
AC
```

Or:

On the **Edit Object Set** screen, in the command line, enter the following:

```
AC
```

Or:

Enter the following SYSBPM direct command:

```
ADD SET library-name set-name
```

A message appears confirming that the object set was added to the blacklist.



**Note:** The command `AC` denotes `ACTIVATE` which is the equivalent of **Add Object Set to Blacklist**.

## Delete Object Set from Blacklist

---

This function is used to delete all objects of an object set from the blacklist. Note that the **Delete Object Set** function will *not* delete the object set as a source object. The objects of the object set can be added to the blacklist again at any time, as described above. See also [Delete Object Set Source Object](#).

### ▶ To delete an object set from the blacklist

- In the **Blacklist Maintenance** menu, enter function code `D`, a library name and an object set name.

Or:

On the **List Object Sets** screen, in the leftmost column, next to the object set(s) required, enter the following line command:

```
DA
```

Or:

On the **Edit Object Set** screen, in the command line, enter the following:

```
DA
```

Or:

Enter the following SYSBPM direct command:

```
DELETE SET library-name set-name
```

A message appears confirming that the object set was deleted from the blacklist.



**Note:** The command `DA` denotes `DEACTIVATE` which is the equivalent of **Delete Object Set from Blacklist**.

---

## Delete Object Set Source Object

---

▶ To delete the source object of an object set

- 1 In the **Blacklist Maintenance** menu, enter function code L, a library name and an object set name.

Or:

Enter the following SYSBPM direct command:

```
LIST SET library-name list-name
```

The **List Object Sets** screen appears.

- 2 In the leftmost column, next to the object set required, enter the following line command:

```
D
```

The **DELETE** window appears.

- 3 Confirm the deletion by entering the name of the object set.

A confirmation message appears.



**Note:** Deleting the source object of an object set will *not* update the current blacklist.

---

## Additional Object Set Maintenance with Utilities

---

The Natural utilities SYSMAIN and Object Handler provide additional functions for maintaining object sets. Functions include transferring object sets between different Natural libraries and system files and/or different mainframe platforms and deleting or finding object sets in a different environment.

When using a Natural utility, an object set is treated like any other source object of the type Text.

For details, refer to the relevant sections in the *Utilities* documentation.

## Blacklist Maintenance in Batch Mode

SYSBPM blacklist maintenance functions can also be executed in batch mode as described in [Batch Processing](#).

In addition, the Natural system library SYSBPM provides the command BPMBLBAT that further facilitates batch processing of the blacklist maintenance functions add and delete all objects.

Log on to the library SYSBPM and execute the BPMBLBAT command using the syntax indicated in the diagram below. The symbols used in the diagram are explained in the section *System Command Syntax* in the *System Commands* documentation.

```
BPMBLBAT { FUNC=LOCK, BPNAME=bp-name, (object-specification) }
          { FUNC=RLS, BPNAME=bp-name }
```

This section covers the following topics:

- [Explanation of Syntax](#)
- [Examples of Input](#)

### Explanation of Syntax

The keywords and the *object-specification* clause indicated in the BPMBLBAT syntax diagram above are described in the following section. [Examples of Input](#) demonstrate the use of the keywords.

Syntax Item	Explanation						
FUNC=LOCK	Adds object names to a blacklist as described in <a href="#">Adding Objects</a> in the section <i>Maintain Blacklist</i> .						
FUNC=RLS	Deletes a complete blacklist with all object names contained in the list. This functions corresponds to delete function described under <a href="#">To delete all objects from the blacklist</a> in the section <i>Maintain Blacklist</i> . (In batch mode, you cannot delete single object names from a blacklist.)						
BPNAME	The name of the buffer pool where the blacklist is loaded.						
<i>object-specification</i>	The keywords that apply to the <i>object-specification</i> clause (see the respective syntax below) are: <table border="1" style="width: 100%; margin-top: 5px;"> <tr> <td style="width: 30%;"></td> <td></td> </tr> <tr> <td>LIB</td> <td>The name of the library where the objects are stored.</td> </tr> <tr> <td></td> <td></td> </tr> </table>			LIB	The name of the library where the objects are stored.		
LIB	The name of the library where the objects are stored.						

Syntax Item	Explanation	
	DBID FNR	The database ID (DBID) and the file number (FNR) where the objects are stored. If DBID and FNR are left blank, they will be taken from the current system file FUSER or FNAT in libraries whose names start with SYS (except the library SYSTEM).
	<i>object-name</i>	The name(s) of the object(s) to be added to the blacklist: Enter each name in a separate line in the positions 1 to 8. To indicate the end of the input, in a separate line, enter a period (.)

### Syntax of *object-specification*

Shown and explained below is the syntax that applies to *object-specification*:

```
LIB=lib-name, DBID dbid, FNR fnr [object-name]...
```

### Examples of Input

#### Example 1 - Adding Objects to a Blacklist

The example input below demonstrates how to add to a blacklist the objects A, B and C in the buffer pool NATGBP:

```
/*Job
.
.
.
*/Job

LOGON SYSBPM
BPMBLBAT
FUNC=LOCK, BPNAME=NATGBP, LIB=SAGTEST, DBID=10, FNR=32
A
B
C
.
FIN
```

### Example 2 - Deleting a Blacklist:

The example input below demonstrates how to delete a blacklist in the buffer pool NATGBP:

```
/*Job
.
.
.
*/Job
LOGON SYSBPM
BPMBLBAT
FUNC=RLS,BPNAME=NATGBP
.
FIN
```

# 66 Preload List Maintenance

---

- List Preload Lists ..... 372
- Edit Preload List ..... 373
- Generate Preload List from Buffer Pool ..... 376
- Delete Preload List ..... 377
- Additional Maintenance Functions with Utilities ..... 377

This function only applies to buffer pools of the type Natural.

**Preload List Maintenance** is used to maintain preload lists. In a preload list, you can specify the names of Natural objects that are to be loaded into the buffer pool when the buffer pool is initialized.

The source code of a preload list is stored as a Natural source object of the type Text in the Natural system library SYSBPM. The first line of a preload list source contains the comment **\*\*BPL\*\*** (buffer pool preload list).

For further information on the preload list, see the relevant section in *Natural Buffer Pool* in the *Operations* documentation.

▶ **To invoke Preload List Maintenance**

- In the **SYSBPM Main Menu**, enter the following function code:

```
P
```

Or:

Enter the following SYSBPM direct command:

```
PRELOADLIST
```

The **Preload List Maintenance** menu appears.

The functions provided in the **Preload List Maintenance** menu are explained in the following section. This section also provides information on additional preload list maintenance functions.

## List Preload Lists

---

This function invokes the **List Preload Lists** screen which displays a list of all existing preload lists.

▶ **To invoke List Preload Lists**

- In the **Preload List Maintenance** menu, enter function code **L** and the name of a preload list. Asterisk (\*) notation is also allowed for a preload list name.

Or:

Enter the following SYSBPM direct command:

```
LIST PRELOADLIST list-name
```

Asterisk (\*) notation is also allowed for *list-name*.

The **List Preload Lists** screen appears.

For a list of available line commands, enter a question mark (?) in the leftmost screen column (prefix information).

For a list of commands that can be entered in the command line of the **List Preload Lists** screen, invoke the **Help** window by entering a question mark (?) in the command line. If required, press PF7 to scroll backward and PF8 to scroll forward in the window.

## Edit Preload List

---

This function invokes the **Edit Preload List** screen where you can create a new preload list, add objects to an existing list or delete objects from it.

 **Important:** The editing functions provided on the **Edit Preload List** screen are a subset of the functions provided by the Software AG Editor (described in the *Editors* documentation). Therefore, before you start a Natural session to edit a preload list, set the Natural profile parameter EDPSIZE to a value greater than 0 (zero); see also *Profile Parameters* in the *Parameter Reference* documentation. We recommend that you set EDPSIZE to a minimum of 100.

For a list of available line commands, enter a question mark (?) in the leftmost screen column (prefix information).

For a list of commands that can be entered in the command line of the **Edit Preload List** screen, invoke the **Help** window by entering a question mark (?) in the command line. If required, press PF7 to scroll backward and PF8 to scroll forward in the window.

This section covers the following topics:

- [Creating Preload Lists](#)
- [Modifying Preload Lists](#)

### Creating Preload Lists

#### ▶ To create a preload list

- 1 In the **Preload List Maintenance** menu, specify the following:
  - Enter function code E.
  - Clear the contents of the field **Preload List Name**, that is, do *not* enter the name of a preload list.
  - In the fields **Library** and **Objects**, leave the default asterisk (\*).

The **Edit Preload List** screen appears.

- 2 In the relevant input fields, enter the name of the library where the objects are stored, the names of the objects and the corresponding database IDs (DBID) and file numbers (FNR).

If DBID and FNR are left blank, they will be taken from the current system file FUSER or FNAT in libraries whose names start with SYS (except the library SYSTEM).

The resident flag will be set to Y (Yes) in column **R** on the editing screen if no value is entered. Resident means that the object is not deleted from the buffer pool, not even if its **Use Count** changes to 0 (zero). (**Use Count** corresponds to **Current Use Count** described in *Directory Information*.)

- 3 To save the preload list as a source object of the type Text in the library SYSBPM, in the command line of the **Edit Preload List** screen, enter the following command:

```
SA set-name
```

See also [Generate Preload List from Buffer Pool](#).

## Modifying Preload Lists

### ► To add a new object to a preload list

- 1 In the **Preload List Maintenance** menu, enter function code E and the name of a preload list.

Or:

On the **List Preload Lists** screen, in the leftmost column, next to the preload list required, enter the following line command:

```
E
```

Or:

Enter the following SYSBPM direct command:

```
EDIT PRELOADLIST list-name
```

The **Edit Preload List** screen appears with the specified preload list.

- 2 Complete the input fields by entering the name of the library where the objects are stored, the names of the objects and the corresponding database IDs (DBID) and file numbers (FNR).

If DBID and FNR are left blank, they will be taken from the current system file FUSER or FNAT in libraries whose names start with SYS (except the library SYSTEM).

- 3 To save the modifications, in the command line of the **Edit Preload List** screen, enter the following:

```
SA
```

▶ **To modify an object of a preload list**

- 1 In the **Preload List Maintenance** menu, enter function code E, a library name and the name of a preload list.

Or:

On the **List Preload Lists** screen, in the leftmost column, next to the preload list required, enter the following line command:

```
E
```

Or:

Enter the following SYSBPM direct command:

```
EDIT PRELOADLIST list-name
```

The **Edit Preload List** screen appears with the specified preload list.

- 2 In the relevant input field(s), replace the existing entries with new values.
- 3 To save the modifications, in the command line of the **Edit Preload List** screen, enter the following:

```
SA
```

▶ **To delete an object from a preload list**

- 1 In the **Preload List Maintenance** menu, enter function code E, a library name and the name of a preload list.

Or:

On the **List Preload Lists** screen, in the leftmost column, next to the preload list(s) required, enter the following line command:

```
E
```

Or:

Enter the following SYSBPM direct command:

```
EDIT PRELOADLIST list-name
```

The **Edit Preload List** screen appears with the specified preload list.

- 2 In the leftmost column, next to the object required, enter the following line command:

```
D
```

The specified object is removed from the preload list.

- To save the modification, in the command line of the **Edit Preload List** screen, enter the following:

```
SA
```

## Generate Preload List from Buffer Pool

This function is used to generate a new preload list by using the names of the objects currently loaded in the buffer pool. From the objects that are currently in the buffer pool, you can select those you wish to be included in the preload list.

- ▶ To generate a preload list, use either of the following options:

- In the **Preload List Maintenance** menu, enter function code **G** and the name of a preload list. In the fields **Library**, **Objects**, **Resident**, **Use Count** and **Total Use Count**, specify the objects to be included in the list:
  - To include all objects that are currently in the buffer pool, enter an asterisk (\*) in the fields **Library**, **Objects** and **Resident**, and leave the fields **Use Count** and **Total Use Count** blank.
  - Or:  
To include specified objects in the buffer pool, in the fields described below, you can enter the following values:

<b>Library</b>	A single name or asterisk (*) notation.
<b>Objects</b>	A single name or asterisk (*) notation.
<b>Resident</b>	An asterisk (*) for all objects or Y (Yes) for all objects currently marked as resident in the buffer pool.
<b>Use Count</b>	A numeric start <i>value</i> (>), for example >10. Selects all objects with a <b>Use Count</b> greater than or equal to <i>value</i> .  This field corresponds to <b>Current Use Count</b> described in <i>Directory Information</i> .
<b>Total Use Count</b>	A numeric start <i>value</i> (>), for example >10. Selects all objects with a <b>Total Use Count</b> greater than or equal to <i>value</i> .  This field corresponds to <b>BP Total Use</b> described in <i>Directory Information</i> .

- Enter either of the following SYSBPM direct commands:

```
GENERATE PRELOADLIST list-name
```

or

```
GENERATE PRELOADLIST list-name gen-library
```

(See also the explanations of field values above).

A message appears confirming that the preload list was generated from the buffer pool.

All preload list objects will be generated as resident (entry **Y** in column **R**) by default. Choose manually, which objects you want to remove from the list.

Objects from the library SYSBPM will not be included in the generated preload list as it can be assumed that these are objects which were only loaded into the buffer pool in order to execute this function.

## Delete Preload List

---

### ▶ To delete a preload list

- 1 In the **Preload List Maintenance** menu, enter function code **L** and the name of a preload list.

Or:

Enter the following SYSBPM direct command:

```
LIST PRELOADLIST list-name
```

The **List Preload Lists** screen appears.

- 2 In the leftmost column, next to the object required, enter the following line command:

```
D
```

The **DELETE** window appears.

- 3 Confirm the deletion by entering the name of the preload list.

A confirmation message appears.

## Additional Maintenance Functions with Utilities

---

The Natural utilities SYSMAIN and Object Handler provide additional functions for maintaining preload lists. Functions include transferring preload lists between different Natural libraries and system files and/or different mainframe platforms and deleting or finding preload lists in a different environment.

When using a Natural utility, an preload lists is treated like any other source object of the type Text.

For details, refer to the relevant sections in the *Utilities* documentation.

# 67 Performance Considerations

---

- Internal Fast Locate Table ..... 380
- Searching in Steplibs ..... 381
- Reusing and Retaining Objects ..... 381
- Local versus Global Buffer Pool ..... 382

This section provides general advice on performance-related issues regarding the buffer pool and the BP cache.

For explanations of the statistics items mentioned in this section, refer to [Buffer Pool Load/Locate Statistics](#).

**Related Topic:**

- [Performance Hints](#)

## Internal Fast Locate Table

---

When a Natural object is referenced for the first time within a Natural session, the buffer pool manager creates a directory entry for this object. A directory entry is used to identify an object and contains information such as the name of the object, the library where it resides (name, database ID and file number) and the address of the object (position) in the buffer pool.

The Natural runtime system remembers names of objects that were recently executed, the libraries (name, database ID and file number) where they reside and the addresses of the corresponding buffer pool directory entries in the internal fast locate table for the duration of a Natural session.

When a user invokes an object that has been used before in the Natural session, the Natural runtime system passes the information from the internal fast locate table to the buffer pool manager, which can then bypass the Normal Locate procedure and perform a time-saving Quick Locate call (see [Quick Locate Calls](#)). This is the most efficient way to locate an object.

If the position of an object in the buffer pool has changed, the buffer pool manager will automatically schedule a Normal Locate call. The position usually changes, if an object that was deleted from the buffer pool or overwritten by another object is reloaded into the buffer pool.

The internal fast locate table contains a maximum of 128 entries. The fast locate table is reset with the LOGON system command and must be filled again using Normal Locate calls. Therefore, an application that performs a LOGON loses performance.

A high ratio of [Normal Locate Calls](#) to [Quick Locate Calls](#) indicates the use of LOGON commands in a Natural application in that each LOGON causes the internal fast locate table to be reset.

---

## Searching in Steplibs

---

The search for a Natural object through a long chain of steplib libraries has a negative impact on performance.

For each steplib, the Natural runtime issues a call to the buffer pool manager until the requested object is found. However, each needless call for a steplib library that does not contain the requested object can usually be avoided.

Depending on the setting of the `BPSFI` profile parameter (described in the *Parameter Reference* documentation), additional database calls may be required.

A long steplib chain is indicated by a high ratio of **Normal Locate Calls** (including steplib searches) to **Normal Locate Calls** (without steplib searches) that is calculated as follows:

**Normal Locate Calls: (Normal Locate Calls - STEPLIB Searches)**

### Example of a Steplib Search

When searching the default steplib chain (library SYSTEM of FUSER, library SYSTEM of FNAT), each attempt to load an object from SYSTEM (FNAT) will result in the following:

3 **Normal Locate Calls** and 2 **STEPLIB Searches**

Explanation:

3 Normal Locate calls result from searches in the current library, the library SYSTEM (FUSER) and the library SYSTEM (FNAT). There are (at least) 2 Normal Locate calls that fail, since the object is stored in neither the current library nor the library SYSTEM (FUSER). Using the above formula, results in a ratio of 3:1.

If the object is located in the current library, the result is as follows:

1 Normal Locate call and 0 (zero) **STEPLIB Searches**. Using the above formula, results in a ratio of 1:1.

---

## Reusing and Retaining Objects

---

An application that contains many Natural objects where each object is rarely ever executed has heavy impact on the performance of the buffer pool. None of the objects resides in the buffer pool for a long time and many objects have to be loaded from a system file. For performance reasons, an application should reuse objects as often as possible, for example, by moving identical source code contained in multiple objects into a single object.

You can check the use of objects with the **List Objects** function (see the section *List Objects*). For example: the **Max** column contains information on the maximum number of applications that have executed an object, and the **TotalUC** column contains the total number of Locate calls of an object that was loaded into the buffer pool.

Objects can be made resident individually by using the **List Objects** function or by specifying them in a preload list (see *Preload List Maintenance*).

## Local versus Global Buffer Pool

---

There are no general recommendations for when to use a local or a global buffer pool as different applications have different requirements. However, rules-of-thumb from experience enable us to give general advice on:

This section covers the following topics:

- [Using Local Buffer Pools](#)
- [Using a Global Buffer Pool](#)

### Using Local Buffer Pools

Performance can improve using several small local buffer pools instead of a single global buffer pool if local buffer pools can be assigned to different application environments.

For example, in CICS environments, performance usually improves when using a local buffer pool for each AOR (Application Operating Region).

### Using a Global Buffer Pool

For different batch applications that reference the same Natural objects, performance may improve if these applications use a common global buffer pool instead of a local buffer pool for each individual application. If so, there is a higher probability that the objects required by each application have already been loaded into the global buffer pool by one of the other applications.

# 68

## SYSBPM Direct Commands

---

- SORT ..... 388
- SORT BPC ..... 389

The SYSBPM direct commands described in this section can be used to directly execute SYSBPM utility functions or navigate in SYSBPM screens in online or batch mode. For additional SYSBPM direct commands that only apply to particular screens, refer to the sections where the individual SYSBPM functions are documented.

SYSBPM direct commands that refer to the BP cache or buffer pool hash table only apply to buffer pools of the type Natural.

The following table lists all SYSBPM direct commands (including subcommands) provided, the parameters that can be used with the commands, and the equivalent SYSBPM menu functions.

The SYSBPM direct commands listed below can be entered in the Command line of any SYSBPM screen. To execute a SYSBPM direct command from any other command prompt or in batch mode, the direct command must be preceded by the keyword SYSBPM, for example:

```
SYSBPM ADD BLACKLIST
```

An underlined portion of a SYSBPM command represents its minimum abbreviation. Parameter values that are required by a command are represented by letters in italics.

Command	Parameters	Function
<u>+</u>	none	Scrolls one page down in a list.
<u>-</u>	none	Scrolls one page up in a list.
<u>ADD</u> <u>BLACKLIST</u>	none	Invokes the <b>Maintain Blacklist</b> screen described in <i>Blacklist Maintenance</i> .
<u>ADD</u> <u>SET</u>	<i>library-name</i> <i>set-name</i>	Adds all objects of a specified object set to the blacklist as described in <i>Add Object Set to Blacklist</i> .
<u>BLACKLIST</u>	none	Invokes the <b>Blacklist Maintenance</b> menu.
<u>BOTTOM</u>	none	Scrolls to the end of a list.
<u>CANCEL</u>	none	Same as <b>EXIT</b> .
<u>CHECK</u> <u>HASH</u> or <u>CHECK</u> <u>HT</u>	none	Checks the BP hash table for consistency and returns the number of inconsistencies found.  See also <b>REBUILD HASH</b> .
<u>CLOSE</u> <u>BPC</u>	none	BP cache required.  Closes the BP cache. The buffer pool runs without BP cache afterwards. You can restart the BP cache by using the <b>INITIALIZE BPC</b> command.
<u>DELETE</u>	none	Deletes all objects from the buffer pool and the BP cache.  If entered on the <b>Directory Information</b> screen: see <b>DELETE</b> in <i>PF Keys and Direct Commands</i> .

Command	Parameters	Function
DELETE	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	Deletes the specified object(s) from the buffer pool and the BP cache as described in <i>Delete Objects</i> .
DELETE ALL	none	Deletes all objects from the blacklist as described in <i>Delete Object Set from Blacklist</i> .
DELETE BUFFERPOOL	none	Deletes all objects from the buffer pool only.
or	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	Deletes the specified object(s) from the buffer pool only.
DELETE BP		
DELETE BPC	none	BP cache required. Deletes all objects from the BP cache only.
	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	BP cache required. Deletes the specified object(s) from the BP cache only.
DELETE BLACKLIST	none	Invokes the <b>Maintain Blacklist</b> screen where you can delete blacklist entries as described in <i>Blacklist Maintenance</i> .
DELETE SET	<i>library-name</i> <i>set-name</i>	Deletes all objects of the specified object set from the blacklist as described in <i>Delete Object Set from Blacklist</i> .
DISPLAY ALL	none	Same as <b>DISPLAY LIST</b> .
DISPLAY BUFFERPOOL	none	See <i>Display Buffer Pools</i> in <i>Select Buffer Pool</i> .
or		
DISPLAY BP		
DISPLAY BLACKLIST	none	Invokes the <b>Maintain Blacklist</b> screen described in <i>Blacklist Maintenance</i> .
DISPLAY CDIRECTORY	none	BP cache required. Invokes the <b>Directory Information</b> screen.
DISPLAY CGENERAL	none	BP cache required. Invokes the <b>General BP Cache Statistics</b> screen described in <i>General BP Cache Statistics</i> .
DISPLAY CHASH	none	Invokes the function <b>BP Cache Hash Table Statistics</b> described in <i>BP Cache Hash Table Statistics</i> and displays the <b>Cache Hash Table Collisions</b> screen.
DISPLAY CLIST	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	BP cache required. Invokes the <b>List Objects</b> screen.  In contrast to the command <b>DISPLAY LIST</b> , this command generates a statistics report that displays data about BP cache objects at the beginning of the list.

Command	Parameters	Function
DISPLAY CLOAD	none	BP cache required.  Invokes the <b>BP Cache Call Statistics</b> screen described in <i>BP Cache Statistics</i> .
DISPLAY CSTATISTICS	none	BP cache required.  Invokes the <b>BP Cache Statistics</b> menu described in <i>BP Cache Statistics</i> .
DISPLAY DIRECTORY	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	Invokes the <b>Directory Information</b> screen.
DISPLAY FRAGMENTATION	none	Invokes the <b>Buffer Pool Fragmentation</b> screen described in <i>Buffer Pool Statistics</i> .
DISPLAY FUNCTION	none	Invokes the <b>Internal Function Usage</b> screen described in <i>Buffer Pool Statistics</i> .
DISPLAY GENERAL	none	Invokes the <b>General Buffer Pool Statistics</b> screen described in <i>Buffer Pool Statistics</i> .
DISPLAY HASH or DISPLAY HT	none	Invokes the function <b>Buffer Pool Hash Table Statistics</b> and displays the <b>Hash Table Collisions</b> screen.
DISPLAY HDIRECTORY	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	Invokes the <b>Directory Information Hex</b> screen that displays in hexadecimal format the directory information of an object.
DISPLAY HEX	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	Invokes the <b>Hexadecimal Display</b> screen that displays in hexadecimal format the source of an object.
DISPLAY LIST	<i>library-name</i> <i>object-name dbid</i> <i>fnr</i>	Invokes the <b>List Objects</b> screen.  In contrast to the command <code>DISPLAY CLIST</code> , this command generates a statistics report that displays data about buffer pool objects at the beginning of the list.
DISPLAY LOAD	none	Invokes the <b>Buffer Pool Load/Locate Statistics</b> screen described in <i>Buffer Pool Statistics</i> .
DISPLAY PERFORMANCE	none	Invokes the <b>Performance Hints</b> screen with performance-related statistics of a Natural buffer pool and BP cache as described in <i>Buffer Pool Statistics</i> and <i>BP Cache Statistics</i> .
DISPLAY STATISTICS	none	Invokes the <b>Buffer Pool Statistics</b> menu described in <i>Buffer Pool Statistics</i> .
EDIT PRELOADLIST	<i>list-name</i>	Invokes the <b>Edit Preload List</b> screen described in <i>Preload List Maintenance</i> .

Command	Parameters	Function
EDIT SET	<i>library-name</i> <i>set-name</i>	Invokes the <b>Edit Object Set</b> screen described in <i>Blacklist Maintenance</i> .
EXIT	none	Leaves the current function/screen and displays the previous screen.
FLIP	none	Switches the PF-key line: toggles between the display of PF1 to PF12 and PF13 to PF24.
GENERATE PRELOADLIST	<i>list-name</i> <i>gen-library</i>	Invokes the function <b>Generate Preload List from Buffer Pool</b> .
INITIALIZE	none  or: 1, 2, 4, 8, 12, 16	Reinitializes the buffer pool and the BP cache. If no text record size is specified, the current text record size will be taken.  Only use this function if the <b>Current Use Count</b> (see <i>Directory Information</i> ) is equal to 0 (see the warning below) or if the buffer pool has been destroyed.  <b>Caution:</b> If you try to reinitialize the buffer pool while objects are being executed by active sessions in this buffer pool, the <b>Confirm Initialization</b> window appears with the <b>Current Use Count</b> for this buffer pool (not counting the SYSBPM user himself). If <b>Current Use Count</b> is <i>not</i> equal to 0 (zero) and you enter a Y to confirm the reinitialization of the buffer, the results of the active sessions are unpredictable and Natural can even abend.
INITIALIZE BP	none  or: 1, 2, 4, 8, 12, 16	Reinitializes the buffer pool only. If no text record size is specified, the current text record size will be used.  See also the <b>Warning</b> above.
INITIALIZE BPC	none	BP cache required.  Reinitializes the BP cache only. The text record size of the BP cache is fixed (4 KB).
LAST	none	Displays the SYSBPM direct command entered most recently.
LIST PRELOADLIST	<i>list-name</i>	Invokes the <b>List Preload Lists</b> screen for the specified object as described in <i>Preload List Maintenance</i> .
LIST SET	<i>library-name</i> <i>set-name</i>	Invokes the <b>List Object Sets</b> screen for the specified library or object as described in <i>Blacklist Maintenance</i> .  Asterisk (*) is also allowed for <i>set-name</i> .
MENU	none	Invokes the SYSBPM <b>Main Menu</b> as described in <i>Invoking and Operating SYSBPM</i> .
PRELOADLIST	none	Invokes the <b>Preload List Maintenance</b> menu described in <i>Preload List Maintenance</i> .
QUIT	none	Same as EXIT.

Command	Parameters	Function
<u>REBUILD HASH</u> or <u>REBUILD HT</u>	none	Rebuilds hash tables if inconsistencies are found with <a href="#">CHECK HASH</a> .  REBUILD HASH deletes the current hash table and rebuilds a new hash table from the current buffer pool contents.
<u>RESET BUFFERPOOL</u> or <u>RESET BP</u>	none	Resets the buffer pool as described in <a href="#">Reset Buffer Pool</a> .
<u>SELECT BUFFERPOOL</u> or <u>SELECT BP</u>	none	Only applies to buffer pools of the type Natural, DL/I or Sort.  Invokes a selection list of buffer pools as described in <a href="#">Select Buffer Pool</a> .
<u>Sort BPC</u>	(syntax below)	Sorts the BP cache as described in <a href="#">Display Sorted Extract</a> .
<u>Sort BUFFERPOOL</u>	(syntax below)	Sorts the buffer pool as described in <a href="#">Display Sorted Extract</a> .
<u>STOP</u>	none	Leaves the SYSBPM utility.
<u>TOP</u>	none	Scrolls to the beginning of a list.
<u>WRITE BP</u> or <u>WRITE BPC</u> or <u>WRITE ALL</u>		Writes object directory data to a local file or a PC text file.  See also <a href="#">Write to Work File</a> .

## SORT

$\text{SORT} \left[ \left\{ \begin{array}{c} \text{BUFFERPOOL} \\ \text{BP} \end{array} \right\} \right] \left[ \left\{ \begin{array}{c} \text{OBJECT} \\ \text{TOTAL} \\ \text{LAST} \end{array} \right\} \right] \left[ \left\{ \begin{array}{c} \text{D} \\ \text{A} \end{array} \right\} \right]$
---

## SORT BPC

---

<u>S</u> ORT BPC	{	<u>O</u> BJECT	}	[	{	<u>D</u>	}	]
		<u>I</u> OTAL				<u>A</u>		
		<u>L</u> AST						



# 69 Batch Processing

---

- Related Topics ..... 392

The functions provided by the SYSBPM utility can also be executed in batch mode.

For this purpose, we recommend that you use the SYSBPM Application Programming Interface USR4340N or USR0340N described in the section [Application Programming Interfaces](#).

You can also use the SYSBPM utility in batch by simulating the online input command sequence.

Since SYSBPM uses functionality of the Software AG Editor, when writing your batch job, observe the following instructions:

- Set the profile parameter `EDPSIZE`. As an alternative, set the profile parameter `BPI` with `TYPE=EDIT` for the Software AG Editor buffer pool and define the editor work file in the batch job
- Note that the field **Code** is not available on every SYSBPM screen.
- Note that you may have to skip input fields in order to position the cursor in the Command line input field for entering direct commands.
- Simulate PF keys by using the terminal commands `%K` to navigate through the SYSBPM utility. For example, use `%K3` to leave the **List Objects** screen.
- Use the continuation character defined with the session parameter `CF` (default is `%`) to enter more than one line of input for a map.

Listed below are the topics and Natural documentation sources that refer to the instructions mentioned above.

## Related Topics

Topic	Documentation
<a href="#">Blacklist Maintenance in Batch Mode</a>	<i>SYSBPM Utility</i>
<a href="#">SYSBPM Direct Commands</a>	<i>SYSBPM Utility</i>
<i>EDPSIZE - Size of Software AG Editor Auxiliary Buffer Pool</i>	<i>Parameter Reference</i>
<i>BPI - Buffer Pool Initialization</i>	<i>Parameter Reference</i>
<i>Installing the Software AG Editor</i>	<i>Installation</i>
<i>%K and %KP - Simulate PF- and PA-Key</i>	<i>Terminal Commands</i>
<i>EDBP - Software AG Editor Buffer Pool Definitions</i>	<i>Parameter Reference</i>
<i>Editor Buffer Pool</i>	<i>Operations</i>
<i>Editor Work File</i>	<i>Operations</i>
<i>Natural in Batch Mode</i>	<i>Operations</i>
<i>Using the INPUT Statement in Non-Screen Modes</i>	<i>Statements</i>
<i>Using the INPUT Statement in Batch Mode</i>	<i>Statements</i>

# 70 Application Programming Interfaces

---

This section describes the Application Programming Interfaces (APIs) USR0340N, USR0341N and USR4340N, which are used for handling Natural objects currently loaded in the buffer pool and/or BP cache. The APIs are supplied in the Natural system library SYSEXT.

For further information on these APIs, see the following:

- The relevant Natural source object of the type Text and the example programs in the Natural system library SYSEXT.
- The SYSBPM functions referenced in the table below.

## Related Topic:

- [SYSBPM Batch Processing](#)

API	Functionality
USR0340N	<ul style="list-style-type: none"><li>■ Deletes objects from the buffer pool and/or BP cache.</li><li>■ Marks objects as resident.</li><li>■ Removes the resident flag from objects.</li><li>■ Reads object directory information.</li><li>■ Retrieves general buffer pool statistics and buffer pool load/locate statistics.</li></ul> <p>Corresponding SYSBPM functions:</p> <ul style="list-style-type: none"><li><a href="#">Delete Objects</a></li><li><a href="#">List Objects</a></li><li><a href="#">Directory Information</a></li><li><a href="#">General Buffer Pool Statistics</a></li><li><a href="#">Buffer Pool Load/Locate Statistics</a></li></ul>

API	Functionality
USR0341N	<p>Collects garbage to clean up the buffer pool by removing objects which are no longer needed.</p> <p>Selection criteria for specified objects are the relative age of an object. Relative age is the time the object has been loaded in the buffer pool which calculates from <b>BP Last Action</b> date. Minimum age is 30 minutes.</p> <p>See also the SYSBPM function <a href="#">List Objects</a>.</p>
USR4340N	<p>We recommend that you use this interface for batch processing instead of using the SYSBPM utility in batch.</p> <p>Lists objects loaded in the buffer pool and/or BP cache sorted by <b>Object Size</b> (ObjSize), <b>Total Use Count</b> (TotalUC) or <b>BP Last Action</b>. <b>BP Last Action</b> only applies to the buffer pool.</p> <p>Corresponding SYSBPM functions:</p> <p><a href="#">Display Sorted Extract</a> <a href="#">List Objects</a></p>
USR4341N	<p>Maintains a blacklist of Natural objects, which are not to be executed and loaded into the buffer pool.</p> <p>See also the SYSBPM function <a href="#">Blacklist Maintenance</a>.</p>

# 71 SYSCP Utility - Code Page Administration

---

▪ Invoking and Terminating SYSCP .....	396
▪ Code Page Maintenance of Sources .....	398
▪ All Code Pages .....	410
▪ Unicode Properties .....	415

The SYSCP utility is used to obtain information on code pages and ICU (International Components for Unicode) data files available in the current Natural mainframe environment. In addition, you can use the SYSCP utility to change the code page assignment of a source object or convert code pages for a source object.

This helps avoid problems that can occur when a code page is not defined or enabled in Natural or when source objects are converted to an incorrect code page or Unicode format.

For detailed information on how Natural supports Unicode and code pages and Unicode-specific items, see the descriptions and presentations in the SYSEXV application and *Related Topics* below.



**Note:** The use of the SYSCP utility can be controlled by Natural Security. For detailed information, see the section *SYSCP - Code Page Administration - Utility Profiles* in the *Natural Security* documentation.

### Related Topics:

- *Unicode and Code Page Support: Natural* documentation
- Unicode: Unicode Consortium at web site at <http://www.unicode.org/>
- ICU: IBM ICU Documentation at web site <http://www-01.ibm.com/software/globalization/icu/index.jsp>
- IBM Converter Explorer documentation at web site <http://demo.icu-project.org/icu-bin/convexp>

## Invoking and Terminating SYSCP

---

Instructions for invoking and terminating the SYSCP utility and performing a function are provided in the following section.

### ▶ To invoke the SYSCP utility

- Enter the following system command:

```
SYSCP
```

A SYSCP menu similar to the example below appears:

```
11:19:07          ***** NATURAL SYSCP UTILITY *****          2007-06-13
User SAG          - Menu -          ICU Version 3.6
                                     Unicode Version 5.0

                                     Function

                                     _ Code Page Maintenance of Sources
                                     _ All Code Pages
                                     _ Unicode Properties
                                     _ Help
                                     _ Exit

Command ==>

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

The current ICU and Unicode versions are indicated at the top of the screen.

The functions contained in the menu are explained in the remainder of this documentation.

▶ **To execute a SYSCP function**

- In the SYSCP menu, place the cursor in the input field next to the required function and press ENTER.

Or:

In the SYSCP menu, in the input field next to the required function, enter any character and press ENTER.



**Note:** In the Command line of any SYSCP utility screen, you can enter any Natural system command. A system command terminates the SYSCP utility.

▶ **To terminate SYSCP**

- Press PF3 or PF12.

Or:

From the SYSCP menu, choose **Exit**.

## Code Page Maintenance of Sources

---

The **Code Page Maintenance of Sources** functions are used to list the code page information of source objects contained in a Natural library, change code page assignments of source objects and convert code pages for source objects.

All code page maintenance functions reference the standard IANA name (see also **Cmd** in *All Code Pages*); you cannot use a code page name other than IANA when you execute a code page maintenance function.

The results of a code page maintenance function are output on a report screen, which is described in *Function Result Report*.

When you invoke **Code Page Maintenance of Sources**, a maintenance menu similar to the example below appears:

```

07:34:21          ***** NATURAL SYSCP UTILITY *****          2009-08-27
User SAG          - Code Page Maintenance of Sources -

                Code  Function

                L   List Code Page Information of Sources
                C   Check Conversion of Unassigned Sources
                A   Assign Code Page Information to Sources
                K   Check Conversion of Assigned Sources
                T   Convert to Different Code Page
                R   Remove Code Page Information from Sources
                ?   Help
                .   Exit

Code ..... _

Library ... SYSTEM__      DBID ..... _____      FNR ..... _____
                          Password ..          Cipher ..

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help           Exit                               Canc

```

The fields and functions contained in the **Code Page Maintenance of Sources** menu and the options and features provided by the functions are explained in the following section:

- [Code Page Maintenance of Sources Menu](#)
- [List Code Page Information of Sources](#)
- [Check Conversion of Unassigned Sources](#)
- [Assign Code Page Information to Sources](#)
- [Check Conversion of Assigned Sources](#)
- [Convert to Different Code Page](#)
- [Remove Code Page Information from Sources](#)
- [Name Specification](#)
- [Object Selection List](#)

- [Function Result Report](#)

## Code Page Maintenance of Sources Menu

The fields contained in the **Code Page Maintenance of Sources** menu are explained in the following table:

Field	Explanation
<b>Code</b>	The code to be entered for the function to be executed, for example, L for <b>List Code Page Information of Sources</b> .
<b>Library</b>	The name of the Natural library that contains the source objects for which to execute a code page maintenance function.  The name entered by default is the name of the current library as specified with the system variable *LIBRARY - ID (see the <i>System Variables</i> documentation).
<b>DBID</b>	The database ID (DBID) of the Natural system file where the specified library is stored.  If no value (or 0) is specified, the current FUSER or FNAT system file is used.
<b>FNR</b>	The file number (FNR) of the Natural system file where the specified Natural library is stored.  If no value (or 0) is specified, the file number of the current FUSER or FNAT system file is used.
<b>Password</b>	If the specified system file is password protected, you must supply the appropriate 8-character Adabas password.
<b>Cipher</b>	If the specified system file is enciphered, you must supply the appropriate 8-digit Adabas cipher code.

## List Code Page Information of Sources

This function is used to list the code page information for all source objects contained in a Natural library as shown in the following example:

```

14:50:26          ***** NATURAL SYSCP UTILITY *****          2006-10-19
User SAG          - List Code Page Information of Sources -
                                     Listed Library SAGTEST
Cmd  Name      Code Page      Type
---  *-----*-----*-----*
___  LDA1      IBM01147      Local
___  LDA2      IBM01147      Local
___  LDA3      IBM037        Local
___  LDA4      IBM01147      Local
___  MAP1      IBM01147      Map
___  MAP2      IBM037        Map
___  MAP3      IBM01147      Map
___  MAP4      Map
___  PGM1      Program

```

___	PGM2	IBM01147	Program
___	PGM3	IBM01147	Program
___	PROG1	IBM01147	Program
___	PROG2		Program
___	PROG3	IBM037	Program
___	PROG4	IBM01147	Program

Command ==>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---

Help Exit -- - + ++ Canc

The fields and columns contained in the **List Code Page Information of Sources** screen are explained in the following table:

Field/Column	Explanation
<b>Listed Library</b>	See <a href="#">Library</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Cmd</b>	Input field for the following line command to be executed for a selected source object:  LD Display object directory information.  This line command corresponds to the command <code>LIST DIRECTORY object-name</code> described in <i>Displaying Directory Information</i> in the <i>System Commands</i> documentation.
<b>Name</b>	The name of the source object.
<b>Code Page</b>	The code page information (IANA name) of the source object. This column appears empty for a source object that is not assigned a code page.
<b>Type</b>	The type of Natural object such as a program or a map.

### Filtering Objects

You can shorten the list of objects displayed on the **List Code Page Information of Sources** screen by specifying selection criteria.

#### ▶ To specify selection criteria

- 1 In the input fields that appear underneath the column headings **Name** and **Code Page**, replace the default asterisk (\*) with any of the input values listed in [Name Specification](#).

- 2 In the input field underneath the column heading **Type**, replace the default asterisk (\*) with one or more (maximum is 11) of the following type codes without a separator character:

Code	Object Type	Code	Object Type
P	Program	A	Parameter data area
N	Subprogram	G	Global data area
S	Subroutine	L	Global data area
M	Map	C	Copycode
H	Helproutine	T	Text
M	Macro	R	Report
7	Function	Z	Recording
3	Dialog	4	Class
5	Processor		
*	All Types		

### Check Conversion of Unassigned Sources

This function is used to check whether an unassigned source object can be converted to a code page.

An unassigned source object is an object without code page information which was originally saved under a Natural version where code page information was not yet supported. Since no code page information is provided, you need to decide which code page to specify for the source object to be checked for conversion. This depends on the character set used in the source.

If you invoke the **Check Conversion of Unassigned Objects** function, a screen similar to the example below appears:

```

14:56:51          ***** NATURAL SYSCP UTILITY *****          2006-10-19
User SAG          - Check Conversion of Unassigned Sources -

Check if source objects that have no code page information can be
converted from a given code page to a target code page.

Use selection list .. Y

Source code page .... IBM01147_____
Target code page .... IBM01140_____
Object name ..... *_____

```

```

Library ..... SAGTEST_      DBID ..... 10____      FNR ..... 32____
                                Password ..              Cipher ..

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help           Exit                                 Canc

```

The fields contained in the **Check Conversion of Unassigned Objects** screen are explained in the following table:

Field	Explanation
<b>Use selection list</b>	Specifies whether selective processing or automated processing is used for the specified source objects: see <a href="#">Object Selection List</a> .
<b>Source code page</b>	The name of the code page to be used to check whether the specified source objects (to which no code pages are yet assigned) can be converted from this code page to the code page entered in the <b>Target code page</b> field. If the conversion check is successful, the code page specified in <b>Target code page</b> can be used with the <a href="#">Assign Code Page Information to Sources</a> function.  The default name entered is the IANA name as returned by the *CODEPAGE system variable (see the <i>System Variables</i> documentation).
<b>Target code page</b>	The name of the code page to check for conversion of the specified unassigned source objects.  The default name entered is the IANA name as returned by the *CODEPAGE system variable (see the <i>System Variables</i> documentation).
<b>Object name</b>	The name of a single source object or a range of names to be processed: see <a href="#">Name Specification</a> for valid input values.
<b>Library</b>	See <a href="#">Library</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>DBID</b>	See <a href="#">DBID</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>FNR</b>	See <a href="#">FNR</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Password</b>	See <a href="#">Password</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Cipher</b>	See <a href="#">Cipher</a> in <i>Code Page Maintenance of Sources Menu</i> .

## Assign Code Page Information to Sources

This function is used to assign a code page to an unassigned source object. The source code of this object is *not* converted to the specified code page.

You can also use the function to change the code page information for a source object to which a code page is already assigned. In this case, only the code page name (IANA name) changes; the source code of this object is *not* converted.

The fields contained in the **Assign Code Page Information to Sources** screen are explained in the following table:

Field	Explanation
<b>Use selection list</b>	Specifies whether selective processing or automated processing is used for the specified source objects: see <a href="#">Object Selection List</a> .
<b>Forced assignment</b>	Specifies whether to process source objects that have already code page information or source objects without code page information.  Possible values are:  Y Yes. Forced assignment is activated: the code page information changes to the specified code page for objects that have already code page information.  N No. Forced assignment is deactivated (this is the default setting): the specified code page is only assigned to objects that have no code page information.
<b>Code page</b>	The name of the code page to be assigned to the specified source objects.  The default name entered is the IANA name as returned by the *CODEPAGE system variable (see the <i>System Variables</i> documentation).
<b>Object name</b>	The name of a single source object or a range of names to be processed: see <a href="#">Name Specification</a> for valid input values.
<b>Library</b>	See <a href="#">Library</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>DBID</b>	See <a href="#">DBID</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>FNR</b>	See <a href="#">FNR</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Password</b>	See <a href="#">Password</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Cipher</b>	See <a href="#">Cipher</a> in <i>Code Page Maintenance of Sources Menu</i> .

## Check Conversion of Assigned Sources

This function is used to test whether an assigned source object can be converted from its current code page (as entered in the object directory information) to another code page.

An assigned source object is an object which has code page information.

The fields contained in the **Check Conversion of Assigned Sources** screen are explained in the following table:

Field	Explanation
<b>Use selection list</b>	Specifies whether selective processing or automated processing is used for the selected source objects: see <a href="#">Object Selection List</a> .
<b>Current code page</b>	The name of the code page or a range of names to be used as an object selection criterion: see <a href="#">Name Specification</a> for valid input values.  The default setting is asterisk (*) indicating all code pages.
<b>New code page</b>	The name of the code page to check for conversion of the specified assigned source objects.  The default name entered is the IANA name as returned by the *CODEPAGE system variable (see the <i>System Variables</i> documentation).
<b>Object name</b>	The name of a single source object or a range of names to be processed: see <a href="#">Name Specification</a> for valid input values.
<b>Library</b>	See <a href="#">Library</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>DBID</b>	See <a href="#">DBID</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>FNR</b>	See <a href="#">FNR</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Password</b>	See <a href="#">Password</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Cipher</b>	See <a href="#">Cipher</a> in <i>Code Page Maintenance of Sources Menu</i> .

## Convert to Different Code Page

This function is used to convert an assigned source object from its current code page (as entered in the object directory information) to another code page. You cannot convert an unassigned source object.

The fields contained in the **Convert to Different Code Page** screen are explained in the following table:

Field	Explanation
<b>Use selection list</b>	Specifies whether selective processing or automated processing is used for the specified source objects: see <a href="#">Object Selection List</a> .
<b>Current code page</b>	The name of the code page or a range of names to be used as an object selection criterion: see <a href="#">Name Specification</a> for valid input values.  The default setting is asterisk (*) indicating all code pages.
<b>New code page</b>	The name of the code page into which to convert the specified source objects.  The default name entered is the IANA name as returned by the *CODEPAGE system variable (see the <i>System Variables</i> documentation).
<b>Object name</b>	The name of a single source object or a range of names to be processed: see <a href="#">Name Specification</a> for valid input values.
<b>Library</b>	See <a href="#">Library</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>DBID</b>	See <a href="#">DBID</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>FNR</b>	See <a href="#">FNR</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Password</b>	See <a href="#">Password</a> in <i>Code Page Maintenance of Sources Menu</i> .
<b>Cipher</b>	See <a href="#">Cipher</a> in <i>Code Page Maintenance of Sources Menu</i> .

## Remove Code Page Information from Sources

This function is used to remove the code page information (as entered in the object directory) from an assigned source object.



**Caution:** Be aware that the code page information is removed without conversion of the source code.

The fields contained in the **Remove Code Page Information from Sources** screen are explained in the following table:

Field	Explanation
<b>Use selection list</b>	Specifies whether selective processing or automated processing is used for the specified source objects (see <a href="#">Object Selection List</a> ).
<b>Current code page</b>	The name of the code page or a range of names to be used as an object selection criterion: see <a href="#">Name Specification</a> for valid input values. The default setting is the IANA name as returned by the *CODEPAGE system variable (see the <i>System Variables</i> documentation).
<b>Object name</b>	The name of a single source object or a range of names to be processed: see <a href="#">Name Specification</a> for valid input values.
<b>Library</b>	See <a href="#">Library</a> in <i>Code Page Maintenance of Sources Menu</i> .

Field	Explanation
DBID	See <b>DBID</b> in <i>Code Page Maintenance of Sources Menu</i> .
FNR	See <b>FNR</b> in <i>Code Page Maintenance of Sources Menu</i> .
Password	See <b>Password</b> in <i>Code Page Maintenance of Sources Menu</i> .
Cipher	See <b>Cipher</b> in <i>Code Page Maintenance of Sources Menu</i> .

## Name Specification

You can specify a name or a range of names as a selection criterion.

In the list of options below, *value* is any combination of one or more characters:

	Input	Items Selected
	<i>value</i>	All items with names equal to <i>value</i> .
	*	All items.
	?	All items with any single character for each question mark (?) entered.
Leading characters	<i>value</i> *	All items with names that start with <i>value</i> . Example: AB* Selected: AB, AB1, ABC, ABEZ Not selected: AA1, ACB
Wildcard	<i>value</i> ?	A wildcard. All items with names that start with <i>value</i> and end with any single character for each question mark (?) entered. Example: ABC? Selected: ABCA, ABCZ Not selected: AXC, ABCAA
	<i>value</i> ? <i>value</i> ? <i>value</i> * <i>value</i> ? * <i>value</i> ? <i>value</i> *	All items that match <i>value</i> combined with asterisk (*) and question mark (?) in any order. Example: A?C*Z Selected: ABCZ, AXCBBBZ, AN CZ Not selected: ACBZ, ABDEZ, AXCBBBZA
Start value	<i>value</i> >	All items with names greater than or equal to <i>value</i> . Example: AB> Selected: AB, AB1, BBB, ZZZZZZ Not selected: AA1, AAB
End value	<i>value</i> <	An end value: All items with names less than or equal to <i>value</i> . Example: AX< Selected: AB, AWW, AX Not selected: AXA, AY

## Object Selection List

You can set the **Use selection list** option to determine whether selective processing or automated processing is used for a maintenance function. If selective processing is used, a selection list of the specified objects is displayed on a selection screen before executing the function.

The **Use selection list** option does not apply to the **List Code Page Information of Sources** function.

Possible settings of **Use selection list** are as follows:

- Y Yes.  
Selective processing is activated (this is the default setting): a selection list of all source objects that meet the specified selection criteria appears. You can then select the objects to be processed from this list.
- N No.  
Selective processing is deactivated and the function is executed immediately for all source objects that meet the specified selection criteria.

An object selection list looks similar to the example shown below:

```

16:28:43          ***** NATURAL SYSCP UTILITY *****          2006-10-19
User SAG          - Check Conversion of Assigned Sources -
  Target code page IBM01140
Cmd  Name        Code Page          Message
-----
___  LDA1         IBM01147
___  LDA2         IBM01147
___  LDA3         IBM01147
___  LDA4         IBM01147
___  MAP1         IBM01147
___  MAP2         IBM01147
___  MAP3         IBM01147
___  PGM2         IBM01147
___  PGM3         IBM01147
___  PROG1        IBM01147
___  PROG3        IBM01147
___  PROG4        IBM01147

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help      Exit      All X      Canc

```

The fields and columns contained in an object selection screen are described in the following table:

Field/Column	Explanation				
<b>Target code page</b>	The code page to be used to check or perform a source-object assignment or conversion.				
<b>Cmd</b>	Input field for either of the following line commands to be executed for a selected source object: <table border="1" data-bbox="430 483 1474 840"> <tbody> <tr> <td>EX or X</td> <td>Execute the maintenance function. You can press PF5 if you want to issue the line command to all objects in one go.</td> </tr> <tr> <td>LD</td> <td>Display object directory information. This line command corresponds to the command LIST DIRECTORY <i>object-name</i> described in <i>Displaying Directory Information</i> in the <i>System Commands</i> documentation.</td> </tr> </tbody> </table>	EX or X	Execute the maintenance function. You can press PF5 if you want to issue the line command to all objects in one go.	LD	Display object directory information. This line command corresponds to the command LIST DIRECTORY <i>object-name</i> described in <i>Displaying Directory Information</i> in the <i>System Commands</i> documentation.
EX or X	Execute the maintenance function. You can press PF5 if you want to issue the line command to all objects in one go.				
LD	Display object directory information. This line command corresponds to the command LIST DIRECTORY <i>object-name</i> described in <i>Displaying Directory Information</i> in the <i>System Commands</i> documentation.				
<b>Name</b>	The name of the source object that meets the specified selection criteria.				
<b>Code Page</b>	The current code page information of the source object.				
<b>Message</b>	This column only contains text when you have finished executing the maintenance function. In this case, the column contains a message that indicates the processing status of the source object. See also <a href="#">Function Result Report</a> .				

## Function Result Report

After a maintenance function has finished executing, the processing results are shown on a report screen. A report screen looks similar to an object selection screen an [example](#) of which is shown in *Object Selection List*.

The fields and columns contained in a result report screen are explained in the following table:

Field/Column	Explanation
<b>Target code page</b>	The code page used to check or perform a source-object assignment or conversion.
<b>Cmd</b>	Input not possible.
<b>Name</b>	The name of the source object that meets the specified selection criteria.
<b>Code Page</b>	The current code page information of the source object.
<b>Message</b>	This column contains a message that indicates the processing status of the source objects selected for processing. The messages indicate successful execution of a function or possible error reasons.  Possible messages are:

Field/Column	Explanation
	Assignment possible The source object can be assigned to the specified code page.
	Conversion error, at least one code point not translated. The source object cannot be assigned or converted to the specified code page.
	Code page assigned The source object has been assigned to the specified code page.
	Conversion possible The source object can be converted to the specified code page.
	Code page converted The source object has been converted from its current code page to another code page.
	Not converted The source object has not been converted to the specified code page because it is already encoded in this code page.

## All Code Pages

This function is used to list all code pages available in your current Natural environment as shown in the following example:

```

17:21:36          ***** NATURAL SYSCP UTILITY *****          2007-08-02
User SAG              - All Code Pages -

Cmd Stat Name                                               Units
-----
_   D   UTF-8                                               1 - 3
_   D   UTF-16                                              2 - 2
_   D   UTF-16BE                                            2 - 2
_   D   UTF-16LE                                            2 - 2
_   D   UTF-32                                              4 - 4
_   D   UTF-32BE                                            4 - 4
_   D   UTF-32LE                                            4 - 4
_   D   UTF16_PlatformEndian                               2 - 2
_   D   UTF16_OppositeEndian                              2 - 2
_   D   UTF32_PlatformEndian                              4 - 4
_   D   UTF32_OppositeEndian                              4 - 4
_   D   UTF-7                                              1 - 4
_   D   IMAP-mailbox-name                                  1 - 4
_   D   SCSU                                               1 - 3

Command ==>>>
    
```

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

You can use the following PF keys:

- PF8 (or ENTER) scrolls down one page in the list.
- PF7 scrolls up one page in the list.
- PF5 sorts the list in ascending order by code page name. Depending on the size of the list, you may have to increase the size of the sort buffer by using the SORT profile parameter as described in *SORT - Control of Sort Program* in the *Parameter Reference* documentation.

The columns contained in the **All Code Pages** screen are explained in the following table:

Column	Explanation	
Cmd	Input field for one of the following line commands to be executed for the selected code page:	
	N	Display all names used for the code page:  The IANA (Internet Assigned Numbers Authority) name is the standard and unambiguous name of the code page. The IANA name is used by Natural as the default code page name (see the CP profile parameter described in the <i>Parameter Reference</i> documentation) for conversions to and from Unicode. The IANA name is returned by the *CODEPAGE system variable (see the <i>System Variables</i> documentation).  CCSID (Coded Character Set Identifier) denotes the character set as identified by IBM.  Alias names: one or more alternate names for the code page.
	C	Display all code points of the selected code page: see <a href="#">Code Point List</a> below.
	T	Invoke a window to test code point conversion to and from Unicode: see <a href="#">Test Conversion</a> below.
Stat	All code pages to be used during a Natural session must be predefined and enabled in the NATCONFIG module.	
	This column shows the NATCONFIG status of the code page:	
	E	Code page is defined in the NATCONFIG module and is enabled.
	D	Code page is defined in NATCONFIG but is disabled.

Column	Explanation
	N
	Code page is not defined in NATCONFIG.
	For detailed information on the NATCONFIG module, refer to <i>Natural Configuration Tables in the Operations</i> documentation.
<b>Name</b>	The internal ICU name.
<b>Units</b>	The code units (minimum and maximum numbers of bytes) assigned to the code points.

This section covers the following topics:

- [Code Point List](#)
- [Test Conversion](#)

### Code Point List

This function is used to list all code points of the selected code page as shown in the following example:

```

13:38:33          ***** NATURAL SYSCP UTILITY *****          2007-08-06
+----- Code Points of UTF-8 -----+
! CP: 00000000  U: 0000      NULL                               !
! CP: 00000001  U: 0001  ?   START OF HEADING                  !
! CP: 00000002  U: 0002  ?   START OF TEXT                      !
! CP: 00000003  U: 0003  ?   END OF TEXT                        !
! CP: 00000004  U: 0004  ?   END OF TRANSMISSION                !
! CP: 00000005  U: 0005  ?   ENQUIRY                           !
! CP: 00000006  U: 0006  ?   ACKNOWLEDGE                        !
! CP: 00000007  U: 0007  ?   BELL                               !
! CP: 00000008  U: 0008  ?   BACKSPACE                          !
! CP: 00000009  U: 0009  ?   CHARACTER TABULATION               !
! CP: 0000000A  U: 000A  ?   LINE FEED (LF)                     !
! CP: 0000000B  U: 000B  ?   LINE TABULATION                    !
! CP: 0000000C  U: 000C  ?   FORM FEED (FF)                     !
! CP: 0000000D  U: 000D  ?   CARRIAGE RETURN (CR)               !
! CP: 0000000E  U: 000E  ?   SHIFT OUT                          !
! CP: 0000000F  U: 000F  ?   SHIFT IN                           !
+-----+
-  D   ibm-912_P100-1995                                     1 - 1
-  D   ibm-913_P100-2000                                     1 - 1
-  N   ISCII,version=0                                       1 - 4
-  N   ISCII,version=1                                       1 - 4

Command ==>
```

```

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help      Exit  LByte Prop  --  -  +          <<  >      Canc

```

The list contains the following information:

- The byte sequence of the code page code points (CP).
- The byte sequence of the corresponding Unicode code points (U).
- The Unicode character. If the character cannot be interpreted by the current terminal emulation, the substitution character (as defined in the code page; here: ?) is displayed instead.
- The normative name of the Unicode character.

The PF keys provided for each code point list are explained in the following table:

PF Key	Function
PF4	Not applicable to a code page with a 1-byte unit as the maximum.  Opens the <b>Leading Bytes of Code Point</b> window (see the relevant section) in which you can enter the byte range you want to view.  Press PF3 or ENTER to confirm your current input of leading bytes and to close the window.  Press PF12 to cancel your current input and to close the window.
PF5	Invokes the <b>Unicode Properties</b> screen (see the relevant section) for the list item where the cursor is placed.
PF6	Resets the first (non-leading) byte of the byte range to the hexadecimal value 0x00.
PF7	Scrolls up one page in the selected byte range (see also <i>Specifying Leading Bytes</i> ).  In a UTF-16 or UTF-32 code page, you can scroll through all byte ranges.
PF8 (or ENTER)	Scrolls down one page in the selected byte range (see also <i>Specifying Leading Bytes</i> ).  In a UTF-16 or UTF-32 code page, you can scroll through all byte ranges.
PF10	Moves to the leftmost screen position.
PF11	Moves to the right of the screen.

### Specifying Leading Bytes

This function does not apply to a code page with a 1-byte unit as the maximum.

You can use the **Leading Byte of Code Point** window to view the byte range (hexadecimal values 0x00 to 0xFF) of a particular leading byte for a code point.

In the following example of a UTF-8 code page, the hexadecimal values 0x22 and 0x32 have been entered as the leading bytes:

```
+----- Leading Bytes of Code Point -----+
!
!   Maximum number of bytes .. 3           !
!
!   Enter leading bytes ..... 00 22 32 00  !
!
!                                           !
!                                           !
!                                           !
+-----+
```

After pressing PF3 (or ENTER) the code point list then displays the bytes from hexadecimal 0x00223200 to 0x002232FF.



**Note:** For byte-swapped code pages such as UTF-16LE or UTF-32LE, the bytes are read and displayed in a reversed byte order.

## Test Conversion

You can test code-point conversion from a selected code page to the default code page (value of \*CODEPAGE) defined with the CP profile parameter:

- from an alphanumeric character string to Unicode code points and vice versa, or
- from hexadecimal values to Unicode code points and vice versa.

The example below shows the conversion window of a code page (here: ibm-1140\_P100-1997) which contains the following information:

- the number of byte units (minimum and maximum numbers of bytes) assigned to the code points,
- an alphanumeric character string and its equivalent hexadecimal values and
- the corresponding Unicode code points.

```
+----- Test Conversion of ibm-1140_P100-1997 -----+
!
! Code page units .. 1 - 1   (minimum, maximum of bytes) !
!
!
! Code Page Characters      !
! Alphanumeric .. ABC      !
! Hexadecimal ... C1 C2 C3 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 !
+-----+
```

```

!           40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 !
!
! Unicode                                         !
! Code points ... 0041 0042 0043 0020 0020 0020 0020 0020 0020 0020 !
!           0020 0020 0020 0020 0020 0020 0020 0020 0020 0020 0020 !
!           0020 0020 0020 0020 0020 0020 0020 0020 0020 0020 0020 !
!           0020 0020 0020 0020 0020 0020 0020 0020 0020 0020 0020 !
!
!
!
!
+-----+

```

### ► To convert a character or code point

- 1 Activate the field where you want to enter the literal string or code unit sequence to be converted:

Press PF6 to enter a literal string in the **Alphanumeric** field (default input field).

Or:

Press PF7 to enter hexadecimal values in the **Hexadecimal** field.

Or:

Press PF8 to enter Unicode code points in the **Unicode** field.

- 2 Press ENTER.

The value entered in one of the fields is converted to its equivalent code points or literal string.

## Unicode Properties

This function is used to display whether a Unicode character property is true (yes) or false (no) for a character contained in the default code page (value of \*CODEPAGE) as shown in the example of the letter A in code page IBM01140 below:

```

14:43:19          ***** NATURAL SYSCP UTILITY *****          2008-09-23
User SAG          - Unicode Properties -

Default code page ... IBM01140

Alpha character ..... A      C1      hexadecimal      Substitution .. ? 3F
Unicode code point .. 0041

```

```
Unicode char. name .. LATIN CAPITAL LETTER A

Alphabetic ..... yes           Control ..... no
Alphanumeric ..... yes        Space ..... no
Lower case ..... no           Whitespace ..... no
Upper case ..... yes          Blank ..... no
Digit ..... no                Punctuation .... no
Hexadecimal ..... yes         Combining ..... no
Graphic ..... yes             Surrogate ..... no
Printable ..... yes           Right to left .. no

Command ==>

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

In the **Alpha character** field, you can enter the character whose properties you want to view. Press PF5 if you want to enter a Unicode code point.

For explanations of the Unicode character properties displayed on the screen, refer to Unicode Consortium's documentation *Unicode Character Database* at web site <http://www.unicode.org/Public/4.1.0/ucd/UCD.html>.

# 72

## SYSEDT Utility - Editor Buffer Pool Administration

---

▪ Defining a Natural Security Library Profile .....	418
▪ Invoking SYSEDT and Executing a Function .....	418
▪ General Information .....	420
▪ Generation Parameters .....	421
▪ Users .....	422
▪ Logical Files .....	423
▪ Recovery Files .....	424
▪ Administration Facilities .....	425
▪ Help on Direct Commands and Menu Functions .....	425

The SYSEDT utility provides administration functions for the editor buffer pool, which is a data container for the Software AG editor. The SYSEDT utility can be used for the following:

- Displaying parameters and runtime information of the editor buffer pool.
- Modifying parameters that control and initialize the editor buffer pool and its work file.
- Deleting logical work files and recovery files.

The *SYSEDT Utility - Editor Buffer Pool Administration* documentation is associated with the following documentation:

- *Operations: Operating the Software AG Editor*
- *Parameter Reference: EDBP - Software AG Editor Buffer Pool Definitions*

## Defining a Natural Security Library Profile

---

If you have Natural Security installed, you must create a library security profile for the SYSEDT utility.

For details, see *Library Maintenance* described in the *Natural Security* documentation.

## Invoking SYSEDT and Executing a Function

---

This section provides instructions for invoking the SYSEDT utility and executing a SYSEDT utility function.

### ▶ To invoke the SYSEDT utility

- Issue the following Natural system command:

```
SYSEDT
```

A SYSEDT **Main Menu** similar to the example below appears:

```
15:47:50          ***** NATURAL SYSEDT UTILITY *****          2009-01-29
User MM0          - Main Menu -          TID 1    33

          Code   Function
          G     General Information
          P     Generation Parameters
```

```

      U   Users
      F   Logical Files
      R   Recovery Files
      A   Administration Facilities
      .   Exit

Code ..

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Cont  Help  Menu  Exit           Admin Files Users Recov           GInfo ParmS Canc

```

The functions available in the menu are explained in the following sections.

#### ▶ To execute a SYSEDT function

- In the **Code** field of the SYSEDT **Main Menu**, enter the one-character code that corresponds to the function you want to execute and press ENTER. For example:

```
G
```

Executes the **General Information** function.

Or:

In the SYSEDT **Main Menu**, press the PF key that corresponds to the function you want to execute. For example:

```
PF10 (GInfo)
```

Executes the **General Information** function.

Or:

Issue the SYSEDT system command followed by any function code available in the SYSEDT **Main Menu**. For example:

```
SYSEDT G
```

Executes the **General Information** function.

For a list of all SYSERR direct commands available, use the help function described in [Help on Direct Commands and Menu Functions](#).

## General Information

This function invokes the **General Information** screen which provides an overview of the current status of the editor buffer pool. The fields contained on the screen are described in the following table. For detailed information on these fields, see *Editor Work File* and *Editor Buffer Pool* in the *Operations* documentation.

Field	Explanation
<b>Usage Statistics</b>	The currently available total number, the currently used number, and the currently used percentage of the available number of the items that follow.
<b>Buffer Pool Blocks</b>	The number of blocks in the editor buffer pool. See also <i>Obtaining Free Blocks</i> .*
<b>Work File Records</b>	The number of records in the editor work file. See also <i>Editor Work File</i> .*
<b>Control</b>	The number of control records, which is always one. See also <i>Control Record</i> .*
<b>Work</b>	The number of work records. See also <i>Work Record</i> .*
<b>Recovery</b>	The number of recovery records. See also <i>Recovery Records</i> .*
<b>Logical Files</b>	The number of logical files.
<b>Requests</b>	The total number of read and write requests, the number of read and write requests for buffer pool blocks ( <b>Pool</b> column), and the number of read and write requests for work or recovery files ( <b>File</b> column). The <b>Copy</b> column shows read requests, which (in contrast to locked read requests) result in the deletion of the corresponding buffer pool block.
<b>Read Work</b>	The number of read requests for logical file records. A logical file record can be found in the buffer pool ( <b>Pool</b> column) or on the work file ( <b>File</b> column). It can be read by a locked or by a copy request: locked means that the record is kept in the buffer pool for some time; copy means that it is deleted from the buffer pool after having been read.
<b>Write Work</b>	The number of write requests for logical file records. A record can be either written to the buffer pool ( <b>Pool</b> ) or moved to the work file ( <b>File</b> ) if there are no free blocks available.
<b>Read Recovery</b>	The number of read requests for recovery records in the editor work file.
<b>Write Recovery</b>	The number of write requests for recovery records in the editor work file.
<b>Timeout Values in Seconds</b>	Items with timeout values specified in seconds. These timeout values can be modified after pressing PF5 (Updat) and dynamically set after pressing PF5 (Save) again. The modified values are not kept during a restart of the buffer pool. The values from the work file control record are used instead. See also <i>Obtaining Free Blocks</i> .*
<b>Logical Files</b>	The time after which a logical file is deleted if it has not been accessed during this time.
<b>Files Delete Check</b>	The time after which all logical files are checked periodically whether they can be deleted.

Field	Explanation
<b>Changed Blocks</b>	The time after which blocks that have been modified can be freed by writing them to the work file.
<b>Unchanged Blocks</b>	The time after which blocks that have not been modified can be freed by writing them to the work file.
<b>Locked Blocks</b>	The time after which blocks that have been read with locked can be freed by writing them to the work file.

\* described in the *Operations* documentation

## Generation Parameters

This function invokes the **Generation Parameters** screen where you can view and change the current parameter settings of the editor buffer pool:

Parameter	Explanation
DDNAME	The name of the editor work file for the JCL definition.
DSNAME	The name of the work file dataset. This parameter is not displayed in a Com-plete, VM/CMS or z/VSE batch environment.
RECNUM	The total number of work file records.
LRECL	The work file record length.
PWORK	The percentage of work file records used as work records.
RWORK	The percentage of work records for regular logical files.
FMODE	The mode of the work file name, which can be from A1 to Z9. This parameter applies under VM/CMS only; it is not displayed in any other environment.
MAXLF	The maximum number of logical files in the editor buffer pool.
FTOUT	The timeout value (in seconds) for a logical file, which has not been accessed, to be deleted.
DTOUT	The period of time (in seconds) for logical files to be checked for deletion.
CTOUT	The timeout value (in seconds) for changed buffer pool blocks.
UTOUT	The timeout value (in seconds) for unchanged buffer pool blocks.
LTOUT	The timeout value (seconds) for locked buffer pool blocks.
ITOUT	The timeout value (in seconds) for the buffer pool initialization.

The parameters above are keyword subparameters of the **EDBP** profile parameter (or corresponding **NTEDBP** macro), which is used to define the initial editor buffer pool settings. For detailed information on **EDBP** and all keyword subparameters available, see *EDBP - Software AG Editor Buffer Pool Definitions* in the *Parameter Reference* documentation.

The **Start** column on the **Generation Parameters** screen refers to the buffer pool restart (see also *Initializing the Editor Buffer Pool* in the *Operations* documentation). The following start values can be displayed for the parameters:

Value	Explanation
L	The value for the corresponding parameter is taken from either the editor parameter module or the work file definition.
C	A modification of the corresponding parameter value forces a buffer pool cold start. Recovery records are lost.
W	A modification of the corresponding parameter value results in a buffer pool warm start. Recovery records are kept.

### ► To modify parameter values

- 1 On the **Generation Parameters** screen, press PF5 (Updat).

The write-protected fields next to the parameters change to input fields.

- 2 Replace the parameter value(s) required and press PF5 (Save) to save the new values in the control record of the editor work file. They will be activated when the editor buffer pool is started again.

If you change the value of PWORK, a window prompts you to confirm the cold start of the editor work file. Enter YES to confirm the new parameter value and to execute a buffer pool cold start for the work file initialization. (Any other entry or no entry resets the parameter to its previous value.)

## Users

This function invokes the **Users** screen which provides the following information:

Column	Explanation
User ID	The Natural user ID.
Logical Files	The number of logical files defined per user.
Pool Blocks	The number of buffer pool blocks per user.
Work Records	The number of work records per user.
Recovery Files	The number of recovery files per user.
Recovery Records	The number of recovery records per user.

For each user listed, you can execute one of the following line commands. You enter a line command in the **C** column, next to the user required.

Line Command	Explanation
? or *	Opens a window with a list of all valid line commands and corresponding PF keys, if available.
.	Exits the current screen and returns to the previous screen.
/ or P	Positions the line to the top of the screen.
F	Selects the logical files of this user.
R	Selects the recovery files of this user.
D	Deletes all logical files and/or recovery files for this user.

## Logical Files

This function invokes the **Logical Files** screen which provides the following information:

Column	Explanation
File No.	The logical file number.
User ID	The Natural user ID.
Type	The logical file type.
Pool Blks	The number of buffer pool blocks currently used per logical file.
File Recs	The number of work file records currently allocated per logical file.
Last Access	The date and time of the last read or write request per logical file.

For each logical file listed, you can execute one of the following line commands. You enter a line command in the **C** column, next to the user required.

Line Command	Explanation
? or *	Opens a window with a list of all valid line commands and corresponding PF keys, if available.
.	Exits the current screen and returns to the previous screen.
/ or P	Positions the line to the top of the screen.

Line Command	Explanation
or P	
S	Selects the logical files of this user.
D	Deletes the logical file.

## Recovery Files

---

This function invokes the **Recovery Files** screen which provides the following information:

Column	Explanation
<b>User ID</b>	The Natural user ID.
<b>Member</b>	The library member name.
<b>Library</b>	The library name.
<b>Type</b>	The library type.
<b>Recs</b>	The number of recovery records per recovery file.
<b>Creation Date</b>	The creation date of the recovery file.
<b>Creation Time</b>	The creation time of the recovery file.

For each recovery file listed, you can execute one of the following line commands. You enter a line command in the **C** column, next to the user required.

Line Command	Explanation
? or *	Opens a window with a list of all valid line commands and corresponding PF keys, if available.
.	Exits the current screen and returns to the previous screen.
/ or P	Positions the line to the top of the screen.
S	Selects the recovery files of this user.
D	Deletes the recovery file.

## Administration Facilities

---

This function invokes the **Administration Facilities** screen which can be used to terminate the editor buffer pool or the SYSEDT utility, or display the editor buffer pool in dump (hexadecimal) format for problem diagnosis.

### ▶ To use the Administration Facilities screen

- Terminate the editor buffer pool, by entering function code T.

A window prompts you for confirmation:

- If you enter YES, an additional window opens, asking you whether or not you want the editor buffer pool to be immediately restarted.

If you enter YES again, the buffer pool is immediately restarted, which gives you the possibility to immediately activate modified generation parameters.

If you enter NO, you leave SYSEDT and can perform actions outside your TP environment, for example, change the size of your editor work file.

Or:

Display the storage dump of the editor buffer pool, by entering function code D.

The offsets and addresses of the editor buffer pool are displayed on the screen.

If required, press PF10 (RelO) to position offset 00000000 to the top of the screen. PF1 (Help) invokes a list of all PF keys available.

Or:

Return to the SYSEDT **Main Menu** by entering a period (.).

## Help on Direct Commands and Menu Functions

---

You can obtain online help information on all SYSEDT direct commands available or on the function currently used.

### ▶ To view all SYSEDT direct commands

- In the command line of any SYSEDT utility screen, enter one of the following characters:

?

or

\*

All direct commands available within the SYSEDT utility are listed in alphabetical order.

▶ **To view help on a specific SYSEDT function**

- In the command line of a SYSEDT utility screen invoked by a SYSEDT utility function, enter the following:

H

Or:

Press PF1 (Help).

Help information on the fields and columns contained on the current SYSEDT utility screen is displayed.

# 73

## SYSERR Utility

---

When you develop a Natural application, you may want to separate error or information messages from your Natural code and manage them separately. This makes it easy for you, for example, to standardize messages, to have predefined message ranges for different types of message, to translate messages into other languages or to attach to a message a long text that explains it in more detail.

The SYSERR utility provides the option to write application-specific messages. In addition, you can use the SYSERR utility to customize the texts of the existing Natural system messages.

	<a href="#">General Information on Messages</a>
	<a href="#">Invoking SYSERR</a>
	<a href="#">Functions</a>
	<a href="#">Parameters</a>
	<a href="#">Direct Commands</a>
	<a href="#">Upper Case Conversion - ERRUPPER</a>
	<a href="#">Replacing Characters - ERRCHAR</a>
	<a href="#">Managing Messages in Different Libraries</a>
	<a href="#">Application Programming Interface USR0020P</a>



# 74

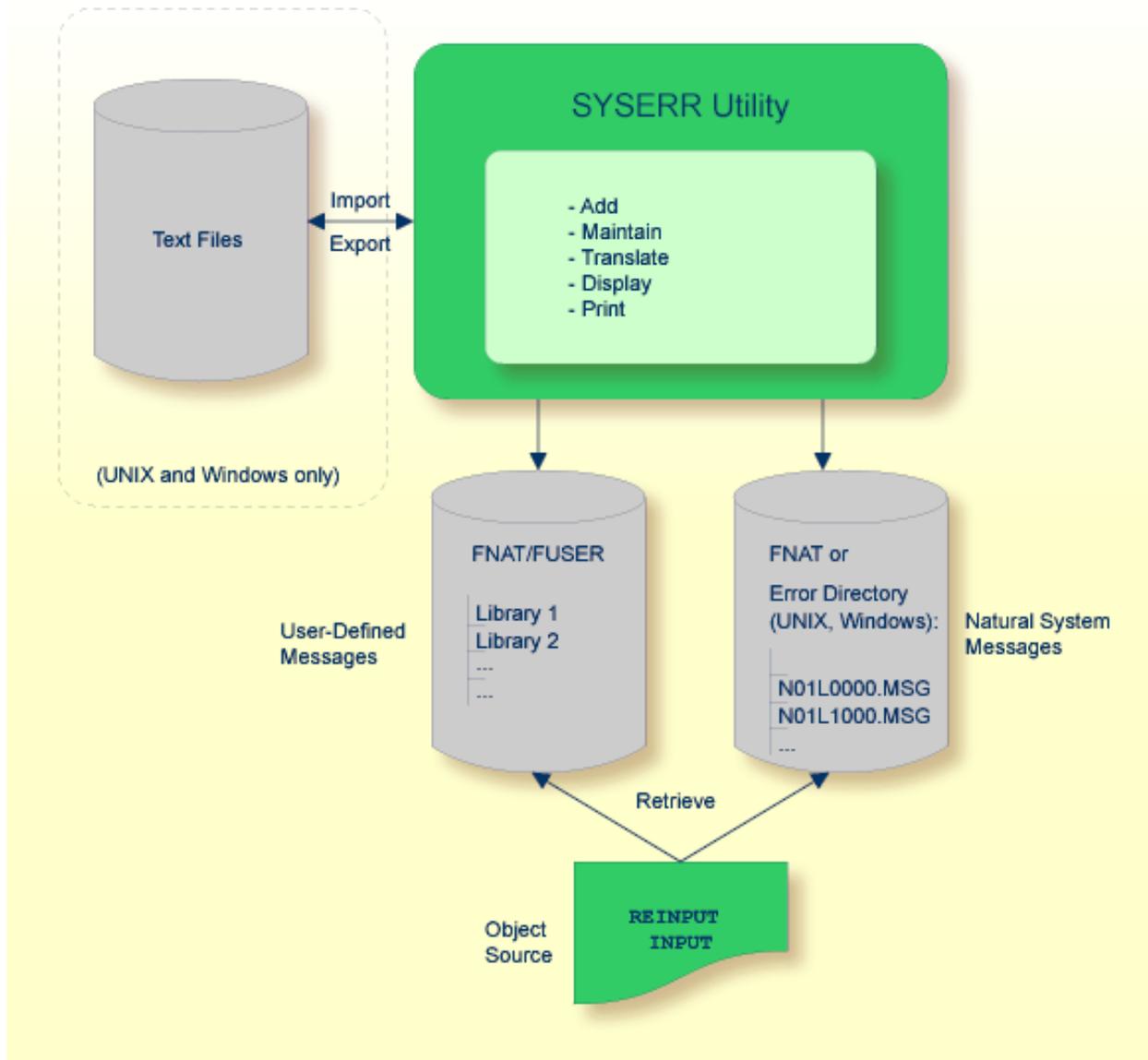
## General Information on Messages

---

- Message Types ..... 431
- Message Languages ..... 431
- Issuing Messages ..... 432
- Retrieving Natural System Short Messages ..... 433
- Retrieving User-Defined Short Messages ..... 433
- Obtaining Message Information ..... 434

This section contains information on the types of message and message languages that can be managed with the SYSERR utility and how messages are issued and retrieved in your Natural system environment.

The following graphic illustrates the features of the SYSERR utility and how messages are processed within Natural:



## Message Types

---

There are two types of message: Natural (system) messages and user-defined messages:

Natural system messages are issued by the Natural nucleus and Natural utilities. Natural system messages are delivered by Software AG and stored as message files in the Natural system file FNAT. Natural system messages begin with NAT, followed by a four-digit number, for example, NAT0230.

User-defined messages are issued by applications written by a user. User-defined messages are stored as message files in libraries (including SYS-libraries) in the system file FUSER or FNAT.

A message can be translated into different languages. Each language is stored in a separate message file. A maximum of 9999 messages can be stored per library and message file.

There are four types of message text:

- Natural system short message
- Natural system long message
- User-defined short message
- User-defined long message

A short message is the one-line message which is displayed in the message line when the corresponding error situation occurs.

A long message is a detailed explanation of the corresponding short message and includes instructions for solving problems.



**Caution:** Keep in mind that any modifications of Natural system messages can result in wrong messages or a loss of modifications when a new Natural version is released.

## Message Languages

---

Messages can be created in up to 60 languages as described for the system variable \*LANGUAGE in the *System Variables* documentation.

The following rules and restrictions apply:

- Natural system short messages must be entered in English first, and can then be translated into any other language.
- Natural system long messages can be entered in English, but cannot be translated into other languages.

- User-defined short messages can be entered in any language, and then be translated into any other language.
- User-defined long messages can be entered in any language, but only if the corresponding short message in the same language already exists.

## Issuing Messages

---

This section contains information on the Natural statements `INPUT` and `REINPUT` that are used to issue a Natural system short message or a user-defined short message in a Natural program.

### ▶ To issue a Natural system short message in a program

- Specify one of the following Natural statements:

```
INPUT WITH TEXT *-nnnn
```

or

```
REINPUT WITH TEXT *-nnnn
```

where *nnnn* is the number of the requested message (you can omit leading zeros).

### ▶ To issue a user-defined short message in a program

- Specify one of the following Natural statements:

```
INPUT WITH TEXT *nnnn
```

or

```
REINPUT WITH TEXT *nnnn
```

where *nnnn* is the number of the requested message (you can omit leading zeros).

## Dynamic Replacement of Message Text

A message text can contain variable parts that are identified by the notation `:n:`, where *n* represents occurrences 1 to 7. These variable parts are replaced by a value at runtime.

For details, see *operand3* in the section *INPUT Syntax 1 - Dynamic Screen Layout Specification* and *operand3* in the section *REINPUT* in the *Statements* documentation.

## Retrieving Natural System Short Messages

---

When a program references a Natural system short message, Natural looks for the requested message number in the Natural system file FNAT in the following order:

1. Under the current language code as determined by the system variable `*LANGUAGE`,
2. Under language code 1 (English).

If neither of the above is found, a program references a message that does not exist and you only receive the message number prefixed with `NAT`, for example, `NAT0230`.

## Retrieving User-Defined Short Messages

---

When a program references a user-defined short message, Natural first looks for the requested message number *nnnn* under the current language code as determined by the system variable `*LANGUAGE` (see the *System Variables* documentation). If that message does not exist, Natural looks for the requested message number *nnnn* under language code 1 (English). If that message does not exist either, Natural looks for message number *n000* (where *n* is the first digit of the requested message number) under language code 1.

These three search steps are first performed in the current library. If nothing is found there, further libraries are searched in the same way until a corresponding message is found.

The sequence of libraries for the search is as follows:

1. The current library as determined by the system variable `*LIBRARY-ID`,
2. The steplibs; if Natural Security is installed, the sequence in which the steplibs are specified in the Natural Security profile of the current library,
3. The default steplib as determined by the system variable `*STEPLIB`,
4. The library `SYSTEM` in the system file `FUSER (*)`,
5. The library `SYSTEM` in the system file `FNAT (*)`.

(\*) If the name of the current library begins with SYS, SYSTEM FNAT is searched before SYSTEM FUSER.

## Obtaining Message Information

---

When you receive a short message, you may be looking for additional information on the problem situation.

- With the system command `HELP`, you can display Natural system long messages or user-defined long messages.
- With the system command `LASTMSG`, you can list the short text of the message(s) that occurred last and additional information on the error situation. The information displayed includes associated error messages that possibly preceded the last message.

Both commands are described in the *System Commands* documentation.

# 75 Invoking SYSERR

▶ To invoke the SYSERR utility

- Enter the following system command:

```
SYSERR
```

The main menu of the SYSERR utility is displayed:

```
16:15:35          ***** NATURAL SYSERR UTILITY *****          2008-09-18
                          - Menu -

      Code  Function
      ----  -
      AD    Add new messages
      DE    Delete messages
      DI    Display messages
      MO    Modify messages
      PR    Print messages
      SC    Scan in messages
      SE    Select messages from a list
      TR    Translate messages into another language
      ?    Help
      .    Exit
      ----  -

Code .. __  Message type .... US
           Library ..... SYSTEM__
           Message number .. 1__ - 9999
           Language codes .. 1_____

Command ==>
```

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

From the main menu of the SYSERR utility, you can execute all SYSERR utility functions available for creating and maintaining messages. The individual functions are explained in the section *Functions*. The parameters that apply with the functions are explained in general in the section *Parameters*; any restrictions that apply to the use of these parameters are described for each function concerned in the section *Functions*.

The SYSERR utility provides an extensive online help system. To obtain field-specific help information, either enter a question mark in the relevant field and press ENTER or place the cursor in the field and press PF1.

# 76 Functions

---

- Adding Messages ..... 438
- Deleting Messages ..... 442
- Displaying Messages ..... 442
- Modifying Messages ..... 444
- Printing Messages ..... 446
- Scanning Messages ..... 447
- Selecting Messages from a List ..... 450
- Translating Messages into other Languages ..... 452

You invoke a SYSERR utility function by entering the code that corresponds to the required function and one or more parameters in the input fields of the SYSERR main menu. This section describes the functions provided in the menu and the parameters that can be specified for each function. For general instructions on the use of parameters, see the section *Parameters*.

## Adding Messages

▶ **To add new messages**

- 1 If you want to prevent lower- to upper-case translation for the messages to be created, enter the following terminal command:

```
%L
```

Any lower-case characters you type when adding message text, will then *not* be converted to upper-case characters for the duration of the current Natural session.

For detailed information on %L, see the *Terminal Commands* documentation.

- 2 Invoke the SYSERR main menu and enter the following values:

Field	Input Value
<b>Code</b>	AD
<b>Message type</b>	NS Natural system short messages NL Natural system long messages US User-defined short messages UL User-defined long messages  A long message can only be added if the corresponding short message already exists, as the long message is intended to be an explanation of the short message.
<b>Library</b>	Any existing Natural library.
<b>Message number</b>	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be added. If you only want to add one message, either enter the number of the new message in the left <b>Message number</b> field and clear the right field, or enter the number in both fields.
<b>Language codes</b>	The code of the language for which the message is to be added. If the message type is NS or NL, the language code must be 1 for English. For other message types, the first language code entered in the field is used; all others are ignored.

- 3 Press ENTER.

An **Add Short Message** screen similar to the example below is displayed:

```

15:56:01          ***** NATURAL SYSERR UTILITY *****          2008-11-28
Lower Case ON          - Add Short Message -

Number          Short Message (English)
-----
SYSERR1004
          .....1.....2.....3.....4.....5.....+..

Sample ..... Message sample number 0000

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

```

The **Number** field indicates the message number (in the example above, 1004), which is prefixed with the library ID (in the example above, SYSERR).

- 4 In the input line next to the message number, type in a short message text and press ENTER.

Or:

If the line labeled **Sample** contains a sample message text as shown in the example above, copy this text into the input line by entering `.C` and then pressing `ENTER`. If the sample message text contains the string `0000`, this string is replaced by the new message number as illustrated in the following example:

```

15:30:46          ***** NATURAL SYSERR UTILITY *****          2008-11-28
Lower Case ON          - Add Short Message -

Number          Short Message (English)
-----
SYSERR1004      Message sample number 1004
                .....1.....2.....3.....4.....5.....+..

Sample ..... Message sample number 0000

Message has been added.

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

For instructions on creating a sample message, see the `SAMPLE` command described in *Direct Commands*.

- 5 Press `PF9` to add a corresponding long message text.

An **Add Long** *message-number* screen similar to the example below appears:

```

16:20:24          - Add Long SYSERR1004 (English) - Lower Case ON          2008-11-28
 1 Tx. Message sample number 1004
 2      .
 3      .
 4 Ex. .
 5      .
 6      .
 7      .
 8      .
 9      .
10     .
11     .
12     .
13     .
14     .
15     .
16     .
17     .
18 Ac. .
19     .
20     .

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Add          Exit          Short Copy          Canc

```

- 6 Enter text in the three input areas: **Tx.** (text), **Ex.** (explanation) and **Ac.** (action).
- 7 Press ENTER to save the long message.
- 8 Press PF9 to return to the short message or to add the next short message in ascending order if you selected a range of message numbers.
- 9 Press PF3 or PF12 to return to the SYSERR main menu.

Or:

Press PF8 or PF7 to add the next short message in ascending or descending order if you selected a range of message numbers.

## Deleting Messages

▶ **To delete messages**

- In the fields of the SYSERR main menu, enter the following values:

Field	Input Value
<b>Code</b>	DE
<b>Message type</b>	NS Natural system short messages NL Natural system long messages US User-defined short messages UL User-defined long messages  It is possible to delete a long message without deleting the corresponding short message, but not vice versa. If you try to delete a short message for which a long message exists, you are asked to confirm the deletion of both.
<b>Library</b>	Any existing Natural library.
<b>Message number</b>	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be deleted.
<b>Language codes</b>	The code(s) of the language(s) in which the messages are to be deleted. To indicate that the messages specified are to be deleted in all languages available, enter an asterisk (*).

## Displaying Messages

▶ **To display messages**

- 1 In the fields of the SYSERR main menu, enter the following values:

Field	Input Value
<b>Code</b>	DI
<b>Message type</b>	NS Natural system short messages NL Natural system long messages US User-defined short messages UL User-defined long messages

Field	Input Value
<b>Library</b>	Any existing Natural library.
<b>Message number</b>	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be displayed.
<b>Language codes</b>	The code of the language in which the messages are to be displayed. Only one language code is accepted. If more than one code is specified, only the first one is used; all others are ignored.

2 Press ENTER.

For short messages, a **Display Short Messages** screen similar to the example below appears:

```

15:41:11          ***** NATURAL SYSERR UTILITY *****          2008-11-28
                   - Display Short Messages -

Number           Short Message (English)
-----
NAT0001         Missing/invalid syntax; undefined variable name/keyword.
NAT0002         No file is available with specified name or number.
NAT0003         Invalid character string for file name or file number.
NAT0004         DEFINE DATA must be the first statement if present.
NAT0005         Closing parenthesis missing in arithm/logical expression.
NAT0006         ESCAPE statement used when no processing loop active.
NAT0007         Invalid THRU or TO clause in READ LOGICAL or HISTOGRAM.

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
+                Exit                +                Cancell

```

Press PF8 to page forwards.

For long messages, the **Display Long Message** screen is displayed where the messages are displayed one after another by pressing PF8 to page forwards or PF7 to page backwards. The **Display Long Message** screen is similar to the **Modify Long Message** screen shown in *Modifying Messages*.

## Modifying Messages

▶ **To modify messages**

- 1 If you want to prevent lower- to upper-case translation for the messages to be modified, enter the following terminal command:

```
%L
```

Any lower-case characters you type when editing message text, will then *not* be converted to upper-case characters for the duration of the current Natural session.

For detailed information on %L, see the *Terminal Commands* documentation.

- 2 In the fields of the SYSERR main menu, enter the following values:

Field	Input Value
<b>Code</b>	MO
<b>Message type</b>	NS Natural system short messages NL Natural system long messages US User-defined short messages UL User-defined long messages
<b>Library</b>	Any existing Natural library.
<b>Message number</b>	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be modified.
<b>Language codes</b>	The code of the language in which the messages are to be modified. Only one language code is accepted. If more than one code is specified, only the first one is used; all others are ignored.

- 3 Press ENTER.

A **Modify Short Message** screen similar to the example below is displayed:

```
16:46:31          ***** NATURAL SYSERR UTILITY *****          2008-11-28
Lower Case ON          - Modify Short Message -

Number          Short Message (English)
-----
SYSERR1004      Message sample number 1004
                .....1.....2.....3.....4.....5.....+..
```

```

1 Tx. Input missing.
2   .
3   .
4 Ex. Input value missing in field XYZ.
5   Enter an alphanumeric value.
6   .
7   .
8   .
18 Ac. Enter value in field XYZ.
19  .
20  .

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Mod          Exit          -      +          Copy          Canc

```

For reference purposes, the long message is displayed in the bottom half of the screen.

When you modify long messages, the **Modify Long** *message-number* screen is displayed:

```

16:56:08      - Modify Long SYSERR1004 (English) - Lower Case ON      2008-11-28
1 Tx. Input missing.
2   .
3   .
4 Ex. Input value missing in field XYZ.
5   Enter an alphanumeric value.
6   .
7   .
8   .
9   .
10  .
11  .
12  .
13  .
14  .
15  .
16  .
17  .
18 Ac. Enter value in field XYZ.
19  .
20  .

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Mod          Exit          -      +          Copy          Canc

```

- 4 Press ENTER to save any modifications.
- 5 Press PF8 or PF7 to modify the next message in ascending or descending order if you selected a range of numbers.



See the `DEFINE PRINTER` statement in the *Natural Statements* documentation for details on logical printer names.

- 4 Press `ENTER` to output the selected messages on a printer.

▶ **To print all Natural system messages**

- In the fields of the `SYSERR` main menu, enter the following values:
  - Code `PR`,
  - Message type `NS` or `NL`,
  - Message number range 1 - 9999,
  - Language code 1 (English) or 2 (German).

A library ID is not required and possible entries are ignored.

## Scanning Messages

This function is used to scan messages for a specific string of characters. Only short messages can be scanned.

▶ **To scan messages**

- 1 In the fields of the `SYSERR` main menu, enter the following values:

Field	Input Value
<b>Code</b>	<code>SC</code>
<b>Message type</b>	<code>NS</code> Natural system short messages <code>US</code> User-defined short messages
<b>Library</b>	Any existing Natural library.
<b>Message number</b>	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be scanned.
<b>Language codes</b>	Specify a maximum of nine language codes from the ranges 1 - 9, A - Z and a - y, or enter an asterisk (*) for all languages.

- 2 Press `ENTER`.

A scan window similar to the example below opens:

```

+-----+-----+
| Scan value(s)      Or/And/Not |
|-----|-----|
| _____         OR_         |
| _____         |
| _____         |
| _____         |
|
| Absolute ..... X  |
| Immediate ..... _ |
+-----+-----+
  
```

In the fields provided, you can specify the search criteria to be used for scanning:

<b>Scan value(s)</b>	In the four empty fields, enter up to four character strings to be searched for. The scan finds the specified terms in both upper and lower case.	
<b>Or/And/Not</b>	You can perform a Boolean search query by entering one of the following operators:	
	OR	Searches for one or more of the character strings entered in <b>Scan value(s)</b> . This is the default setting.
	AND	Searches for all of the character strings entered in <b>Scan value(s)</b> .
	NOT	Searches for none of the character strings entered in <b>Scan value(s)</b> .
	The operator is ignored if you only fill one of the <b>Scan value(s)</b> fields.	
<b>Absolute</b>	If you mark this field, the string of characters is found even if it is part of a word. For example, if you scan for the value <code>meter</code> , the search would also find words such as <code>parameter</code> and <code>millimeter</code> .  If you remove the mark, the search is restricted to match entire words only.	
<b>Immediate</b>	If you mark this field, messages are displayed individually, one after another. Otherwise, a list of messages is displayed after the search is completed.  If you specify more than one language or an asterisk (*) in the <b>Language codes</b> field, <b>Immediate</b> must be marked.	

3 Specify search criteria as shown in the following example:

```
+-----+
! Scan value(s)      Or/And/Not  !
! -----          - - - - -    !
! BUFFER_____     AND          !
! POOL_____        !
! _____        !
! _____        !
! Absolute ..... X  !
! Immediate ..... _ !
!                   !
+-----+
```

In the example above, the scan finds all short messages that contain both the words `buffer` and `pool`.

4 Press ENTER.

All messages to which the specified search criteria apply are listed on the screen as shown in the following example:

```
11:32:27          ***** NATURAL SYSERR UTILITY *****          2008-11-28
                   - Scan in Short Messages -

Number           Short Message (English)
-----
NAT0777          Buffer pool full.

End of scan reached.

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

The word in which the search string is found is highlighted.

From this screen, you can display the search criteria used for the current scan by pressing PF10.

## Selecting Messages from a List

This function is used to display a range of messages and select single ones for further processing. Only short messages can be displayed.

### ► To select messages

- 1 In the fields of the SYSERR main menu, enter the following values:

Field	Input Value
<b>Code</b>	SE
<b>Message type</b>	NS Natural system short messages US User-defined short messages
<b>Library</b>	Any existing Natural library.  If an asterisk (*) is appended to the library ID, a list of all libraries available is displayed for selection.
<b>Message number</b>	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be displayed for selection.
<b>Language codes</b>	The code of the language in which the messages are to be displayed.  If more than one code is specified, only the short message text of the first one is displayed. Enter an asterisk (*) to display the languages available for each message.

- 2 Press ENTER.

A **Select Messages** screen similar to the example below is displayed:

```

17:08:13          ***** NATURAL SYSERR UTILITY *****          2008-11-28
                  - Select Messages -
Se Number      Short Message (English)          Languages
                                     short      long
-----
__ NAT0001     Missing/invalid syntax; undefined variable name/ 1          1
__ NAT0002     No file is available with specified name or numb 1          1
__ NAT0003     Invalid character string for file name or file n 1          1
__ NAT0004     DEFINE DATA must be the first statement if prese 1          1
__ NAT0005     Closing parenthesis missing in arithm/logical ex 1          1
__ NAT0006     ESCAPE statement used when no processing loop ac 1          1
    
```



Command	Explanation
.Y	Defines a shorter message range by listing messages only up to a selected message:  All messages that were listed below the message selected with this command are removed from the display. The message range in the SYSERR main menu is reset accordingly and ends with the message selected here on the <b>Select Messages</b> screen.

4 Press ENTER to continue.

## Translating Messages into other Languages

This function is used to translate short messages from one language to one or more other languages. To translate long messages into other languages, proceed as described in [Adding Messages](#).

### ▶ To translate short messages

1 In the fields of the SYSERR main menu, enter the following values:

Field	Input Value
Code	TR
Message type	NS Natural system short messages US User-defined short messages
Library	Any existing Natural library.
Message number	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be displayed for selection.
Language codes	Specify a maximum of nine language codes. The language codes are single alphanumeric characters in the ranges 1 - 9, A - Z and a - y.

2 Press ENTER.

A **Translate Short Message** screen similar to the example below appears:

```

13:37:14          ***** NATURAL SYSERR UTILITY *****          2009-01-16
Lower Case ON          - Translate Short Message -

Number ..... SYSERR0001
Languages ... 1..45.....

-----+...1...+...2...+...3...+...4...+...5...+..
English      Short message English (1)_____
German      _____
    
```

```

French
Spanish Short message Spanish (4)
Italian Short message Italian (5)
-----
.....1.....2.....3.....4.....5.....+..
1 Short message English (1)
4 Explanation: English long message
18 Action: English long message

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

```

The **Languages** field displays the language codes in which the message already exists (in the example above: 1, 4 and 5).

The section below **Number** and **Languages** lists all languages for which a language code was entered earlier in the **Language codes** field of the SYSERR main menu (in the example above: 1, 2, 3, 4, 5). English (1), Spanish (4) and Italian (5) translations already exist whereas new translations can be entered for German (2) and French (3).

For reference purposes, the bottom section of the screen displays three lines of the long message that corresponds to the language that is listed first in the languages/short messages section (in the example above, English). Lines 1, 4 and 18 are displayed by default. You can display any other line of the long message by overwriting any of the three line numbers (**1**, **4** or **18**) with another line number and pressing ENTER.

- 3 Enter the translation in the input line next to the new language specified.
- 4 Press ENTER.

▶ **To modify translations of short messages**

- 1 On the **Translate Short Message** screen, press PF10.

An **Options** window similar to the example below opens:

```
+----- Options -----+
!
! Modification of all fields allowed ..... N      !
!
! Currently recognized language codes ..... 123456789 !
!
+-----+

```

- 2 In the upper field, replace N (default) by Y and enter the required language code(s) in the lower field. You can specify up to nine new language codes for translation.

▶ **To copy a translation into an empty input line**

- 1 On the **Translate Short Message** screen, enter .C in the first two positions of an empty line.
- 2 Place the cursor anywhere in the line of a short message that already exists for another language. (You can only copy text that appears in display mode.)
- 3 Press ENTER.

# 77 Parameters

---

- Message Type ..... 456
- Library ..... 456
- Message Number ..... 456
- Language Codes ..... 456

This section describes the parameters that can be specified for a function in the SYSERR main menu. Any restrictions that apply to the use of parameters with a particular function are described in the section *Functions*.

## Message Type

---

Specifies the type of message to be processed. The table below lists the message types available:

Type	Explanation
NS	Natural system short messages
NL	Natural system long messages
US	User-defined short messages
UL	User-defined long messages

## Library

---

Specifies the library for which messages are to be created or maintained. The specification of a library is not required when accessing Natural system messages (Message types NS and NL); any input values in the **Library** field are ignored.

## Message Number

---

Specifies the first and last number of a message range. The maximum message number for a library and language is 9999. The message number 0000 is not allowed. To specify only one message number, either enter the number of the message in the left **Message number** field and clear the right field, or enter the number in both fields.

## Language Codes

---

Specifies a maximum of 9 from 60 language codes available. The language codes are single alphanumeric characters in the ranges 1 - 9, A - Z and a - y. To view or select language codes, enter a question mark (?) in the first position of the **Language codes** field and press ENTER. For more information, see the system variable \*LANGUAGE in the *System Variables* documentation.

# 78 Direct Commands

From the SYSERR main menu, you can execute the following commands by entering them in the command line:

Command	Explanation
LAYOUT	Specifies valid message ranges to categorize messages. Overlapping of ranges is possible. A new message can only be added if its number is within the range specified in the layout.
NEXT	Searches for the next free message number within the message number range specified. Free means that this message number is available and has not yet been assigned to a message file in any language.
NEXTTAB	Same as NEXT, but returns a list of message numbers from which you can select a number.
RESTART	Re-initializes SYSERR (and its default values) without leaving the utility.
SAMPLE	<p>Invokes the <b>Edit SAMPLE message</b> window where you create or modify a sample message to be used as a master for creating new short messages.</p> <p>To create or modify a sample message, proceed as follows:</p> <ul style="list-style-type: none"><li>■ In the editor area of the <b>Edit SAMPLE message</b> window, type in the message text required or modify the existing text. If you enter the string 0000 (combined with text or not), the string 0000 is replaced by the number of the new message when copying the message. See also <a href="#">Step 4</a> of <i>Adding Messages</i> in the section <i>Functions</i>.</li><li>■ In the <b>Read or Write sample</b> field, enter a W to save your entries.</li><li>■ In the <b>Library</b> field, enter the name of the library for which the sample message is to be used. If you leave the <b>Library</b> field blank, the sample applies to Natural system messages.</li><li>■ Press PF3 to exit the <b>Edit SAMPLE message</b> window.</li></ul> <p>You can define one sample message for each language and library.</p>
SECURITY	Invokes a security window where you can enter a password and cipher code for accessing security-protected Adabas and VSAM files.
SHIFT	If activated, automatically shifts the text of a short message to the left margin when confirming a modification or adding a new message.

Command	Explanation
TRACE	Counts the number of database accesses. When the message number specified has been reached, a window is displayed. The default number is 900. If set to 0, the trace facility is shut off. The commands TRACE ON and TRACE OFF can be entered directly in the command line. TRACE ON sets the access counter to 900; TRACE OFF sets the access counter to 0.
USEREXIT	Invokes the USEREXIT program in the Natural system library SYSERR.

# 79

## Upper Case Conversion - ERRUPPER

---

Natural system messages are provided in lower case. If your terminals cannot display lower-case characters correctly, convert the messages from lower to upper case by executing the program ERRUPPER in the Natural system library SYSERR.

However, once the messages have been converted to upper case, you cannot convert them back to lower case. To recover lower-case messages, you have two options:

- Reload the messages by using the ERRLODUS program or the Object Handler.
- Unload the lower-case messages to a free language code by using the ERRULDUS program or the Object Handler before conversion so that a backup always exists.

For detailed information, see *Unloading Messages - ERRULDUS* and *Loading Messages - ERRLODUS*, and the *Object Handler* documentation.



# 80

## Replacing Characters - ERRCHAR

---

If your terminal does not display certain characters correctly, it is possible to search for these characters and replace them by new characters of your choice. This is done by executing the program ERRCHAR in the Natural system library SYSERR. However, it is only possible to replace characters in Natural system short messages. When using ERRCHAR, you scan for a specific character and replace the hexadecimal code that represents this character with another hexadecimal code.

After executing the program ERRCHAR, the **ERRCHAR** menu is displayed with the following functions:

- Scan for a given character
- Scan and Replace characters
- Display one message in hexadecimal format
- EBCDIC character table for your terminal
- Translate using character set ERRCSET

The following input fields are provided in the **ERRCHAR** menu:

Field	Explanation
<b>Message Number</b>	The range of messages to be included in the search or search/replace operation.
<b>Language Code</b>	The language code of Natural system short messages to be included in the search or search/replace operation.
<b>Scan Value</b>	The hexadecimal value to be scanned for.
<b>Replace Value</b>	The hexadecimal value to replace all scan values found. Use the function <b>EBCDIC character table for your terminal</b> to determine which characters your terminal can represent.



# 81

## Managing Messages in Different Libraries

---

- Unloading Messages - ERRULDUS ..... 464
- Loading Messages - ERRLODUS ..... 466

You can transfer messages between different libraries and rename, find, list or delete messages in different libraries. Depending on the Natural system environment(s) affected, you can either unload and load the messages to work files, or directly copy/move them from one library to another.

▶ **To copy/move, rename, find, list or delete messages**

- For libraries contained in the current Natural environment located on a mainframe platform, use the **SYSMAIN** utility as described in the relevant documentation.

Or:

For libraries contained in different Natural environments located on different mainframe, UNIX, OpenVMS or Windows platforms, use the **Object Handler** as described in the relevant documentation.

▶ **To unload or load messages from or to a work file**

- For libraries contained in a local mainframe environment, use the ERRULDUS and ERRLODUS programs described in the following sections.

Or:

For libraries contained in different Natural environments located on different mainframe, UNIX, OpenVMS or Windows platforms, use the **Object Handler** as described in the relevant documentation.

## Unloading Messages - ERRULDUS

---

The ERRULDUS program is used to unload all message types supported by Natural; that is, user-defined long and short messages created with the SYSERR utility and Natural system messages. The messages are read from the FNAT or FUSER system file and written into work file 2.

▶ **To invoke ERRULDUS**

- In the command line of the SYSERR main menu, enter ERRULDUS.

An **ERRULDUS** menu similar to the example below is displayed:

```

16:11:13          ***** NATURAL SYSERR UTILITY *****          08-09-18
- ERRULDUS (Unload Texts to Work File 2) -

Code  Function
-----
US   User supplied short error texts
UL   User supplied long  error texts
U    User supplied short AND long texts
NS   NATURAL short error texts
NL   NATURAL long  error texts
H    NATURAL help texts
.    Exit
-----

Code .....  ___
Source Library .. SYSERR__      Source Language Code .. 01
Target Library .. SYSERR__      Target Language Code .. 01
Error Number .... 1___ - 9999
Replace .....  N
Report .....  ON_

Please enter valid code.
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Exec  Help          Exit                                     Canc
    
```

The **ERRULDUS** menu contains the following input fields:

Field	Explanation
<b>Code</b>	The type of messages to be unloaded. Valid types are:  US User-defined short messages UL User-defined long messages U User-defined short and long messages NS Natural system short messages NL Natural system long messages H Natural help texts . Terminate processing
<b>Source Library</b>	The name of the library from which messages are to be unloaded. The name can be truncated by using an asterisk (*). If you enter an asterisk (*) only, messages from all libraries are unloaded. <b>Source Library</b> is ignored for messages of type NS or NL.
<b>Source Language Code</b>	The language code of the messages to be unloaded.

Field	Explanation
<b>Target Library</b>	The name of the library to which messages are to be loaded. <b>Target Library</b> is ignored for messages of type NS or NL.
<b>Target Language Code</b>	The language code to which messages are to be loaded.
<b>Error Number</b>	Two numbers of up to four digits corresponding to the first and last numbers of the range of messages to be unloaded. If no first number is specified, it is assumed to be 1.
<b>Replace</b>	Enter Y to overwrite the target library. Default is N (No).
<b>Report</b>	If specified as ON, a list of all messages that were unloaded is displayed when unloading is complete. If specified as OFF, no such list is displayed.

ERRULDUS reads input until a period (.) is detected in the **Code** field. After unloading, a statistical listing is displayed.

## Loading Messages - ERRLODUS

---

The ERRLODUS program is used to load messages previously unloaded with the ERRULDUS program.

The messages are read from work file 2 and are written to the FNAT or FUSER system file. ERRLODUS overwrites existing messages in the system file if **Replace** was specified as Y when unloading.

If you specify ERRLODUS as ON, a list of all messages that were loaded (added or replaced) is displayed when loading is complete.



**Note:** Under Natural Security, the online use of ERRLODUS is only available for users of type Administrator, regardless of the setting of the **Utilities** option in the security profile of the library. See also *Utilities in Library Maintenance* in the *Natural Security* documentation.

# 82

## Application Programming Interface USR0020P

---

The application programming interface USR0020P in the Natural system library SYSEXT is provided to read messages from the FNAT or FUSER system file. Thus, it is possible, for example, to have long messages displayed in an application (as part of your own user-defined help system) without having to use the Natural system library SYSERR.

Log on to the Natural system library SYSEXT and, in the command line, enter the command `MENU`. In the list provided, mark the program USR0020P with a question mark (?). A window is then displayed, in which you can select the function to be executed for the program. If you enter an I, further information on the use of USR0020P is displayed.



# 83 SYSEXT Utility - Natural Application Programming

## Interfaces

---

▪ Basic Features of SYSEXT .....	470
▪ Invoking and Terminating SYSEXT .....	472
▪ SYSEXT Utility Screen .....	473
▪ Performing SYSEXT Utility Functions .....	474
▪ Using a Natural API .....	478
▪ List of Natural APIs .....	478

The utility SYSEXT is used to locate and test Natural Application Programming Interfaces (APIs) contained in the current system library SYSEXT.

A Natural API is a Natural subprogram (cataloged object) that is used for accessing and possibly modifying data or performing services that are specific to Natural, a subcomponent or a subproduct. The purpose of a Natural API is to retrieve or modify information or use services that are not accessible by Natural statements.

**Related Topics:**

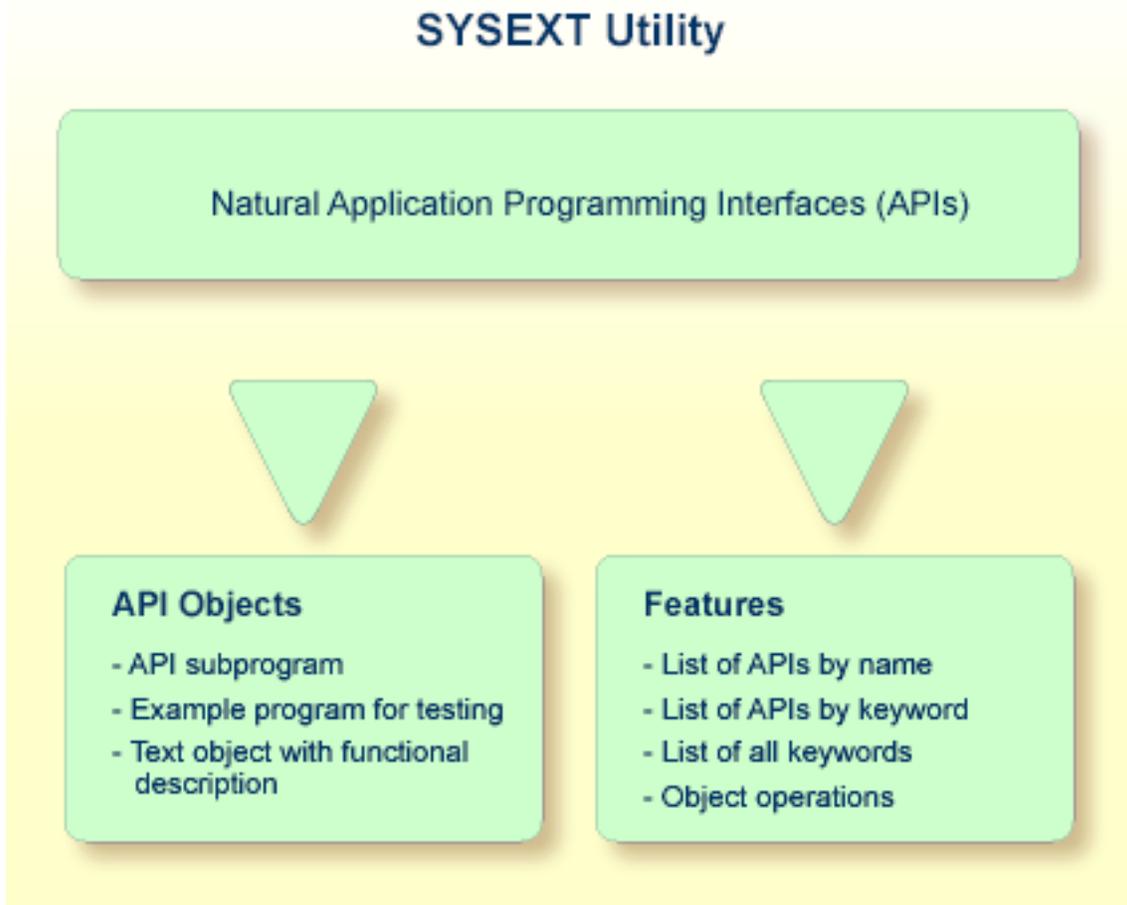
- *Natural User Exits - Operations* documentation
- *Application Programming Interfaces - Natural Security* documentation
- *Application Programming Interfaces - Natural SAF Security* documentation
- *SYSAPI - APIs of Natural Add-on Products - Utilities* documentation

## Basic Features of SYSEXT

---

For each Natural API, the utility SYSEXT provides a functional description, one example program and API-specific keywords.

The following diagram is an overview of the Natural objects and major features SYSEXT provides to test and implement an API:



### Objects Provided for Natural APIs

The types of Natural object typically provided for each Natural API are listed in the following section. The section does not list additional objects that may be required for particular APIs.

All API-related objects are contained in the library SYSEXT on the system file FNAT.

In the following table, *nnnn* denotes the 4-digit number assigned to the API. All API objects that relate to one another have identical numbers.

Object Name	Explanation
USR <i>nnnn</i> N	The API subprogram (cataloged object) that performs the designated function.
USR <i>nnnn</i> P	An example program (source object) that can be used to test the effect of the API. The example program invokes the corresponding subprogram USR <i>nnnn</i> N.
USR <i>nnnn</i> T	A text object that contains a description of the corresponding API. The description comprises purpose, function and calling conventions of the API and relevant keywords.

Object Name	Explanation
	You can display a text object by using the line command D as described in <a href="#">Line Commands</a> in <i>Performing SYSEXT Utility Functions</i> .
	You can search for a text object by specifying a keyword as described in <a href="#">Keyword Search</a> in <i>Performing SYSEXT Utility Functions</i> .

## Invoking and Terminating SYSEXT

This section provides instructions for invoking and terminating the SYSEXT utility.

### ▶ To invoke SYSEXT

- Enter the following system command:

```
SYSEXT
```

A SYSEXT utility screen similar to the example below appears with a list of all available Natural APIs:

```

13:11:42          ***** NATURAL SYSEXT UTILITY *****          2005-08-01
User SAG                      - Menu -                      Library SYSEXT

Cmd  Source   Interface  Comment
Prod
_   USR0010P  USR0010N  Get 'SYSPROF' Information
NAT
_   USR0011P  USR0011N  Information about logical file
NAT
_   USR0020P  USR0020N  Read any error text from FNAT / FUSER
NAT
_   USR0040P  USR0040N  Get type of last error
NAT
_   USR0050P  USR0050N  Get 'SYSPROD' Information
NAT
_   USR0060P  USR0060N  Copy LFILE definition from 'FNAT' to 'FUSER'
NAT
_   USR0070P  USR0070N  Default Editor Profile 'SYSTEM'
NAT
_   USR0080P  USR0080N  Handle Type/Name of Editor Contents
NAT
_   USR0100P  USR0100N  Control LRECL

```

```

NVS
_   USR0120P  USR0120N  Read Natural Short Error Message
NAT
_   USR0210P  USR0210N  Save, cat or stow NATURAL object
NAT
_   USR0220P  USR0220N  Read Natural Long Error Message
NAT
_   USR0320P  USR0320N  Read User Short Error Message from FNAT or FUSER
NAT
_   USR0330P  USR0330N  Read Natural Object Directory
NAT

Keyword .. _____

Command ==>

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

```

The list is sorted by the names of the example programs (USR $nnnn$ P). If required, press PF8 to scroll down one page in the list or PF7 to go to the beginning of the list.

#### ► To terminate SYSEXT

- On the SYSEXT utility screen, press PF3 or PF12.

Or:

In the **Command** line, enter a period (.) or enter EXIT.

## SYSEXT Utility Screen

The field and columns contained on the SYSEXT utility screen are explained in the following table:

Column	Explanation
<b>Cmd</b>	The input field for a line command to be executed on a text object or an example program: see <a href="#">Line Commands</a> in <i>Performing SYSEXT Utility Functions</i> .
<b>Source</b>	The name of the example program (source object USR $nnnn$ P) that can be used to invoke the API.
<b>Interface</b>	The name of the API subprogram USR $nnnn$ N.
<b>Comment</b>	A brief description of the purpose of the API.
<b>Prod</b>	The product code of Natural (NAT) or a Natural add-on product affected by the API.

Column	Explanation
	For example: NAT = Natural, NDB = Natural for DB2, NVS = Natural for VSAM, PRD = Predict, RPC = Natural Remote Procedure Call.
<b>Keyword</b>	The input field for a keyword relevant to an API: see <a href="#">Keyword Search</a> in <i>Performing SYSEXT Utility Functions</i> .

## Performing SYSEXT Utility Functions

The SYSEXT utility functions can be used to perform operations on the text objects and example programs referenced in the API list on the SYSEXT utility screen. In addition, you can use the keyword search function to shorten the list of APIs or list all keywords available for the APIs.

The section below covers the following topics:

- [Line Commands](#)
- [Keyword Search](#)

### Line Commands

Line commands are used to perform object operations. You enter a line command in the **Cmd** column next to the API required. For a list of valid line commands, enter a question mark (?) in this column.

The following line commands are available on the SYSEXT utility screen:

Line Command	Function
E	Edit example program USRnnnnP.
L	List example program USRnnnnP.
R	Run example program USRnnnnP.
X	Execute example program USRnnnnP.
D	List text object USRnnnnT for a description of the corresponding API.  The description comprises purpose, function and calling conventions of the API and keywords that are relevant to the API.
K	List keywords relevant to the specified API.
.	Terminate the SYSEXT utility.

## Keyword Search

Keywords help you find the Natural APIs relevant to your current task. You can use the keyword search function to list APIs by keyword or list the keywords relevant to a specific API.

### ▶ To list APIs by keyword

#### ■ Choose any of the following methods:

##### ■ Enter a single keyword:

On the SYSEXT utility screen, in the **Keyword** field, type in a keyword and press ENTER.

A list of all APIs to which the specified keyword applies appears.

##### ■ Select a single keyword from a list of valid keywords:

1. On the SYSEXT utility screen, in the **Keyword** field, enter an asterisk (\*) and press ENTER.

The **List Keywords** window with a list of all keywords available for all APIs appears as shown in the [example](#) below.

If required, press PF8 to scroll down the list of keywords or press PF7 to go to the beginning of the list.

2. In the **M** column, next to the desired keyword, type in any character and press ENTER.

Or: In the **Direct** field, enter the desired keyword and press ENTER.

A list of all APIs to which the specified keyword applies appears.

##### ■ Select multiple keywords from a list of valid keywords:

1. On the SYSEXT utility screen, in the **Keyword** field, enter an asterisk (\*) and press ENTER.

The **List Keywords** window with a list of all keywords available for all APIs appears as shown in the [example](#) below.

If required, press PF8 to scroll down the list of keywords or press PF7 to go to the beginning of the list.

2. In the **M** column, next to the desired keywords, type in an asterisk (\*) and press ENTER.

The **Keyword** window appears with a list of all APIs to which the first keyword selected applies (here: \*LANGUAGE) as shown in the example below:

```

11:08:58          ***** NATURAL SYSEXT UTILITY *****
2005-11-10
User SAG          - Menu -          Library
SYSEXT

Cm +-----List Keywords-----+ +-----Keyword *LANGUAGE-----+
  Prod
  _ !                      ! !                      !
  NAT
  _ ! M Keyword            ! ! Object                !
  NAT
  _ ! _ *DATX              ! ! USR0020P              !
  NAT
  _ ! _ *ERROR-NR         ! ! USR0120P              !
  NAT
  _ ! _ *ERROR-TA        ! ! USR0320P              !
  NAT
  _ ! * *LANGUAGE         ! ! USR1027P              !
  NAT
  _ ! _ *LEVEL            ! ! USR1030P              !
  NAT
  _ ! _ *STEPLIB          ! ! USR2034P              !
  NAT
  _ ! * *TIMESTAMP       ! ! USR3320P              !
  NVS
  _ ! _ *TIMX             ! !                      !
  NAT
  _ ! _ ABEND             ! !                      !
  NAT
  _ !                      ! !                      !
  NAT
  _ ! Direct ... _____ ! !                      !
R NAT
  _ !                      ! !                      !
  NAT
  +-----+ +-----+

Keyword .. *_____

Command ==>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
```

```

          Help      Exit      --      +
Canc

```

3. Press ENTER to open the **Keyword** window for the next keyword selected (here: \*TIMESTAMP).
4. Press ENTER or PF3 to return to the **List Keywords** window.
5. Press PF3 to leave all windows and return to the SYSEXT utility screen.

► **To list keywords relevant to a specific API**

- Next to the API required, enter the line command K. The **Keywords** window appears for the specified API with a list of all keywords that are relevant to this API as shown in the example below:

```

11:24:27          ***** NATURAL SYSEXT UTILITY *****          2005-11-10
User SAG          - Menu -          Library SYSEXT

Cmd  Source      Interface  Comment          Prod
K   USR0010P  USR0010  +- -Keywords for USR0010P--+  NAT
_   USR0011P  USR0011  !                   ! file          NAT
_   USR0020P  USR0020  ! DBID              ! FNAT / FUSER  NAT
_   USR0040P  USR0040  ! ENVIRONMENT       !               NAT
_   USR0050P  USR0050  ! FILE              !               NAT
_   USR0060P  USR0060  ! FNAT              ! m 'FNAT' to 'FUSER' NAT
_   USR0070P  USR0070  ! FNR               ! YSTEM'        NAT
_   USR0080P  USR0080  ! FUSER            ! r Contents     NAT
_   USR0100P  USR0100  ! SYSPROF          !               NVS
_   USR0120P  USR0120  !                   ! Message       NAT
_   USR0210P  USR0210  !                   ! object        NAT
_   USR0220P  USR0220  !                   ! essage       NAT
_   USR0320P  USR0320  +-----+ sage from FNAT or FUSER NAT
_   USR0330P  USR0330N  Read Natural Object Directory  NAT

Keyword .. _____

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---;

```

If required, press PF8 to scroll down one page in the list. Press PF7 to go to the beginning of the list.

## Using a Natural API

If you want to use a Natural API contained in the SYSEXT system library, perform one of the following steps:

- Define the SYSEXT system library in the FNAT system file as a steplib library for the user library which contains the Natural objects that use this API. Thus, no API-specific actions are required when upgrading your Natural version.
- Copy the required API to the SYSTEM system library in the FNAT system file. Thus, you only need to check a single library for APIs when upgrading your Natural version.
- Copy the required API to the SYSTEM library in the FUSER system file (not recommended).
- Copy the required API to the user library (or one of its steplibs) in the FUSER system file which contains the Natural objects that use this API (not recommended).

An API can only be used in the Natural version with which it is delivered. It cannot be executed in any other Natural version. Therefore, it is strongly recommended that APIs are stored only in the FNAT system file.

## List of Natural APIs

The following list of Natural Application Programming Interfaces (APIs) available with the SYSEXT utility is for general information only. For the current status of APIs, execute the system command SYSEXT.

The list columns correspond to the columns contained on the SYSEXT utility screen. For explanations on the columns, see the section [SYSEXT Utility Screen](#).

For detailed information on the use of APIs that apply to Natural Remote Procedure Call (RPC), refer to the section *Application Programming Interfaces for Use with Natural RPC* in the *Natural Remote Procedure Call* documentation.

Interface	Comment	Product
USR0010N	Get SYSPROF information	NAT
USR0011N	Information about logical file	NAT
USR0020N	Read any error text from FNAT / FUSER	NAT
USR0040N	Get type of last error	NAT
USR0050N	Get SYSPROD information	NAT
USR0060N	Copy LFILE definition from FNAT to FUSER	NAT
USR0070P	Default Editor Profile SYSTEM	NAT

Interface	Comment	Product
USR0080N	Handle type/name of editor contents	NAT
USR0100N	Control LRECL	NVS
USR0120N	Read Natural short error message	NAT
USR0210N	Save, catalog or stow Natural object	NAT
USR0220N	Read Natural long error message	NAT
USR0320N	Read user short error message from FNAT or FUSER	NAT
USR0330N	Read Natural object directory	NAT
USR0340N	Natural buffer pool interface	NAT
USR0341N	Natural buffer pool interface	NAT
USR0350N	Read current recording flags	NAT
USR0360N	Modify user short error message	NAT
USR0400N	Number of rows affected by searched UPDATE	NDB
USR0420N	Read user long error message from FUSER	NAT
USR0421N	Update user long error message on FUSER	NAT
USR0500N	Display a string in the title bar of a window	NAT
USR0600N	Display program level information	NAT
USR0610N	Display DB error information	NAT
USR0620N	Translate strings	NAT
USR0622N	Reset error counter	NAT
USR1002N	Save and restore Natural environment parameter	NAT
USR1005N	Information about some Natural system parameters	NAT
USR1006N	Support skip-sequential processing	NVS
USR1007N	Display work file and printer file assignments	NAT
USR1009N	Convert *TIMESTMP to numeric variable	NAT
USR1011N	Wildcard / asterisk check (short)	NAT
USR1012N	Read dynamic error part :1:	NAT
USR1013N	Display current character set	NAT
USR1014N	*** Line Calculator ***	NAT
USR1016N	Display error level for copycode	NAT
USR1017N	Add CATAL call to CATAL control list	NAT
USR1018N	Dynamic OPEN	NVS
USR1019N	Get SYSBUS information	NAT
USR1020N	Add user short error message to FUSER	NAT
USR1021N	Wildcard / asterisk check (long)	NAT
USR1022N	Type of data base	NAT
USR1023N	Date and time variables conversion	NAT

Interface	Comment	Product
USR1024N	Read results of CATALL	NAT
USR1025N	Handle multiple steplibs	NAT
USR1026N	Display RETURN information	NAT
USR1027N	Search user short error message	NAT
USR1028N	Bit/byte conversion	NAT
USR1029N	Get type of Natural object	NAT
USR1030N	Language code conversion	NAT
USR1031N	Check object name	NAT
USR1032N	List cataloged Natural objects with type	NAT
USR1033N	Find DBID/FNR of a cataloged DDM	NAT
USR1034N	Display NTTF file table	NAT
USR1035N	Maintain objects via the Software AG editor engine	NAT
USR1036N	Maintain the user profile of the Software AG editor	NAT
USR1037N	Information about Natural ABEND data	NAT
USR1038N	Retrieve characteristics of the current platform	NAT
USR1040N	Get or set the UDB parameter	NAT
USR1041N	Sample error transaction program (*ERROR-TA)	NAT
USR1042N	Get or set the value of the UPDATE command	NAT
USR1043N	Perform Adabas direct calls	NAT
USR1047N	Dynamic switch of file name	NVS
USR1048N	Modify PF-key labels	NAT
USR1050N	Get or set a work file name	NAT
USR1051N	Interface to various Predict data	PRD
USR1054N	List libraries	NAT
USR1055N	List objects in a library	NAT
USR1056N	List DDMs on the FDIC file or in a library	NAT
USR1057N	Read a Natural source into an array	NAT
USR1058N	Read a DDM source into an array	NAT
USR1066N	Display the Natural "Executing ..." message	NAT
USR1067N	Check library name	NAT
USR1068N	Get or reset the value of DBMS calls	NAT
USR1070N	Issue operator commands to Entire Net-Work	NAT
USR1071N	Set user ID and password for RPC	RPC
USR1072N	Get command ID of a retain set	NAT
USR2001N	Read information about last error	NAT
USR2002P	Default text strings for Help function	NAT

Interface	Comment	Product
USR2003P	Default settings for MAINMENU	NAT
USR2004N	Information about logical file	NAT
USR2005N	Access to the internal file translation table	NAT
USR2006N	Get detailed message information	NAT
USR2007N	Set/get RPC default server information	RPC
USR2008N	Dynamic OPEN for VSAM/ISAM datasets	NVS
USR2009N	Read dynamic error part :1:	NAT
USR2010N	Display DB error information	NAT
USR2011N	Get or set a work file name	NAT
USR2012N	Get value of system variable *NET-USER	NAT
USR2013N	Get SYSPROF information	NAT
USR2014N	Maintain objects via the Software AG editor engine	NAT
USR2015N	EBCDIC or ASCII translation table for Natural RPC	NAT
USR2016N	Copy map profile from FNAT to FUSER	NAT
USR2017N	Activate map profile handling from FUSER	NAT
USR2018N	Read Natural object directory	NAT
USR2019N	Read/save Natural source into/from the source area	NAT
USR2020N	Perform Adabas direct calls	NAT
USR2021N	Dynamic dataset allocation	NAT
USR2022N	Insert GUID into saved data area  <b>Note:</b> See the section <i>Obsolete Application Programming Interface</i> in the current <i>Natural Release Notes</i> .	NAT
USR2023N	Type of data base (2 bytes)	NAT
USR2026N	Get TECH information	NAT
USR2027N	Define a wait interval for the session	NAT
USR2028N	Output the Natural version	NAT
USR2029N	Dynamic file allocation (BS2000/OSD)	NAT
USR2030N	Read dynamic error parts :1;...	NAT
USR2031N	Get SYSPROD information	NAT
USR2032N	Support of commit for CLOSE CONVERSATION	RPC
USR2033N	Information about PRD List Xref sets	PRD
USR2034N	Read system or user error text from either FNAT or FUSER	NAT
USR2035N	Support of SSL	RPC
USR2036N	Convert *TIMESTAMP to numeric variable	NAT
USR2071N	Support of EntireX Security on client side	RPC
USR2072N	Support of EntireX Security on server side	RPC

Interface	Comment	Product
USR2073N	Ping or terminate an RPC server	RPC
USR2075N	Terminate EntireX Broker Service	RPC
USR3001N	List Roll Server directory entries	NAT
USR3002N	Delete Roll Server directory entries	NAT
USR3005N	Process documentation objects	PRD
USR3013N	Get SYSPROF information	NAT
USR3025N	Handle multiple steplibs	NAT
USR3320N	Find user short error message from FNAT or FUSER	NAT
USR4001N	Set Natural profile parameter PROGRAM  (corresponds to USR6002N but can be used on mainframes only)	NAT
USR4002N	Retrieve variables of the current system	NAT
USR4003N	Retrieve Natural stack information (alphanumeric)	NAT
USR4004N	Retrieve dynamic Natural profile parameters	NAT
USR4005N	Read all current key settings	NAT
USR4006N	Select (and cancel) Natural sessions under CICS	NCI
USR4007N	Get/set current value of profile parameter SYNERR	NAT
USR4008N	Set library for RPC execution	RPC
USR4009N	Set parameters for EntireX	RPC
USR4010N	Retrieve runtime settings of server	RPC
USR4011N	Create A20 hash value for variable input	NAT
USR4201N	Process data area sources	NAT
USR4202N	Get code page table information	NAT
USR4203N	Check for lock, lock or unlock Natural object	NAT
USR4204N	Set or reset CICS Container Name	NCI
USR4206N	List objects in a library and return directory information	NAT
USR4207N	Return machine characteristics as provided by NATQVS	NAT
USR4208N	Read or write a Natural resource	NAT
USR4209N	Return short name of subroutine	NAT
USR4210N	Base64 conversion of alpha and binary bytes	NAT
USR4211N	Get DBCS characters	NAT
USR4212N	Analyze data area	NAT
USR4213N	String handling for DBCS Support	NAT
USR4214N	Enhanced program level information	NAT
USR4215N	Return a list of resources of a Natural library	NAT
USR4216N	Return a list of Natural objects of a Natural library	NAT
USR4340N	List buffer pool contents or buffer pool cache	NAT

Interface	Comment	Product
USR4341N	Blacklist maintenance of Natural buffer pools	NAT
USR4371N	Set user ID and ETID for RPC	RPC
USR6002N	Get the current values of some internal counters	NAT
USR6204N	Set Natural profile parameter PROGRAM (corresponds to USR4001N but can be used on all platforms)	NAT
USR6303N	Retrieve Natural stack information (Unicode)	NAT
USR6304N	Set/get reliable state for RPC execution	RPC
USR6305N	Commit/rollback reliable RPC message(s)	RPC
USR6306N	Status of UOWs of current EntireX Broker user	RPC



# 84 SYSEXV Utility

---

- Executing Example Programs of Current Versions ..... 486
- Executing Example Programs of Non-current Versions ..... 486
- Terminating the SYSEXV Utility ..... 487

The utility SYSEXV provides example programs to illustrate new features of current Natural versions. All programs are available as source objects and display a detailed functionality description when they are executed. Example programs for non-current Natural versions are also available.

This chapter covers the following topics:

## Executing Example Programs of Current Versions

---

### ▶ To execute an example program of a current version

- 1 Enter the following system command:

```
SYSEXV
```

Or:

Log on to the SYSEXV library:

```
LOGON SYSEXV
```

and continue by entering

```
VERSION
```

As a result, a menu is displayed from which you can choose a Natural version.

- 2 Choose a version from the menu.

As a result, a commented menu is displayed from which you can choose an example program relevant to the selected version.

- 3 Choose a program.

As a result, the program is executed.

## Executing Example Programs of Non-current Versions

---

### ▶ To execute an example program of a non-current version

- 1 Log on to the SYSEXV library:

```
LOGON SYSEXV
```

- 2 List the text object A-README.



**Note:** Do not modify the text object A-README.

A-README lists and comments example programs for non-current versions.

- 3 Choose a program and execute it.

## Terminating the SYSEXV Utility

---

### ▶ To terminate the SYSEXV utility

- In the SYSEXV utility menu, choose **Exit**.

Or:

Press PF3 or PF12.



# 85

## SYSMAIN Utility - Object Maintenance

---

The SYSMAIN utility is used to perform object maintenance functions such as copy, move, replace and delete.

- **General Information on SYSMAIN**
- **Invoking and Terminating SYSMAIN**
- **Using Menu Functions and Commands**
- **Processing Programming Objects**
- **Processing Debug Environments**
- **Processing Error Messages**
- **Processing Profiles**
- **Processing Rules**
- **Processing DL/I Subfiles**
- **Processing DDMs**
- **Keywords and Variables in Direct Commands**
- **Special Commands Issued to SYSMAIN**
- **Processing Status and Error Notification**
- **Special Considerations for Administrators**



# 86

## General Information on SYSMAIN

---

■ Basic SYSMAIN Functionality .....	492
■ Object Types and Storage Location .....	493
■ Overview of Functions .....	493

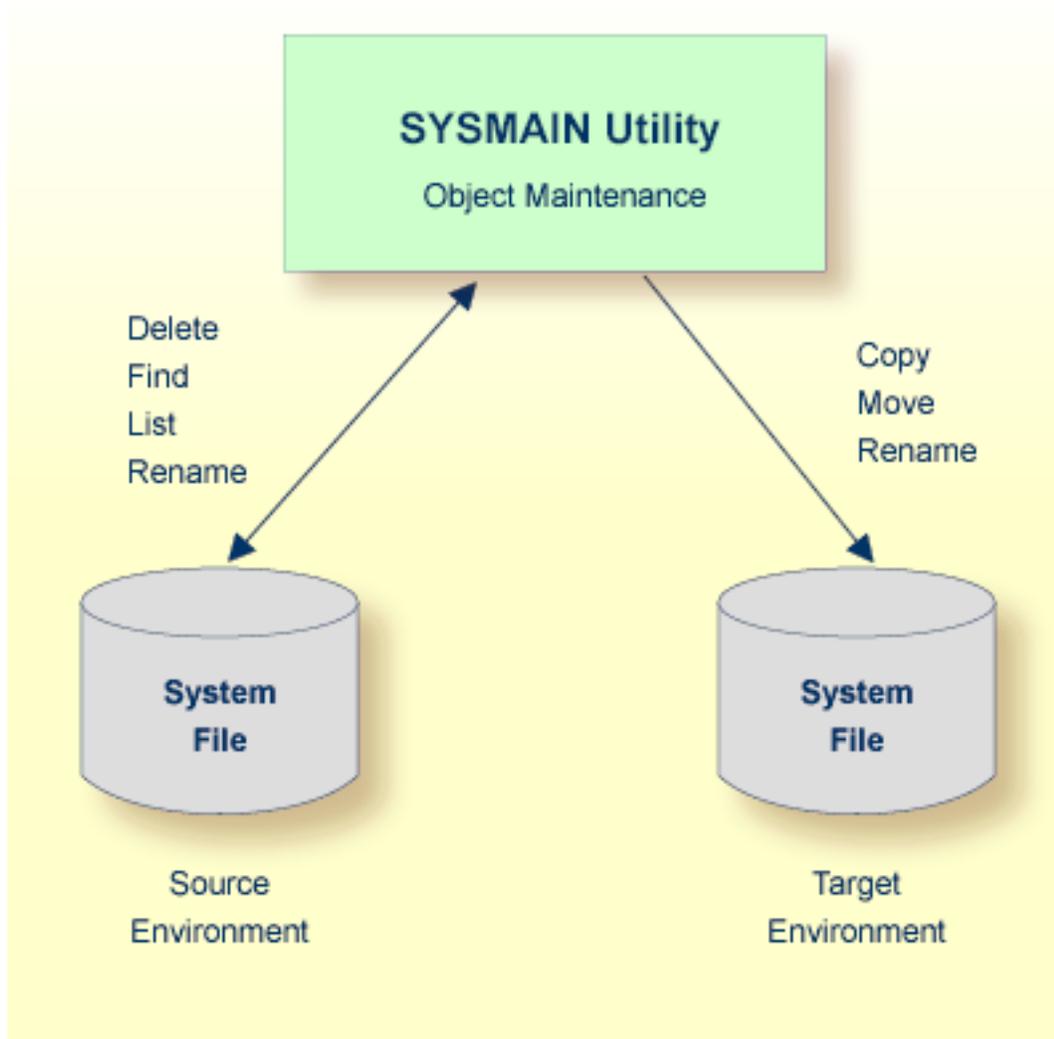
The SYSMAIN utility is used to maintain Natural objects in a Natural environment or across different environments.

This section provides basic information on object maintenance functions and the types of Natural object that can be processed with SYSMAIN.

## Basic SYSMAIN Functionality

---

The following diagram is a basic illustration of the SYSMAIN functionality:



The SYSMAIN utility copies or moves Natural objects from a source environment to a target environment and performs object operations such as delete in a source environment. The rename function can be performed in both a source and a target environment. An overview of the functions

and the Natural objects to which they apply is provided in the following section. The SYSMAIN utility functions are available online and in batch mode.

A source or target environment is an FNAT, FUSER or FDIC system file contained in an Adabas database or a VSAM file system. Natural objects on the FNAT or FUSER system file can be contained in libraries as indicated in the following section.

## Object Types and Storage Location

The types of Natural object that can be maintained with SYSMAIN are listed in the table below. The location of a Natural object depends on the object type as indicated in the table below:

Objects/Subfiles	Description	Location
Programming objects	All Natural object types that are stored in a Natural library except debug environments and error messages.	Libraries in FUSER and FNAT
Debug environments	Debug environments for online program testing.	Library in FUSER
Error messages	Short and extended (long) Natural system error messages and user-defined error messages.	Libraries in FUSER and FNAT
Profiles	Editor profiles, map profiles, device profiles and parameter profiles.	FNAT
Rules	Automatic and free rules.	FDIC
DDMs	Data definition modules.	FDIC
DL/I Subfiles	Natural NSBs, NDBs and UDFs.	FDIC

## Overview of Functions

The functions provided by SYSMAIN are listed in the table below. The table also indicates which function is valid for each type of Natural object. For details on each function, refer to [Description of Functions](#) in *Using Menu Functions and Commands*.

Function	Programming Object	Debug Environment	Error Message	Profile	Rule	DDM	DL/I Subfile
<b>Copy</b> Copy object from one system file to another.	x	x	x	x	x	x	x
<b>Delete</b> Delete object from a system file.	x	x	x	x	x	x	x
<b>Find</b> Locate object in a system file.	x		x				

Function	Programming Object	Debug Environment	Error Message	Profile	Rule	DDM	DL/ Subfile
<b>List</b> Display a range of objects in a system file.	x	x	x	x	x	x	x
<b>Move</b> Transfer object from one system file to another.	x	x	x	x	x	x	x
<b>Rename</b> Give an object a new name, and (optionally) transfer it to a new system file.	x	x	x	x	x		

# 87 Invoking and Terminating SYSMAIN

---

- Invoking SYSMAIN Online or Batch ..... 496
- Invoking SYSMAIN with Appl. Programming Interface ..... 497
- Terminating SYSMAIN ..... 498

This section describes how to invoke and terminate the SYSMAIN utility online, in batch or by using an Application Programming Interface.

## Invoking SYSMAIN Online or Batch

The following instructions describe the methods of invoking the SYSMAIN utility by using a direct command (online or batch) or menu functions.

▶ **To invoke SYSMAIN online**

- From any library, enter the following Natural system command:

```
SYSMAIN
```

Or:

From the **Natural Main Menu**, invoke the **Maintenance and Transfer Utilities** menu and choose **Transfer Objects to Other Libraries**.

A **SYSMAIN Utility Main Menu** similar to the example below appears:

```

10:39:24          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG                - Main Menu -

      Code  Object                                Code  Function

      A    Programming Objects                    C    Copy
      D    Debug Environments                      D    Delete
      E    Error Message Texts                    F    Find
      P    Profiles                                L    List
      R    Rules                                    M    Move
      S    DL/I Subfiles                           R    Rename
      V    DDMs                                    ?    Help
      ?    Help                                    .    Exit
      .    Exit

Object Code .. A          Function Code .. _

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Copy Del Find List Move Ren
    
```

The current setting of the system variable \*LIBRARY-ID is passed to SYSMAIN and used as the default source library for processing programming objects and debug environments.

▶ **To invoke SYSMAIN in batch**

- Use the following direct command:

```
SYSMAIN
```

followed by one or more command strings. See also *Issuing Direct Commands*.

## Invoking SYSMAIN with Appl. Programming Interface

MAINUSER is an Application Programming Interface (API) that can be used to perform SYSMAIN functions directly from any user-written object (for example, from a subroutine, program or subprogram) without going through the normal steps of invoking SYSMAIN.

Upon completion of processing of the SYSMAIN functions, the utility is terminated and control is returned to the program, subprogram or subroutine from which the request was issued.

MAINUSER is supplied as a cataloged object of the type subprogram in the system library SYSMAIN. MAINUSER can be used in either online or batch mode.



**Note:** MAINUSER must *not* be located in a user library. You must therefore copy it to the library SYSTEM on the system file FNAT or FUSER or to any SYS-prefixed library which is the steplib for the application.

## Invoking MAINUSER

MAINUSER is invoked with the CALLNAT statement and its relevant parameters (see also *CALLNAT* in the *Natural Statements* documentation). MAINUSER must *not* be invoked from within the library SYSMAIN.

▶ **To invoke MAINUSER**

- Issue a CALLNAT statement that contains the following syntax elements:

```
CALLNAT 'MAINUSER' command error message library
```

where the variable values denote the following parameters:

Parameter	Natural Data Format/Length	Explanation
<i>command</i>	A250	The direct command string to be executed by SYSMAIN.
<i>error</i>	N4	The return code issued by SYSMAIN at the end of processing to indicate a normal end of processing or an error.
<i>message</i>	A72	The message corresponding to the error given online.
<i>library</i>	A8	The library containing SYSMAIN. If not specified, the default is SYSMAIN.

An example of a callable routine is the program MAINCALL in the library SYSMAIN.

## Terminating SYSMAIN

---

### ▶ To terminate SYSMAIN

- In the Command line of any SYSMAIN screen, enter one of the following direct commands:

```
.

```

(a period)

or

```
END

```

or

```
QUIT

```

Or:

Press PF3 (Exit), if required repeatedly.

Or:

In batch mode, use the either of the following direct command:

```
END

```

or

```
QUIT

```

See also *Issuing Direct Commands* in the section *Using Menu Functions and Commands*.



**Important:** Do not terminate the SYSMAIN utility with the terminal command `%%`, because the environment may not be reset correctly.



# 88

## Using Menu Functions and Commands

---

■ Performing Menu Functions .....	502
■ Executing Commands .....	505
■ Description of Functions .....	506
■ Function Processing and Reporting .....	514
■ SYSMAN Online Help .....	519

A SYSMAIN function can be performed in either menu or command mode.

This section describes how to use a SYSMAIN menu or a direct command and the options provided when performing a SYSMAIN function.

## Performing Menu Functions

In menu mode, you perform a SYSMAIN function by entering codes for the object type to be processed and the function to be performed. As an alternative to entering a function code, you can press a corresponding PF key.

### ▶ To perform a SYSMAIN menu function

- In the **SYSMAIN Utility Main Menu**, in the **Object Code** field, enter the one-letter code that corresponds to the objects required (in the example below, **A** for **Programming Objects**) and, in the **Function Code** field, enter the one-letter code that corresponds to the function required (in the example below, **C** for **Copy**). As an alternative to entering the function code, you can press the PF key that corresponds to the function (here: PF4); see also the list of **PF keys**.

```

10:55:14          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG                - Main Menu -

      Code  Object                                Code  Function
      A    Programming Objects                    C    Copy
      D    Debug Environments                      D    Delete
      E    Error Message Texts                     F    Find
      P    Profiles                                L    List
      R    Rules                                    M    Move
      S    DL/I Subfiles                            R    Rename
      V    DDMS                                     ?    Help
      ?    Help                                     .    Exit
      .    Exit

Object Code .. A                Function Code .. C

Command ==>
Enter-PF1---PF2---PF3-PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Copy Del Find List Move Ren

```

The appropriate object-type specific menu is then displayed as shown in the example of a **Copy Programming Objects** menu below:

```

17:28:52          ***** NATURAL SYSMAN UTILITY *****          2005-08-10
User SAG          - Copy Programming Objects -

                                Code  Function

                                A    Copy All/Individual Objects
                                C    Copy only Cataloged Objects
                                S    Copy only Saved Objects
                                W    Copy only Stowed Objects
                                ?    Help
                                .    Exit

                                Code ..... A          Sel. List ... Y
Object Name ..... *_____ Type ..... _____
                                Set Number .. __      XREF .. N
Source Library ... OLDLIB__ Database .... 10__ File .. 50__
Target Library ... NEWLIB__ Database .... 10__ File .. 60__
Options Replace ... N          Criteria .... N

Command ==>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Copy Del Find List Move Ren Fsec Fdic Fnac

```

The fields provided in an object-type specific menu depend on the SYSMAIN function performed. Fields that do not apply to a particular function are not displayed on the respective screen.

The fields are used to specify object selection criteria and processing options. For explanations of the fields, refer to the object-type specific sections of the SYSMAIN utility documentation.

## Using PF Keys

You can use PF keys to perform SYSMAIN menu functions. PF keys which are not valid within a menu are not displayed for this menu. The PF keys are summarized in the table below:

PF Key	Name	Function
PF1	Help	Display online help depending on the current cursor position.  If the cursor is positioned in the <b>Object Code</b> or <b>Function Code</b> field, SYSMAIN general help is displayed.  If the cursor is in another field, field-specific help is displayed. See also <a href="#">SYSMAIN Online Help</a> .
PF2	Menu	Display the <b>SYSMAIN Utility Main Menu</b> .
PF3	Exit	Return to the previous screen. If you press PF3 on the <b>SYSMAIN Utility Main Menu</b> , SYSMAIN is terminated.
PF4	Copy	Perform the <b>Copy</b> function for the object specified.
PF5	Del	Perform the <b>Delete</b> function for the object specified.
PF6	Find	Only applies to programming objects, error messages and rules.  Perform the <b>Find</b> function for the object specified.
PF7	List	Perform the <b>List</b> function for the object specified.
PF8	Move	Perform the <b>Move</b> function for the object specified.
PF9	Ren	Not applicable to DDMs and DL/I subfiles.  Perform the <b>Rename</b> function for the object specified.
PF10	Fsec	Invoke the security screen for specifying the Adabas security information of the FSEC system file if Natural Security is installed.
PF11	Fdic	Not applicable to debug environments, error messages and profiles.  Only applies to programming objects, rules and DDMs.  Invoke the security screen for specifying the Adabas security information of the FDIC system file. If Predict is installed, you can also specify a Predict set user when performing the <b>Copy</b> , <b>Delete</b> , <b>Move</b> or <b>Rename</b> function on programming objects.

PF Key	Name	Function
PF12	Fnat	Not applicable to rules, DL/I subfiles and DDMs.  Invoke the security screen for specifying the system file information of the FNAT and/or FUSER system files.

## Executing Commands

This section provides instructions for executing a SYSMAIN function by using a direct command or issuing a system command from within the SYSMAIN utility.

A direct command is used to perform a SYSMAIN function in either online or batch mode.

In batch mode, a report is automatically provided that shows the status of objects processed. This report can also be displayed in online mode: see [Online Report Mode](#).

- [Issuing Direct Commands](#)
- [Using the SYSMAIN Command Line](#)

### Issuing Direct Commands

Direct commands consist of a string of keywords that represent parameters. A period (.) indicates the end of a command. If this character is detected anywhere within a command string, all subsequent data is ignored.

The syntax that applies when issuing a direct command is described in the object-type specific sections of the SYSMAIN Utility documentation. The respective keywords and variables are explained in [Keywords and Variables in Direct Commands](#). Examples of direct commands are shown in the object-type specific sections, the SYSMAIN online help and on a **Selection** screen (see the [example screen](#) in *Selective Processing*).

#### ▶ To issue a direct command online

- At any Natural command prompt, enter the following:

```
SYSMAIN
```

followed by a command string where each keyword can be delimited by a blank character instead of the delimiter.

Or:

In the Command line of any SYSMAIN menu, enter a direct command string where each keyword can be delimited by a blank character instead of the delimiter.

Or:

From within a Natural object, invoke the subprogram MAINUSER with the direct command string as a parameter where each keyword can be delimited by a blank character instead of the delimiter.

► **To issue a direct command in batch**

- 1 Use the command `SYSMAIN` and specify a command string in either of the following ways:
  1. The command string follows the `SYSMAIN` command in the same input line; each keyword in the command string can be delimited by a blank character instead of the delimiter.
  2. The command string follows the `SYSMAIN` command in the next input line; each keyword in the command string must be delimited by the delimiter and *not* by a blank character.

If the direct command string is longer than one single line, the characters `CF` (see also the session parameter `CF` described in *CF - Character for Terminal Commands* in the *Parameter Reference* documentation) must be placed at the end of the line to continue with the direct command in the next line.

- 2 If you want to execute other Natural commands after the `SYSMAIN` command(s), you must first terminate `SYSMAIN` by using the direct command `END` or `QUIT`.

### Using the `SYSMAIN` Command Line

In the Command line of any `SYSMAIN` menu, you can enter one of the following:

- A direct command for processing a `SYSMAIN` function.
- A special command to the `SYSMAIN` utility described in *Special Commands Issued to SYSMAIN*.
- A system command. If the command is not uniquely identifiable as a system command, it must be preceded by two slashes (`//`). See also **Remarks** in *MAINEX05 - User Exit Routine for Verification of Direct Commands*.

## Description of Functions

---

The functions provided in a `SYSMAIN` menu or as direct commands are described in the following section. For each function provided in a menu, there is a corresponding direct command with the same name. Exception: **Help**.

For the syntax that applies when using a direct command, refer to the object-type specific sections of the *SYSMAIN Utility* documentation.

For the special commands that can be issued to the SYSMAIN utility, refer to *Special Commands Issued to SYSMAIN*.

Function/Command	Explanation
<b>Copy</b>	<p>Copies a Natural object from a source environment to a target environment. The object remains unchanged in the source environment.</p> <p>If the target environment already contains an object with the same name (or in the case of an error message, the same number) as the object to be copied, the specified object is not copied.</p> <p>You can use the replace option (see <i>Using the Replace Option</i>) to overwrite an object in the target environment.</p>
<b>Delete</b>	<p>Deletes a Natural object from a source environment. During online <b>automated processing</b> (see <i>Function Processing and Reporting</i>), a confirmation window is displayed, which gives you the option of continuing or terminating the function.</p> <p>For error messages: If Natural Security is installed, the delete function is disallowed for system error messages. When attempting to delete a system error message the following Natural system message occurs: 4897:Invalid error application specified.</p>
<b>Find</b>	<p>Only applies to programming objects and error messages.</p> <p>Locates one or more programming objects or error messages in a source environment.</p> <p>During online processing, a window showing the library currently scanned is displayed.</p> <p>In menu mode, the find function also provides the option to list and select libraries or reduce the number of objects displayed on a selection list: see <i>Listing and Selecting Libraries</i> or <i>To shorten a selection list (Using a Selection List)</i> respectively.</p>
<b>List</b>	<p>Displays a range of Natural objects in a source environment.</p> <p>For programming objects, debug environments and error messages: In menu mode, the list function also provides the option to list and select libraries or reduce the number of objects displayed on a selection list: see <i>Listing and Selecting Libraries</i> or <i>To shorten a selection list (Using a Selection List)</i> respectively.</p> <p>For programming objects: In batch mode, you can obtain a list of library names by using the direct command LISTLIB.</p>
<b>Move</b>	<p>Transfers a Natural object from a source environment to a target environment. The object is deleted from the source environment and added to the target environment. If the target environment already contains an object with the same name (or in the case of an error message, the same number) as the object to be moved, the specified object is not moved.</p> <p>During online automated processing, a confirmation window is displayed, which gives you the option of continuing or terminating the function.</p> <p>You can use the replace option (see <i>Using the Replace Option</i>) to overwrite the object in the target environment.</p>

Function/Command	Explanation
<b>Rename</b>	<p>Not applicable to DL/I subfiles and DDMs.</p> <p>Gives an object a new name (or in the case of an error message, a new number) using either of the following options:</p> <ol style="list-style-type: none"> <li>1. Rename the object in the source environment.</li> <li>2. Rename the object and transfer it to another (that is, target) environment. You cannot rename objects in both a source and a target environment at the same time. The rename function can only be performed separately for each environment.</li> </ol> <p>The rename function deletes the original object in the source environment; therefore, you are prompted with an option to retain the original object. If the original object is to be retained, it is not deleted.</p> <p>If the target environment already contains an object with the same name (or in the case of an error message, the same number) as the object to be renamed, the specified object is not renamed. You can use the replace option (see <a href="#">Using the Replace Option</a>) to overwrite the object in the target environment.</p> <p>Only a single programming object, environment, profile or rule can be renamed using automated processing. If a range of programming objects, environments, profiles or rules is to be renamed, <a href="#">selective processing</a> (see below) must be used.</p> <p>For error messages: When renumbering a range of error messages within a single (source) library, the range values must not overlap. For example, it is not possible to rename error numbers 1 - 6 as new error numbers 5 - 10. A range of error messages can be renamed with <a href="#">automated processing</a> (see below). When large ranges of error messages are being processed, the processing of messages may require significant resources. In such cases, batch-mode processing may be preferable.</p>
<b>Help</b>	<p>Provides help information on SYSMAIN: see <a href="#">SYSMAIN Online Help</a>.</p> <p>The help function is only available in a SYSMAIN menu; there is no corresponding direct command.</p>
<b>Exit</b>	<p>Terminates the SYSMAIN utility.</p> <p>PF3 (Exit) also terminates SYSMAIN if pressed in the <b>SYSMAIN Utility Main Menu</b>.</p>

This section covers the following topics:

- [Using the Replace Option](#)

- [Listing and Selecting Libraries](#)

## Using the Replace Option

If the target environment already contains an object with the same name as the object to be copied, moved or renamed, the specified object is not processed and processing continues with the next object. You can use the replace option to override this default feature and overwrite the object in the target environment.

If a programming object is replaced, it is also deleted from the Natural buffer pool; any existing cross-reference records are also deleted if Predict is installed.

### ▶ To activate the replace option in command mode

- In the command string, specify the keyword `REPLACE`.

### ▶ To activate the replace option in menu mode

- 1 In a `SYSMAIN` menu, in the **Replace** field, enter a `Y`.
- 2 Perform a function. If the **Sel. List** (Selection List) option has been set to `N`, a window appears where you can choose whether to confirm every replace operation before it is performed: Enter a `Y` to confirm each replacement, or press `ENTER` to continue processing without confirmation.
- 3 If you entered a `Y` to confirm a replacement, for each object that is to be replaced a window appears where you can enter one of the following characters:
  - `Y` replaces the object indicated in the window,
  - `N` does not replace the object indicated in the window (this is the default setting),
  - A period (`.`) terminates function processing. Alternatively, you can press `PF3`.

## Listing and Selecting Libraries

This option only applies to programming objects, debug environments and error messages.

When using the list function, you can invoke a selection list of libraries that contain the specified object(s). This also applies to the find function if the specified object is contained in several libraries.

### ▶ To invoke a list of libraries for programming objects

- In a **List** menu, in the **Library** field, enter a range of library names (see also [Specifying a Range of Names](#)), and, in the **Name** field, enter an object name or a range of names.

If you enter an asterisk (\*) in the **Library** field (list function only), a list of all libraries available in the specified system file will be displayed.

In the example of a **Library Selection** screen shown below, all libraries with names that start with LIB are selected:

```

21:23:03          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG          - Library Selection -

LIST ALL * IN LIB* WHERE DBID 10 FNR 32

  C Library      S/C      C Library      S/C      C Library      S/C      C Library      S/C
  - - - - -      - - -      - - - - -      - - -      - - - - -      - - -      - - - - -      - - -
  _ LIB1         S/C      _ LIB2         S/C      _ LIB3         C          _ LIB4         S/C
  _ LIB5         S          _ LIB6         S/C      _ LIB7         S/C      _ LIB8         S/C
  _ LIB9         S/C      _ LIB10        S/C      _ LIB11        S/C      _ LIB12        S/C
  _ LIB13        S          _ LIB14        S/C      _ LIB15        C          _ LIB16        S/C
  _ LIB17        S/C      _ LIB18        S/C      _ LIB19        S/C      _ LIB20        S/C

                                     Object Start Value: *

Enter options (above), or '?' (Help) or '.' (Exit): _
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Copy Del Find List Move Ren Canc
  
```

The **Library** column lists all libraries that match the specified range. The **S/C** column indicates whether a library contains saved (source) objects and/or cataloged objects.

For a list of line commands available on the **Library Selection** screen, see [Selection Lists for Programming Objects](#).

When using the find function, the **Library Selection** screen appears if the specified object is found in several libraries.

The **Library Selection** screen also appears when you leave a **Find Selection** or a **List Selection** screen by pressing PF3 (Quit).

▶ **To invoke a list of libraries for debug environments**

- In the **Debug Environments** menu:

In the **Code** field, enter an L and, in the **Environment Name** field, enter the name of a debug environment or a range of names and, in the **Source Library** field, enter a range of library names (see also [Specifying a Range of Names](#)).

If you enter an asterisk (\*) in **Environment Name** and **Source Library**, a list of all libraries that contain debug environments in the specified system file will be displayed.

In the example of a **Library Selection** screen shown below, all libraries with names that start with L were selected:

```

09:55:08          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG          - Library Selection -
LIST DEBUG * IN LIB* FROM DBID 10 FNR 32

C Library          C Library          C Library          C Library
- - - - -          - - - - -          - - - - -          - - - - -
_ LIB1             _ LIB5             _ LIB9             _ LIB11
_ LIB20

Object Start Value ... *
Enter options (above), or '?' (Help) or '.' (Exit): _

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Copy Del Find List Move Ren Canc

```

The **Library** column lists all libraries that contain the specified debug environment(s). For a list of line commands available on the **Library Selection** screen, see [Selection Lists for Debug Environments](#).

#### ▶ To invoke a list of libraries for error messages

##### ■ Choose either of the following methods:

1. In a **List** menu, in the **No. From** field, enter an error message number (in the example below, 1) or specify a range of numbers by entering a start number in the **No. From** field and an end number in the **No. To** field.

In the **Library** field, enter a range of library names (see also *Specifying a Range of Names*).

A **Library Help** window similar to the example below appears:

```

14:02:47          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG          - List Error Message Texts -

          +-----+
          ! --- Library Help --- !
Code  Fun !      - Source -      !
          !                                !
      A  Lis !    1  system messages !
      E  Lis !    2  LIB1             !
      S  Lis !    3  LIB2             !
      ?  He1 !    4  LIB3             !
      .  Exi !    5  LIB4             !
          !    6  LIB5             !
Code ..... A    !    7  LIB6             !
          !    8  LIB7             !
Error  No. From .. 1__ !
Source Library ... LIB*_ ! 0__ FNR .. 32__
          ! Enter selection or !
          ! '.' to Exit: 8_ !
          +-----+

Command ===>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Copy Del Find List Move Ren Fsec Fnat
    
```

From the window, select a library by entering the number that corresponds to the required library as shown in the example above where 8 was entered to select LIB7. If you have specified the correct FNAT system file, you can also select system error messages. If required, press enter to scroll to the end of the list.

2. In a **Find** menu, in the **Number** field enter an error message number, and, in the **Library** field, enter a range of library names (see also *Specifying a Range of Names*).

A **Find Selection** screen similar to the example below appears for the specified range of libraries (here: LIB\*):

```

19:09:34          ***** NATURAL SYSMAN UTILITY *****          2005-08-10
User SAG                - Find Selection -

FIND ERR 1 TYPE A FROM LIB* LANG * WHERE DBID 10 FNR 32

C  Library      Error Message Text(s) for Error number: 1          Type
-  - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
_  LIB1         Short message of Error 1: wrong input value.          S
_  LIB2         Short message of Error 1: incorrect syntax.            S
_  LIB3         Short message of Error 1: undefined variable.          S
_  LIB4         Short message of Error 1: timeout error.                S
_  LIB5         Short message of Error 1: initialization failed.
S/E
_  LIB6         Short message of Error 1: invalid format.
S/E
_  LIB7         Short message of Error 1: wrong password.
S/E
_  LIB8         Short message of Error 1: input missing.
S/E
_  LIB9         Short message of Error 1: undefined keyword.
S/E
_  LIB10        Short message of Error 1: invalid command.                S

Enter options (above), or '?' (Help) or '.' (Exit): __

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Menu  Exit  Copy  Del          List  Move  Ren          Canc

```

The **Library** column lists all libraries that contain the specified error message(s) (here: 1). The **Type** column indicates whether the short (S) and/or the extended (E) error message is available for the specified message.

For a list of line commands available on the **Find Selection** screen, see [Selection Lists for Error Messages](#).

## Function Processing and Reporting

---

There are two types of processing that can be used when performing a SYSMAIN function: selective processing and automated processing.

This section describes the two types of function processing and how to obtain a status report about function processing:

- [Automated Processing](#)
- [Selective Processing](#)
- [Using a Selection List](#)
- [Online Report Mode](#)

### Automated Processing

Automated processing is the default type of processing when operating online in command mode. In batch mode, automated processing is the standard type of processing.

Automated processing is an online or batch facility which processes objects without displaying an intervening selection list. This requires little or no terminal I/O after a function has been selected.

Using automated processing online, the status of individual objects is not displayed, but an appropriate message is displayed upon completion of processing. If any of the following Natural system error messages is displayed, some or all of the specified objects were not processed:

```
4867:Nothing found for this request.  
4810:All data rejected by these selection criteria.  
4893:Normal completion, but some data were rejected.
```

The extended (long) message text of error NAT4810 lists reasons why an object may not have been processed. Reasons for an object not to be processed are also discussed in the section [Processing Status and Error Notification](#).

Batch mode or selective processing should be used if it is necessary to see the status of each object after it is processed.

If required, when operating online, you can obtain a batch report with the processing status of all objects as described in [Online Report Mode](#).

## Selective Processing

Selective processing is the default type of processing when operating in menu mode.

Selective processing is an online facility which displays a selection list of all objects that meet the specified selection criteria. A selection list provides the following options:

- Select single or multiple objects for further processing.
- Perform an additional function on an object (for example, displaying the source code) before processing it (see: [Using a Selection List](#)).
- Obtain individual messages on the processing status of each object (for example, *Moved* or *Copied*).
- View the direct command that corresponds to the input values entered in the SYSMAIN menu (see the [example of a selection screen](#) below). This may be of help when using direct commands instead of menu functions.

## Using a Selection List

This section provides instructions for invoking a selection list to further process single or multiple objects. In addition, this section describes how to shorten a selection when using the list or find function. This option is useful with large libraries in order to restrict the number of objects being displayed.

### ▶ To use a selection list online

- 1 In menu mode: In a SYSMAIN menu, in the **Sel. List** field, enter a Y (this is the default setting) to activate selective processing. Input of an N deactivates selective processing.

Or:

In command mode: Include the keyword `HELP` in the *with-clause* of the direct command or enter a question mark (?) immediately after the object name; see also the syntax diagrams in the object-type specific sections of the *SYSMAIN Utility* documentation.

When selective processing has been activated, a **Selection** screen similar to the example below appears:

```

11:05:22          ***** NATURAL SYSMAN UTILITY *****          2005-08-10
User SAG          -      Copy      Selection      -

COPY ALL * WITH XREF N FROM OLDLIB WHERE DBID 10 FNR 50 TO NEWLIB WHERE DBID
10 FNR 60

C  Name      Type  S/C  Message      C  Name      Type  S/C  Message
-  - - - - -  - - - - -  - - - - -  - - - - -  - - - - -  - - - - -  - - - - -  - - - - -
_  AA        Copycd S/C
_  G0000002 Global S/C
_  L0000001 Local  S/C
_  L0000003 Local  S
_  M0000002 Map    S/C
_  P0000001 Progrm S
_  P0000003 Progrm S/C
_  P0000005 Progrm S
_  P0000007 Progrm S/C
_  S0000002 Subpgm S/C
_  T0000001 Text   S

                                Listed Library: OLDLIB

          Enter options (above), or '?' (Help) or '.' (Exit) _
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
          Help Menu Exit Copy Del Find List Move Ren          Canc

```

The list contains the names and the types of the specified objects and indicates whether saved objects (S) and/or cataloged (C) objects exist.

The lines above the selection list (highlighted in the example above) displays the SYSMAN direct command that corresponds to the input values you entered in the menu fields.

This command corresponds to the command syntax that applies when you work in command mode, although some keywords are optional, as shown in the syntax diagrams in the object-type specific sections of the SYSMAN documentation.

- 2 Select one or more objects for further processing: In the **C** (command) column next to the object(s) desired, enter one of the line commands described in the object-type specific sections of the *SYSMAN Utility* documentation.
- 3 Press ENTER to perform one or more line commands.

Line commands are executed in alphabetical order of the specified object names whereby commands that perform a SYSMAN maintenance function (for example, delete) are always executed last.

If *no* SYSMAIN maintenance function (for example, displaying source code) was performed on any object, you can again enter a line command for any object(s). However, once a SYSMAIN maintenance function has been performed on any object(s), the fields of the **C** column are no longer available for input.

If a line command has been executed, the status of the object(s) processed is displayed in the **Message** column as shown in the example below:

```

11:05:22          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG          -      Copy      Selection      -

COPY ALL * WITH XREF N FROM OLDLIB WHERE DBID 10 FNR 50 TO NEWLIB WHERE DBID
10 FNR 60

C  Name      Type  S/C  Message      C  Name      Type  S/C  Message
-  - - - - -  - - - - -  - - - - -  - - - - -  -  - - - - -  - - - - -  - - - - -
_  AA        Copycd S/C  Copied      _  G0000001  Global S    Copied
_  G0000002  Global S/C  Copied      _  G0000003  Global S/C  Copied
_  L0000001  Local  S/C  Copied      _  L0000002  Local  S/C  Copied
_  L0000003  Local  S    Replaced    _  M0000001  Map    S    Replaced
_  M0000002  Map    S/C  Replaced    _  M0000003  Map    S/C  Replaced
_  P0000001  Progrm S
_  P0000003  Progrm S/C
_  P0000005  Progrm S
_  P0000007  Progrm S/C
_  S0000002  Subpgm S/C
_  T0000001  Text    S
_                                     _  T0000002  Text

Listed Library: OLDLIB

Enter options (above), or '?' (Help) or '.' (Exit) _
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit Copy Del Find List Move Ren Canc

```

For a list of possible status messages, see [Status Messages](#) in the section *Processing Status and Error Notification*.

#### ► To shorten a selection list

- On a **List Selection** or **Find Selection** screen, in the **Reposition to** field, enter the name of an object or specify a range of names (see [Specifying a Range of Names](#)) and press ENTER.

If you have specified a name, the list now starts from the specified name. If you have specified a range of names, the list now displays only objects within that range.

The **Reposition to** option is only valid in online mode. It is not a selection criterion for the list function.

## Online Report Mode

Online report mode can be used to obtain a SYSMAIN batch report online instead of a selection list. An online batch report lists all objects that were affected by a SYSMAIN function and indicates the action performed on each of these objects.

### ▶ To use a batch report online

- 1 In a SYSMAIN menu, in the **Sel. List** (Selection List) field, enter an N to activate automated processing.
- 2 In the Command line, enter one of the following direct commands:

```
BATCH
```

or

```
BAT
```

A message appears confirming that batch mode has been activated.

- 3 If required, you can obtain a hardcopy of the report by entering the following terminal command:

```
%H
```

(See also *%H - Hardcopy Output* in the *Terminal Commands* documentation.)

- 4 Execute a SYSMAIN function. SYSMAIN now processes the function as if in batch mode. Hence, only the result of each action is present in a report-type format as shown in the example below:

```
10:50:30          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG          - Copy   Function -

COPY ALL * WITH XREF N FROM TESTLIB1 WHERE DBID 10 FNR 30 TO TESTLIB2 WHERE
DBID 10 FNR 40

Saved   Progrm TEST1   has now been Copied
Saved   Progrm TEST2   has now been Copied
Saved   Progrm TEST3   has now been Copied
Stowed  Progrm TEST4   has now been Copied
Stowed  Progrm TEST5   has now been Copied
Stowed  Progrm TEST6   has now been Copied
Stowed  Progrm TEST7   has now been Copied
Saved   Record TEST8   has now been Copied
Saved   Text   TEST9   has now been Copied
```

```

Cataloged Progrm TEST10  has now been Copied
Saved      Progrm TEST11  has now been Copied
Stowed     Progrm TEST12  has now been Copied
MORE
Stowed     Progrm TEST13  has now been Copied
Cataloged Progrm TEST14  has now been Copied
Stowed     Progrm TEST15  has now been Copied

```

- 5 If required, you can interrupt function processing by entering a system command or terminal command at a MORE prompt. Otherwise, press ENTER until you have reached the end of the list and return to the SYSMAIN menu where a message confirms successful execution of the function.

## SYSMAIN Online Help

The SYSMAIN online help facility provides information on all functions provided by SYSMAIN including detailed explanations of the direct command syntax and examples of direct commands.

In addition to the help facility, SYSMAIN provides individual information on any input field available on any SYSMAIN screen.

### ▶ To invoke SYSMAIN help topics

- 1 In the **SYSMAIN Utility Main Menu**, position the cursor in the **Object Code** or the **Function Code** field and press PF1 (Help) or enter a question mark (?).

The **Help Menu** of the SYSMAIN utility appears similar to the example below with a list of help topics provided:

```

15:54:28          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG                - Help Menu -                               Help Name SHT-0001

      Code  Topic
      A    SYSMAIN General Overview
      C    Command Mode
      E    Environment Definition
      F    Functions / Commands
      S    Security Environment
      .    Exit

```

```
Code ... _  
Select a function code.  
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---  
Menu Exit                                         Canc
```

- 2 In the **Code** field, enter the one-letter code that corresponds to the help topic desired.
- 3 Press ENTER.

A result screen appears with information on the help topic selected, or another menu is invoked with further help topics that help narrow down your search.

▶ **To invoke help on an input field**

- Position the cursor in the field in question and press PF1 (Help) or enter a question mark (?) and press ENTER.

A window appears with field-specific instructions and, where applicable, a list of valid input values.

# 89

## Processing Programming Objects

---

▪ Fields in Programming Objects Menus .....	522
▪ Using Profile Parameter RECAT .....	525
▪ Selection Lists for Programming Objects .....	525
▪ XRef Considerations .....	529
▪ Specifying Additional Criteria .....	531
▪ Direct Command Syntax for Programming Objects .....	532

All SYSMAIN functions can be performed on programming objects. Programming objects that can be maintained with SYSMAIN include the following types of Natural object: program, subprogram, subroutine, copycode, help routine, map, local data area, global data area, parameter data area, class, text, recording, Natural command processor, dialog, function, ISPF macro, report, adapter and resource.

Programming objects are stored in the system files according to the name of the library in which they are contained: if the library begins with SYS (except for the library SYSTEM), objects are stored in the FNAT system file. In all other libraries, the objects are stored in the FUSER system file.

This section describes menu functions and selection list options provided to perform a SYSMAIN function on programming objects and the syntax that applies when using direct commands.

## Fields in Programming Objects Menus

The **Programming Objects** menus contain all SYSMAIN functions required for the processing of programming objects. The fields that can be provided in a menu are described in the following table:

Field	Explanation	
Recat: ON	Indicates that the profile parameter RECAT has been set to ON: see <a href="#">Using Profile Parameter RECAT</a> .	
Code	Specifies whether a saved (source) object and/or a cataloged object is to be processed:	
	A	Any object that exists as a saved object and/or a cataloged object is processed.
	C	Any object that exists as a cataloged object is processed.
	S	Any object that exists as a saved object is processed.
W	Only an object that exists as both a saved object <i>and</i> a cataloged object is processed. The exceptions to this are copycode, text and recording which cannot be cataloged. However, they are included in processing when this option is specified.	
Sel. List	Specifies whether selective processing or automated processing is used:	
	Y	Yes. Selective processing is activated. A selection list is displayed when processing objects. This is the default setting. For the columns, fields and line commands available on a selection list, see <a href="#">Selection Lists for Programming Objects</a> .

Field	Explanation								
	N No. Selective processing is deactivated.								
<b>Name</b>	The name of the object to be processed or a range of names: see also <a href="#">Specifying a Range of Names</a> .  The default setting is an asterisk (*) which means that all names are selected.								
<b>New Name</b>	The name to be given to an object when it is renamed with the rename function.								
<b>Type</b>	The code that corresponds to the object type(s) to be processed such as P for program and M for map: see <a href="#">TYPE Specification - Programming Objects</a> .  You can enter one or more codes in any sequence. For example, if you enter PAM, programs, parameter data areas and maps are processed.								
<b>Set Number</b>	The number of the retained set created with the Predict XRef save set option of the LIST XREF command. You can apply all SYSMAIN processing functions to the objects included in this set.  If any valid number is specified, SYSMAIN assumes a Predict set. If no number is specified, normal object processing is assumed.  You can specify a library and a user ID for the Predict set by using the fields <b>Set Library</b> and <b>Set User</b> in the <b>Additional Criteria</b> window described in <a href="#">Specifying Additional Criteria</a> .								
<b>XREF</b>	Indicates whether XRef (cross-reference) data stored on Predict system files is to be processed: <table border="1" data-bbox="412 1045 1474 1585"> <tbody> <tr> <td>N</td> <td>No. XRef data is not processed, except when using the delete function. If a cataloged object is deleted or replaced, SYSMAIN always deletes any existing XRef data for this object. N is the default setting.</td> </tr> <tr> <td>Y</td> <td>Yes. All XRef data is processed.</td> </tr> <tr> <td>S</td> <td>Special. A specified object is processed regardless of whether it has cross-reference data or not. Any existing XRef data is processed.</td> </tr> <tr> <td>F</td> <td>Force. All XRef data is processed and the object must be documented in Predict.</td> </tr> </tbody> </table> For further details, see <a href="#">XRef Considerations</a> .	N	No. XRef data is not processed, except when using the delete function. If a cataloged object is deleted or replaced, SYSMAIN always deletes any existing XRef data for this object. N is the default setting.	Y	Yes. All XRef data is processed.	S	Special. A specified object is processed regardless of whether it has cross-reference data or not. Any existing XRef data is processed.	F	Force. All XRef data is processed and the object must be documented in Predict.
N	No. XRef data is not processed, except when using the delete function. If a cataloged object is deleted or replaced, SYSMAIN always deletes any existing XRef data for this object. N is the default setting.								
Y	Yes. All XRef data is processed.								
S	Special. A specified object is processed regardless of whether it has cross-reference data or not. Any existing XRef data is processed.								
F	Force. All XRef data is processed and the object must be documented in Predict.								
<b>Library</b>	The name of a source or a target library.  The source library contains the object to be processed. The target library is the library to which the object is to be copied or moved, or where the object is renamed.  See also <a href="#">Listing and Selecting Libraries</a> .								

Field	Explanation								
<b>Database</b>	<p>The database ID (DBID) of a source or a target database.</p> <p>The source database contains the library and system file where the object to be processed is stored. The target database contains the library and system file to which the object is to be copied or moved, or where the object is renamed.</p> <p>Valid database IDs are 1 to 65535.</p>								
<b>File</b>	<p>The file number (FNR) of a source or a target system file (FNAT or FUSER).</p> <p>Valid file numbers are 1 to 65535.</p> <p>The source file contains the library where the object to be processed is stored. The target file contains the library to which the object is to be copied or moved, or where the object is renamed.</p>								
<b>Replace</b>	<p>Specifies whether an object is to be replaced when using the move, copy or rename function:</p> <table border="1" data-bbox="316 758 1385 1083"> <tr> <td data-bbox="316 758 675 806"></td> <td data-bbox="675 758 1385 806"></td> </tr> <tr> <td data-bbox="316 806 675 921">Y</td> <td data-bbox="675 806 1385 921">                     Yes.                      An object with the same name which exists in the target environment is replaced.                 </td> </tr> <tr> <td data-bbox="316 921 675 1037">N</td> <td data-bbox="675 921 1385 1037">                     No.                      An object with the same name which exists in the target environment is not replaced. This is the default setting.                 </td> </tr> <tr> <td data-bbox="316 1037 675 1083"></td> <td data-bbox="675 1037 1385 1083"></td> </tr> </table> <p>See also <a href="#">Using the Replace Option</a>.</p>			Y	Yes. An object with the same name which exists in the target environment is replaced.	N	No. An object with the same name which exists in the target environment is not replaced. This is the default setting.		
Y	Yes. An object with the same name which exists in the target environment is replaced.								
N	No. An object with the same name which exists in the target environment is not replaced. This is the default setting.								
<b>Criteria</b>	<p>Invokes the <b>Additional Criteria</b> window where you can specify additional object selection criteria. Possible values are:</p> <table border="1" data-bbox="316 1220 1385 1503"> <tr> <td data-bbox="316 1220 732 1268"></td> <td data-bbox="732 1220 1385 1268"></td> </tr> <tr> <td data-bbox="316 1268 732 1383">N</td> <td data-bbox="732 1268 1385 1383">                     No.                      The <b>Additional Criteria</b> window is not invoked. This is the default setting.                 </td> </tr> <tr> <td data-bbox="316 1383 732 1503">Y</td> <td data-bbox="732 1383 1385 1503">                     Yes.                      The <b>Additional Criteria</b> window is invoked.                 </td> </tr> <tr> <td data-bbox="316 1503 732 1551"></td> <td data-bbox="732 1503 1385 1551"></td> </tr> </table> <p>See also <a href="#">Specifying Additional Criteria</a>.</p>			N	No. The <b>Additional Criteria</b> window is not invoked. This is the default setting.	Y	Yes. The <b>Additional Criteria</b> window is invoked.		
N	No. The <b>Additional Criteria</b> window is not invoked. This is the default setting.								
Y	Yes. The <b>Additional Criteria</b> window is invoked.								

## Using Profile Parameter RECAT

If the profile parameter `RECAT` has been set to `ON` at session start, this is indicated in the **Programming Objects** menus and on the **Selection** screens.

Using `SYSMAIN` functions with `RECAT=ON`, normal dynamic recatalog rules apply as described in *RECAT - Dynamic Recataloging* in the *Parameter Reference* documentation. This means the following:

- If an object exists as both a saved object and a cataloged object, neither the saved object nor the cataloged object can be processed independently.
- If an object only exists as a cataloged object, you cannot perform a `SYSMAIN` function that only processes cataloged objects (error message `Invalid request with dynamic recatalog` appears) or select a cataloged object from a **Selection** screen.

When automated processing is used, any object not satisfying these rules is ignored and processing continues with the next object.

## Selection Lists for Programming Objects

If selective processing has been activated, a selection list of all programming objects that meet the specified selection criteria is displayed on a **Selection** screen.

This section describes the columns and fields contained on a **Selection** screen and the line commands provided to further process a programming object:

- [Columns and Fields](#)
- [Line Commands](#)

### Columns and Fields

The following columns and fields are displayed on a **Selection** screen:

Column/Field	Explanation
C	Input field for line commands (see below).
Name	The name of the programming object that meets the specified selection criteria.
Type	The code that corresponds to the type of object as listed in <i>TYPE Specification - Programming Objects</i> .
S/C	The object that exists for the programming object: a saved/source (S) object and/or a cataloged (C) object.

Column/Field	Explanation
Message	The message that indicates the processing status of a programming object. For possible messages, see <a href="#">Status Messages</a> .
Listed Library	The name of the library that contains the selected programming object(s).

## Line Commands

One of the following line commands can be entered in the **C** (Command) column of a **Selection** screen:

Line Command	Function
A	<p>Process any object listed in the <b>S/C</b> (Saved/Cataloged) column; that is, saved/source (S) objects and/or cataloged (C) objects.</p> <p>This command does not apply to the find or the list function.</p>
B	<p>Delete the cataloged object from the Natural buffer pool. Deletion of the specified object(s) must be confirmed by entering <code>DELETE</code> in a window that appears once you have specified the object(s) and pressed <code>ENTER</code>.</p>
C	<p>Process only the cataloged object, even if there is a corresponding saved object. If C is specified for an object that exists only as a saved object, an error occurs.</p> <p>This command does not apply to the find or the list function.</p>
D	<p>Only applies to a <b>Library Selection</b> screen (see also <a href="#">Listing and Selecting Libraries</a>).</p> <p>Display a short list of the objects contained in the specified library. The information contained in the list (name, type, source/cataloged object) is identical to the information displayed on the <a href="#">Selection screen</a> shown in <i>Selective Processing in Using Menu Functions and Commands</i>.</p>
H	<p>Produce a hardcopy of the saved (source) object.</p> <p>The source code of the specified object is printed and displayed on the screen.</p>
I	<p>Display directory information of an object.</p> <p>This command is similar to the system command <code>LIST DIR object-name</code>; for details on the directory information displayed, refer to <i>Displaying Directory Information</i> in the <i>System Commands</i> documentation.</p>
L	<p>Display the source code of a saved (source) object.</p> <p>This command corresponds to the system command <code>LIST object-name</code>; for the commands that can be executed from a source-code screen, refer to <i>List of Source</i> in the <i>System Commands</i> documentation. Exceptions: The command <code>EXPAND</code> is not available. The command <code>ZOOM</code> cannot be used if the source object to be displayed is contained in a steplib library.</p> <p>On a <b>Library Selection</b> screen (see also <a href="#">Listing and Selecting Libraries</a>):</p>

Line Command	Function
	Display an extended list of the objects contained in the library. In addition to the information displayed by using line command D, the extended list provides information of the object directory: programming mode, Natural version, user ID, saved/cataloged date and time.
R	<p>Display the long name of a Natural object of the type resource.</p> <p>On a <b>Library Selection</b> screen (see also <a href="#">Listing and Selecting Libraries</a>):</p> <p>Verify use of external subroutines: For the library and object range specified, display the objects for which a cataloged object exists and indicate whether an object references an external subroutine.</p> <p>For each object that references an external subroutine, display the name of the external subroutine and the name of the cataloged object that exists for this subroutine if available. For an object of the type subroutine, the name of its equivalent alias long name is displayed.</p>
S	<p>Process only the saved (source) object, even if there is a corresponding cataloged object. If S is specified for an object which exists only as a cataloged object, an error occurs.</p> <p>This command does not apply to the find or the list function.</p> <p>On a <b>Library Selection</b> screen (see also <a href="#">Listing and Selecting Libraries</a>):</p> <p>List all external subroutines contained in the specified library by their alias long names and/or the short names of their equivalent cataloged objects.</p>
X	<p>Only applies if Natural Connection and Entire Connection are installed.</p> <p>Download saved (source) object(s) to a PC: see also <a href="#">Downloading Source Objects to a PC</a>.</p>
Z	Calculate sizes: review the sizes of the saved (source) and cataloged objects, for example, the DATSIZE, ESIZE and MCG size.

## Downloading Source Objects to a PC

The download option only applies if Natural Connection and Entire Connection are installed.

### ▶ To download one or more source objects

- Before you invoke SYSMAIN:
  - Define work files 6 and 7 as PC work files.
  - After session start, activate the PC connection by entering the following terminal command:

```
%+
```

(See also *Enable/Disable Use of Natural Connection* in the *Terminal Commands* documentation).

- Invoke any **Selection** screen.
- Next to the object(s) you want to download, enter the following line command:

```
X
```

and press ENTER.

A **PC Download Options** window similar to the example below appears:

```

13:23:15          ***** NATURAL SYSMAIN UTILITY *****          2005-08-10
User SAG          -      Copy      Selection      -

COPY ALL * WITH XREF N FROM OLDLIB WHERE DBID 10 FNR 32 TO NEWLIB WHERE
DBID 10 FNR 32

C  Name      Type  S/C  Message          C  Name      Type  S/C  Message
-  - - - - -  - - - - -  - - - - -  - - - - -  -  - - - - -  - - - - -  - - - - -
X  SUB1      Subpgm S          X  SUB2      Subpgm S/C
X  SUB3      Subpgm S/C        X  SUB4      Subpgm S
X  SUB5      Subpgm S/C        X  SUB6      Subpgm S/C
+----- PC Download Options -----+ S/C
! Specify the relevant PC options          ! S/C
!                                          ! S
! Drive ..... C                          ! S/C
! Path ..... SOURCES/SUBPROGRAMS         ! S/C
! Extension .. NS*                        ! S
!                                          ! S
!                                          ! S
!                                          ! ed Library: OLDLIB
! Warning: This will overwrite any existing objects with !
!           the same path, name and extension.          ! Exit) _
+-----+ -PF10--PF11--PF12---
      Help  Menu  Exit  Copy  Del  Find  List  Move  Ren          Canc
    
```

In the **Drive** field, enter the name of the PC drive to which you want the object(s) to be downloaded. The default setting is C.

In the **Path** field, enter the name of the PC directory/subdirectory to which you want the object(s) to be downloaded (in the example above, subdirectory *SUBPROGRAMS* in directory *SOURCES*). Enter a slash (/) as the separator between a directory and subdirectory, and between subdirectories. If the specified directory/subdirectory does not exist, you will receive an appropriate error message.

In the **Extension** field, enter the extension of the text file into which the source code of the object is to be loaded. If you specify *NS\**, the asterisk (\*) will be replaced by the specified object type (see *TYPE Specification - Programming Objects*).

For example: a subprogram with the name *SUB1* will be loaded into a file with the name *SUB1.NSN*.

- 4 Press **ENTER** to download the object(s).

The following message appears in the message line: 4824:Requested option(s) processed successfully.

Additionally, if the **Message** column is displayed on the screen, the message *Exported* appears next to the object(s) downloaded.

- 5 If desired, continue downloading other objects from this selection list.

The **PC Download Options** window will not appear again for any further downloads from the current **Selection** screen. This window only appears when the line command *X* is issued for the first time after invoking a **Selection** screen, or after executing the direct command *SET PC*.

The settings in the **PC Download Options** window remain active until you terminate *SYS-MAIN*.

## XRef Considerations

---

All cross-reference (XRef) data stored in the Predict system file can be processed with *SYSMAIN*. The *XREF* option indicates whether *SYSMAIN* should process XRef data. XRef data is always deleted if the delete or replace function is performed on a cataloged object.

If Predict has not been installed, set the *XREF* option to *N* and thus no validation of Predict files is performed. If the *XREF* option is set to *Y* and the *FDIC* file(s) being used are not valid Predict files, an error message is returned.

The rules for setting the *XREF* option are the same as the ones imposed by Natural Security. However, in a non-security environment there are no restrictions.

This section covers the following topics:

- XREF set to N
- XREF set to Y or F
- XREF set to F
- XREF set to S
- XREF Errors

### **XREF set to N**

If the XREF option is set to N, no XRef data is processed, but in situations where a cataloged programming object is deleted or replaced, SYSMAIN deletes the XRef data. The target Predict system file is determined according to the current settings of the source or target FDIC system file. The default is the value assigned to the profile parameter FDIC (see *FDIC - Predict System File* in the *Parameter Reference* documentation) at the start of the Natural session.

### **XREF set to Y or F**

If the XREF option is set to Y or F, the following actions are applied during processing:

- SYSMAIN verifies that XRef data already exists in the Predict system source file.
- If the replace option is active (set to Y) and a programming object is to be deleted from the target environment, XRef data is deleted from the Predict system target file.
- If a programming object is being copied to a new environment, the XRef data of the programming object is copied from the Predict system source file to the Predict system target file. The library name is changed accordingly and in the case of the rename function, the object name is also changed.
- If the move function was requested, the XRef data of the programming object is deleted from the Predict system source file.

### **XREF set to F**

If the XREF option is set to F, SYSMAIN additionally checks that the programming object (program, subroutine, subprogram, map or helproutine only) has a Predict program entry defined on the Predict system target file. If not, processing of the object is terminated.

### **XREF set to S**

If the XREF option is set to S, the special case applies where a range of specified objects is processed with corresponding XRef data regardless of whether all of the objects have cross-reference data or not: the objects that have cross-reference data are processed with their cross-reference data, and the objects that have none are also processed.

## XREF Errors

If any of the following inconsistencies occur during the SYSMAIN processing of XRef data, all processing for the object or function is terminated and an error message is displayed:

- The value of the XREF option in Natural Security is F or Y and you specified a value of Y or N respectively.
- The XREF option is set to F and SYSMAIN finds no documented program entry in Predict for the object being processed.
- An invalid Predict file is specified.
- The value of the XREF option in Natural Security is F or Y and you specified a value of S.

## Specifying Additional Criteria

In addition to the selection criteria specified in the input fields in a **Programming Object** menu, you can select objects by a date/time, user ID and terminal ID that relates to their saving or cataloging.

You can also specify the user ID and the library for a Predict set. This option does not apply to the list and find functions.

For example, you can select only those objects that were cataloged on a specific day between 8:00 and 12:00 by a specific user on a specific terminal, which means that the processing of objects according to the selection criteria is based on all selected criteria as a whole, not on each condition.

### ► To specify additional selection criteria

- 1 In the **Criteria** field of a **Programming Objects** menu, replace N (default) by Y.

The **Additional Criteria** window similar to the example below appears:

```

15:24:55          ***** NATURAL SYSMAIN UTILITY *****          2006-09-15
User MM0          - Copy Programming Objects -
+-----+
!          --- Additional Criteria ---          !
!
! Object Type ..... PN_____          !
! Date/Time From .. 2006-01-01 _____          !
! Date/Time To .... _____          !
! User ID ..... SAG_____          !
! Terminal ID ..... _____          !
! Set Library ..... _____          !
! Set User ..... _____          !

```

```

      !                               !
      Code .. ! Command ==>         !
Object Name .. !                               !
      +-----+-----+-----+-----+ N
Source Library ... SYSTEM__ Database .... 10__ File .. 32__
Target Library ... _____ Database .... 10__ File .. 32__
Options Replace ... N      + Criteria .... Y

Command ==>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10---PF11---PF12---
      Help Menu Exit Copy Del Find List Move Ren Fsec Fdic Fnat

```

Enter the required selection criteria. If one or more object-type codes have been entered in the **Object Type** field in the **Additional Criteria** window, the **Type** field in the **Programming Objects** menu is preset to the same object-type codes.

- When you return to the **Programming Objects** menu, a plus sign (+) in front of the **Criteria** field indicates that additional object selection criteria have been specified; in the example above, user ID SAG.

The plus character (+) is not displayed if only the **Object Type** field contains an entry since this entry already appears in the **Type** field in the **Programming Objects** menu.

## Direct Command Syntax for Programming Objects

This section shows the syntax that applies when performing a SYSMAIN function for programming objects by using direct commands in either online or batch mode. For general instructions on using direct commands, refer to [Executing Commands](#).

For explanations of the keywords and variable values used in the syntax diagrams below, refer to [Keywords and Variables in Direct Commands](#). The symbols in the syntax diagrams correspond to the syntax symbols used for system commands. These symbols are explained in *System Command Syntax* in the *System Commands* documentation.

The syntax of the *where-clause* and the *with-clause* are identical for each command.

This section covers the following topics:

- COPY and MOVE
- DELETE
- FIND, LIST and LISTLIB
- RENAME

- where-clause
- with-clause

## COPY and MOVE

$\left\{ \begin{array}{l} \text{COPY} \\ \text{MOVE} \end{array} \right\}$	$\left[ \left\{ \begin{array}{l} \text{ALL} \\ \text{CATALOGED} \\ \text{SAVED} \\ \text{STOWED} \end{array} \right\} \right]$	<i>name</i> [ $\begin{array}{l} \text{FM} [\text{LIBRARY}] \\ \text{lib-name} \end{array}$ ]	[where-clause]

### Examples:

```
COPY PROG1 FM TESTORD TO ORDERS DBID 1 FNR 6 REP
```

```
C PGM* WITH REP TYPE PNS FM PRODLIB TO TESTLIB
```

```
M PROG1 FM OLDLIB TO NEWLIB
```

```
MOVE STOWED * TO NEWLIB WHERE DBID 100 FNR 160 FMDATE 2007-01-01 FM OLDLIB WITH XREF Y
```

## DELETE

$\text{DELETE}$	$\left[ \left\{ \begin{array}{l} \text{ALL} \\ \text{CATALOGED} \\ \text{SAVED} \\ \text{STOWED} \end{array} \right\} \right]$	<i>name</i> [ $\begin{array}{l} \text{IN} [\text{LIBRARY}] \\ \text{lib-name} \end{array}$ ]	[where-clause] [with-clause]

### Examples:

```
DELETE C M> IN LIB ORDERS
```

```
D * IN TESTLIB DBID 1 FNR 5 NAME SYSNAT
```

```
D SA * IN LIBTEST TYPE GLA
```

```
D * TYPE PM IN TESTORD FMDATE 2007-01-01 TODATE 2007-04-30
```

**FIND, LIST and LISTLIB**

```

{ FIND
  LIST
  LISTLIB } [ { ALL
               CATALOGED
               SAVED
               STOWED } ] name [ IN [LIBRARY]
                               lib-name ] [where-clause] [with-clause]

```



**Note:** The direct command LISTLIB is only available in batch mode and is used to obtain a list of library names.

**Examples:**

```
FIND SAVED MENU IN TESTLIB
```

```
FIND STOWED MAINMENU IN SYS* WHERE DBID 1 FNR 5
```

```
F ALL PROG2 IN PROD* FNR 27 DBID 1
```

```
LIST * IN TESTLIB
```

```
LIST DT* IN TESTLIB
```

```
L SAVED TEST* IN TESTLIB TYPE PNS FNR 6
```

```
L SA TEST* TYPE PM IN TESTLIB FNR 6 DBID 2 FMDATE 2007-01-01
```

```
LISTLIB ALL MENU IN SYS* DBID 10 FNR 44
```

**RENAME**

```

RENAME [ { ALL
          CATALOGED
          SAVED
          STOWED } ] name AS new-name [with-clause]
        [ FM [LIBRARY] lib-name ] [where-clause]
        [ TO [LIBRARY] lib-name [where-clause] ]

```

**Examples:**

```
RENAME PGM1 AS PROG1 FM TESTLIB
```

```
R PGM1 AS PROG1 FM TESTLIB DBID 1 FNR 5 TO PRODLIB DBID 2 FNR 6
```

```
R PGM* TYPE PS RCOP FM TESTLIB TO PRODLIB
```

**where-clause**

```
[WHERE] [DBID dbid] [FNR fnr] [NAME vsam-name] [CIPHER cipher]
    [ { PASSWORD } password ]
    [ { PSW } ]
    [DIC (dbid,fnr,password,cipher)]
    [SEC (dbid,fnr,password,cipher)]
```

**Separators**

Commas must be used as separators between the values following the DIC and SEC keywords, or if a value is missing. For example: DIC (10, ,secret,2a).

If the session parameter ID (see *ID - Input Delimiter Character* in the *Parameter Reference* documentation) has been set to a comma, use a slash (/) as the separator between values.

**with-clause**

```
[WITH] [TYPE type] [FMDATE date-from] [TODATE date-to] [FMTIME time-from]
    [TOTIME time-to] [USER user-id] [TID terminal-id] [XREF xref] [HELP]
    [REPLACE] [RCOP] [EXTEND] [ PROMPT ] [ MON ]
    [NOPROMPT] [ NOMON ]
    [SETUSER set-user] [SETNO set-number] [SETLIBRARY set-library]
```



# 90 Processing Debug Environments

---

- Fields in the Debug Environments Menu ..... 538
- Selection Lists for Debug Environments ..... 540
- Direct Command Syntax for Debug Environments ..... 541

All SYSMAIN functions, except the find function, can be performed on debug environments.

The debug environment specification must always correspond to the database ID (DBID) and file number (FNR) of the relevant FUSER system file.

This section describes menu functions and selection list options provided to perform a SYSMAIN function on debug environments and the syntax that applies when using direct commands.

**Related Topic:**

*Debug Environment Maintenance - Debugger documentation*

## Fields in the Debug Environments Menu

---

The **Debug Environments** menu contains all SYSMAIN functions required for the processing of debug environments. The fields provided in the menu are described in the following table:

Field	Explanation	
Code	Specifies the function to be performed as described in <i>Description of Functions</i> :	
	C	Copy debug environment.
	D	Delete debug environment.
	L	List debug environment.
	M	Move debug environment.
	R	Rename debug environment.
	<b>Note:</b> When a debug environment has been moved or copied from one library to another, the breakpoints and watchpoints must be adapted to the new library. For details, refer to <i>Maintaining Debug Environments in Different Libraries</i> in the <i>Debugger</i> documentation.	
Sel. List	Specifies whether selective processing or automated processing is used:	
	Y	Yes. Selective processing is activated. A selection list is displayed when processing debug environments. This is the default setting. For the columns, fields and line commands available on a selection list, see <i>Selection Lists for Debug Environments</i> .
	N	No. Selective processing is deactivated.

Field	Explanation						
<b>Name</b>	<p>The name of the debug environment to be processed or a range of names: see also <a href="#">Specifying a Range of Names</a>.</p> <p>The default setting is an asterisk (*) which means that all names are selected.</p>						
<b>New Name</b>	The name to be given to a debug environment when it is renamed with the rename function.						
<b>Library</b>	<p>The name of a source or a target library.</p> <p>The source library contains the debug environment to be processed. The target library is the library to which the debug environment is to be copied or moved, or where the debug environment is renamed.</p> <p>See also <a href="#">Listing and Selecting Libraries</a>.</p>						
<b>Database</b>	<p>The database ID (DBID) of a source or a target database.</p> <p>The source database contains the library and system file where the debug environment to be processed is stored. The target database contains the library and system file to which the debug environment is to be copied or moved, or where the debug environment is renamed.</p> <p>Valid database IDs are 1 to 65535.</p>						
<b>File</b>	<p>The file number (FNR) of a source or a target system file (FNAT or FUSER).</p> <p>Valid file numbers are 1 to 65535.</p> <p>The source file contains the library where the debug environment to be processed is stored. The target file contains the library to which the debug environment is to be copied or moved, or where the debug environment is renamed.</p>						
<b>Replace</b>	<p>Specifies whether a debug environment is to be replaced when using the move, copy or rename function:</p> <table border="1"> <tbody> <tr> <td></td> <td></td> </tr> <tr> <td>Y</td> <td>Yes. A debug environment with the same name which exists in the target environment is replaced.</td> </tr> <tr> <td>N</td> <td>No. A debug environment with the same name which exists in the target environment is not replaced. This is the default setting.</td> </tr> </tbody> </table> <p>See also <a href="#">Using the Replace Option</a>.</p>			Y	Yes. A debug environment with the same name which exists in the target environment is replaced.	N	No. A debug environment with the same name which exists in the target environment is not replaced. This is the default setting.
Y	Yes. A debug environment with the same name which exists in the target environment is replaced.						
N	No. A debug environment with the same name which exists in the target environment is not replaced. This is the default setting.						

## Selection Lists for Debug Environments

If selective processing has been activated, a selection list of all debug environments that meet the specified selection criteria is displayed on a **Selection** screen.

This section describes the columns and fields contained on a **Selection** screen and the line commands provided to further process a debug environment:

- [Columns and Fields](#)
- [Line Commands](#)

### Columns and Fields

The following columns and fields are displayed on a **Selection** screen:

Column/Field	Explanation
<b>C</b>	Input field for line commands (see below).
<b>Environment</b>	The name of the debug environment that meets the specified selection criteria.
<b>Message</b>	The message that indicates the processing status of a debug environment. For possible messages, see <a href="#">Status Messages</a> .
<b>Listed Library</b>	The name of the library that contains the selected debug environment(s).

### Line Commands

One of the following line commands can be entered in the **C** (Command) column of a **Selection** screen:

Line Command	Function
A	Process the debug environment.  This line command is not available on the <b>List Selection</b> screen. On a <b>List Selection</b> screen, you can only enter a period (.) to leave the screen.
L	Display a list of the debug environments contained in the specified library.  Only applies to a <b>Library Selection</b> screen (see also <a href="#">Listing and Selecting Libraries</a> ).

## Direct Command Syntax for Debug Environments

This section shows the syntax that applies when performing a SYSMAIN function on debug environments by using direct commands in either online or batch mode. For general instructions on using direct commands, refer to *Executing Commands*.

For explanations of the keywords and variable values used in the syntax diagrams below, refer to *Keywords and Variables in Direct Commands*. The symbols in the syntax diagrams correspond to the syntax symbols used for system commands. These symbols are explained in *System Command Syntax* in the *System Commands* documentation.

The syntax of the *where-clause* and the *with-clause* are identical for each command.

This section covers the following topics:

- COPY and MOVE
- DELETE
- LIST
- RENAME
- where-clause
- with-clause

### COPY and MOVE

```
{ COPY }
{ MOVE }  DEBUG  name  FM [LIBRARY] lib-name [where-clause]
          TO [LIBRARY] lib-name [where-clause] [with-clause]
```

#### Examples:

```
COPY D ENV FM TESTLIB WHERE DBID 1 FNR 5 TO PRODLIB WHERE DBID 2 FNR 5 WITH REP
```

```
C DEBUG ENV FM TESTLIB FNR 6 TO PRODLIB FNR 7 REP
```

```
MOVE DEBUG ENV FM OLDLIB WHERE DBID 1 FNR 5 TO NEWLIB WHERE DBID 2 FNR 5
```

```
M DEBUG ENV FM OLDLIB FNR 6 TO NEWLIB FNR 7 REP
```

**DELETE**

```
DELETE DEBUG name [ IN [LIBRARY] lib-name ] [where-clause] [with-clause]
```

**Examples:**

```
DELETE DEBUG U* IN TESTLIB FNR 150
```

```
D DEBUG TEST* IN TESTLIB IN DBID 177 FNR 205
```

**LIST**

```
LIST DEBUG name [ IN [LIBRARY] lib-name ] [where-clause] [with-clause]
```

**Examples:**

```
LIST DEBUG ENV* IN TESTLIB DBID 1 FNR 5
```

```
L D DT* IN TESTLIB DBID 10
```

**RENAME**

```
RENAME DEBUG name AS new-name [with-clause]  
          IN [LIBRARY] lib-name [where-clause]  
          TO [LIBRARY] lib-name [where-clause]
```

**Examples:**

```
RENAME D OLDENV AS NEWENV IN TESTLIB RCOP
```

```
R DEBUG OLDENV AS NEWENV IN TESTLIB DBID 1 FNR 4 TO PRODLIB DBID 1 FNR 5
```

```
R DEBUG OLDENV AS NEWENV IN TESTLIB FNR 4 TO PRODLIB FNR 5 REPLACE RCOP
```

**where-clause**

```
[WHERE] [DBID dbid] [FNR fnr] [NAME vsam-name ]  
[CIPHER cipher] [ { PASSWORD } password ]  
                  [ PSW ]
```

**with-clause**

```
[WITH] [REPLACE] [RCOP] [ PROMPT ] [ MON ] [HELP]  
[NOPROMPT] [NOMON]
```



# 91 Processing Error Messages

---

▪ Fields in Error Message Menus .....	546
▪ Selection Lists for Error Messages .....	548
▪ Renumbering Error Messages .....	550
▪ Specifying Languages .....	551
▪ Direct Command Syntax for Error Messages .....	551

All SYSMAIN functions can be performed on user-defined and Natural system error messages. Error messages can be moved from one library to another, or the language texts of each error message can be copied, moved or replaced. In addition, it is possible to renumber a single error message or a range of error messages by using the rename function.

Only authorized users can process Natural system error messages if Natural Security is installed.

Error messages are stored in the system files according to their type: Natural system error messages are stored in the FNAT system file and user-defined error messages in the FUSER or FNAT system file.

This section describes menu functions and selection list options provided to perform a SYSMAIN function on error messages and the syntax that applies when using direct commands.

**Related Topic:**

[SYSERR Utility](#) documentation.

## Fields in Error Message Menus

---

The **Error Message Texts** menus contain all SYSMAIN functions required for the processing of error messages. The fields that can be provided in a menu are described in the following table:

Field	Explanation	
Code	Specifies whether a short and/or an extended (long) error message is to be processed. Possible values are:	
	A	Any error message for which a short and/or an extended (long) message exists is processed.
	E	Any error message for which an extended (long) error message exists is processed. Only extended error messages with corresponding short error messages are processed.  An extended error message cannot be transferred to a target environment if there is no corresponding short error message in the target environment.
	S	Any error message for which a short error message exists is processed.
Sel. List	Specifies whether selective processing or automated processing is used:	

Field	Explanation				
	<table border="1"> <tr> <td data-bbox="431 243 699 428">Y</td> <td data-bbox="699 243 1474 428">Yes. Selective processing is activated. A selection list is displayed when processing error messages. This is the default setting. For the columns and line commands available on a selection list, see <a href="#">Selection Lists for Error Messages</a>.</td> </tr> <tr> <td data-bbox="431 428 699 512">N</td> <td data-bbox="699 428 1474 512">No. Selective processing is deactivated.</td> </tr> </table>	Y	Yes. Selective processing is activated. A selection list is displayed when processing error messages. This is the default setting. For the columns and line commands available on a selection list, see <a href="#">Selection Lists for Error Messages</a> .	N	No. Selective processing is deactivated.
Y	Yes. Selective processing is activated. A selection list is displayed when processing error messages. This is the default setting. For the columns and line commands available on a selection list, see <a href="#">Selection Lists for Error Messages</a> .				
N	No. Selective processing is deactivated.				
<b>No. From</b> or <b>Number</b>	The number of the error message to be processed, or the start number of a range of numbers if an end number is entered in the <b>No. To</b> field. See also <a href="#">Renumbering Error Messages</a> .				
<b>No. To</b>	The end number of a range of error messages numbers to be processed if a start number is entered in the <b>No. From</b> field. See also <a href="#">Renumbering Error Messages</a> .				
<b>New From</b>	<p>Applies to the rename function.</p> <p>The new number to be given to an existing error message, or the start number of a range of new numbers to be given to a range of existing error messages. See also <a href="#">Renumbering Error Messages</a>.</p>				
<b>New To</b>	<p>Applies to the rename function.</p> <p>The end number of a range of new error message numbers to be given to a range of existing error message. See also <a href="#">Renumbering Error Messages</a>.</p>				
<b>Library</b>	<p>The name of a source or a target library or a range of names. If you want to process Natural system error messages, leave the <b>Library</b> field empty.</p> <p>The source library contains the error message(s) to be processed. The target library is the library to which the error message is to be copied or moved, or where the error message is renumbered.</p> <p>See also <a href="#">Listing and Selecting Libraries</a>.</p>				
<b>Lang.</b>	<p>The language code(s) in which the error message is available.</p> <p>The languages can be specified using any combination of language codes. For information on which language code is assigned to which language, see <i>Language Code Assignments</i> in *LANGUAGE in the <i>System Variables</i> documentation. For Natural system extended (long) error messages, only language code 1 is available. Enter an asterisk (*) to select error messages in all existing languages.</p> <p>See also <a href="#">Specifying Languages</a>.</p>				
<b>Database</b>	<p>The database ID (DBID) of a source or a target database.</p> <p>The source database contains the library and system file where the error message to be processed is stored. The target database contains the library and system file to which the error message is to be copied or moved, or where the error message is renumbered.</p> <p>Valid database IDs are 1 to 65535.</p>				

Field	Explanation								
<b>File</b>	<p>The file number (FNR) of a source or a target system file.</p> <p>Valid file numbers are 1 to 65535.</p> <p>The source file contains the library where the error message to be processed is stored. The target file contains the library to which the error message is to be copied or moved, or where the error message is renumbered.</p>								
<b>Replace</b>	<p>Specifies whether an error message is to be replaced when using the move, copy or rename function:</p> <table border="1" data-bbox="332 558 1385 877"> <tr> <td data-bbox="332 558 678 604"></td> <td data-bbox="678 558 1385 604"></td> </tr> <tr> <td data-bbox="332 604 678 720">Y</td> <td data-bbox="678 604 1385 720"> <p>Yes. An error message with the same number which exists in the target environment is replaced.</p> </td> </tr> <tr> <td data-bbox="332 720 678 835">N</td> <td data-bbox="678 720 1385 835"> <p>No. An error message with the same number which exists in the target environment is not replaced. This is the default setting.</p> </td> </tr> <tr> <td data-bbox="332 835 678 877"></td> <td data-bbox="678 835 1385 877"></td> </tr> </table> <p>See also <a href="#">Using the Replace Option</a>.</p>			Y	<p>Yes. An error message with the same number which exists in the target environment is replaced.</p>	N	<p>No. An error message with the same number which exists in the target environment is not replaced. This is the default setting.</p>		
Y	<p>Yes. An error message with the same number which exists in the target environment is replaced.</p>								
N	<p>No. An error message with the same number which exists in the target environment is not replaced. This is the default setting.</p>								

## Selection Lists for Error Messages

---

If selective processing has been activated, a selection list of all error messages that meet the specified selection criteria is displayed on a **Selection** screen.

This section describes the columns contained on a **Selection** screen and the line commands provided to further process an error message:

- [Columns](#)
- [Line Commands](#)

## Columns

The following columns are displayed on a **Selection** screen:

Column	Explanation
<b>C</b>	Input field for line commands (see below).
<b>Error</b>	The name of the error message that meets the specified selection criteria.
<b>Error Message Text</b>	The text of the short error message.
<b>Type</b>	The type of error message: S (short) and/or E (extended/long).
<b>Lang</b>	The first language code in sequence or alphabetical order that exists for the error message. If you want to display all languages that exist for an error message, on a <b>Selection</b> screen, enter the line command <b>L</b> or use the list function.
<b>Message</b>	The message that indicates the processing status of an error message.  This column only appears after a line command was executed on an error message.  For possible messages, see <a href="#">Status Messages</a> .

## Line Commands

One of the following line commands can be entered in the **C** (Command) column of a **Selection** screen:

Line Command	Function
A	Process any type of message listed in the <b>Type</b> column. The type can be: short (S), extended/long (E) or short and extended (S/E).
E	Process the extended (long) message if the corresponding short message exists in the target environment.  If E is specified for a message which exists only as a short message, an error is returned.
S	Process all short messages.
L	Review an error message before processing it. The short and/or extended (long) error message is displayed for all existing languages, depending on the function specified previously in the <b>Error Message Texts</b> menu.  This line command can also be used on a <b>Library Selection</b> screen (see also <a href="#">Listing and Selecting Libraries</a> ).

## Renumbering Error Messages

---

You can renumber a single error message or a range of error messages.

### ▶ To renumber a single error message

- 1 In the **No. From** field, enter the number of the error message you want to renumber and, in the **New From** field, enter the new error message number.
- 2 In the **Source Library** field, enter the name of the library that contains the error message to be renumbered.

If you want the renumbered error message to be placed in a different library, enter a name in the **Target Library** field.

### ▶ To renumber a range of error messages

- 1 In the **No. From** field, enter the start number of the range of error messages to be renumbered and, in the **No. To** field, enter the end number.

In the **New From** field, enter the start number of a new range of error messages, and in the **New To** field, enter the end number.

The number of error messages specified by the range in the **No. From** and **No. To** fields of the source library must be equal in number to the range in the **New From** and **New To** fields of the target library. For example, it is not possible to renumber error message numbers 1 - 6 as new error message numbers 7 - 10.

If you want to renumber a range of error messages within a single library, range values must not overlap. For example, it is not possible to renumber error message numbers 1 - 6 as new message numbers 5 - 10.

- 2 In the **Source Library** field, enter the name of the library that contains the error messages to be renumbered.

If you want the renumbered error messages to be placed in a different library, enter a name in the **Target Library** field.

See also the examples of range specifications in [\*RENAME\*](#) in *Direct Command Syntax for Error Messages*.

## Specifying Languages

---

When specifying languages consider the following:

- If the list function is performed and the specified language code does not exist for the error message, SYSMAIN uses the default language specified with the system variable \*LANGUAGE (see \*LANGUAGE in the *System Variables* documentation).
- If an asterisk (\*) is specified for a source error message, all language codes defined for this error message will overwrite any language codes defined for the target error message.

For example: If the source error message exists only in languages 1, 2 and 3, and for the target error message only languages 1, 4 and 6 are defined, after performing a copy function, the resulting target error message exists only in languages 1, 2 and 3.

- If the languages are specified as individual codes, each occurrence of language code is processed individually.

For example: If the source error message contains languages 1, 2 and 3, and the language codes are set to 123, and if the target error message contains languages 1, 4 and 6, and the language codes are also set to 123, after performing a copy function, the resulting target error message contains languages 1, 2, 3, 4 and 6, but only the English target error message (language 1) is overwritten by the English text of the source error message.

- If a single language code is specified for the source error message and multiple language codes are specified for the target error message, after performing the copy function, the resulting target error message is in the first language specified for the target.

## Direct Command Syntax for Error Messages

---

This section shows the syntax that applies when performing a SYSMAIN function on error messages by using direct commands in either online or batch mode. For general instructions on using direct commands, refer to [Executing Commands](#).

For explanations of the keywords and variable values used in the syntax diagrams below, refer to [Keywords and Variables in Direct Commands](#). The symbols in the syntax diagrams correspond to the syntax symbols used for system commands. These symbols are explained in *System Command Syntax* in the *System Commands* documentation.

The syntax of the *where-clause* and the *with-clause* are identical for each command.

 **Important:** For system error messages, specify NATURAL - SYSTEM or NATURAL - SYS as *lib-name*.

This section covers the following topics:

- COPY and MOVE
- DELETE
- FIND
- LIST
- RENAME
- where-clause
- with-clause

### COPY and MOVE

```
{ COPY } ERROR number [THRU number]
{ MOVE } FM [LIBRARY] lib-name [where-clause]
          TO [LIBRARY] lib-name [where-clause] [with-clause]
```

**Examples:**

```
COPY ERROR 1 FM ACCOUNTS TO ACCOUNTS1 REP WITH TYPE A
C ERROR 1 THRU 50 FM ACCT WHERE DBID 1 FNR 10 LANG 123456 TO ACCT WHERE DBID 5 FNR
26 LANG 234567 WITH REP HELP
MOVE E 200 THRU 210 FM ACCT FNR 10 LANG 123 TO ACCT LANG 123 TYPE S
M E 376 TYPE E FM ACCT LANG E TO ACCT LANG G
```

### DELETE

```
DELETE ERROR number [THRU number]
          [ IN [LIBRARY] lib-name ] [where-clause] [with-clause]
```

**Examples:**

```
DELETE ERROR 1 THRU 10 IN LIBRARY ACCT WHERE DBID 1 FNR 2 PSW GUESS CIPH 137561
WITH TYPE E MON HELP
D E 100 IN ACCT
```

**FIND**

```
FIND ERROR number [ IN [LIBRARY] lib-name ] [where-clause] [with-clause]
```

**Examples:**

```
FIND E 4280 IN A* MON
```

```
F ERROR 10 IN LIB ACCT WHERE DBID 1 FNR 3 WITH TYPE E
```

**LIST**

```
LIST ERROR number [THRU number] [ IN [LIBRARY] lib-name ] [where-clause] [with-clause]
```

**Examples:**

```
LIST E 1 THRU 10 IN ACCT
```

```
L ERROR 100 THRU 150 IN LIB ACCT WHERE DBID 12 FNR 5
```

**RENAME**

When renumbering a range of error messages within a single (source) library, the range values must not overlap as demonstrated in [Examples of Invalid Number Ranges](#).

```
RENAME ERROR number [THRU number] AS new-number  
[THRU new-number] [with-clause]  
IN [LIBRARY] lib-name [where-clause]  
TO [LIBRARY] lib-name [where-clause]
```

**Examples:**

```
RENAME ERR 1 AS 101 IN ACCT
```

```
R ERROR 1 THRU 100 AS 101 THRU 200 IN CLAIMS
```

```
R ERROR 101 THRU 200 AS 1 THRU 100 IN CLAIMS
```

```
RENAME ERROR 1 THRU 50 AS 11 THRU 60 WITH TYPE A REP HELP MON RCOP IN LIBRARY ACCT  
WHERE DBID 1 FNR 2 TO LIB ACCOUNT WHERE FNR 3
```

**Examples of Invalid Number Ranges:**

The following examples are *invalid*, because the number ranges overlap:

```
R ERROR 1 THRU 100 AS 51 THRU 150 IN CLAIMS
```

R ERROR 101 THRU 200 AS 51 THRU 150 IN CLAIMS

**where-clause**

```
[WHERE] [DBID dbid] [FNR fnr] [NAME vsam-name] [CIPHER cipher]  
[ { PASSWORD } password ] [LANGUAGE] [SEC  
  PSW ] language (dbid,fnr,password,cipher)]
```

**Separators**

Commas must be used as separators between the values following the SEC keyword, or if a value is missing. For example: SEC (10, , secret, 2a). If the session parameter ID (see *ID - Input Delimiter Character* in the *Parameter Reference* documentation) has been set to a comma, use a slash (/) as the separator between values.

**with-clause**

```
[WITH] [TYPE type] [REPLACE] [RCOP] [ PROMPT ] [ MON ]  
[ NOPROMPT ] [ NOMON ] [HELP]
```

# 92 Processing Profiles

---

- Fields in the Profiles Menu ..... 556
- Selection Lists for Profiles ..... 558
- Direct Command Syntax for Profiles ..... 559

All SYSMAIN functions except the find function can be performed on the following types of profile: device profiles, editor profiles, map profiles and parameter profiles. Device, editor and map profiles are created with the program editor or the map editor. Parameter profiles are created with the SYSPARM utility.

This section describes menu functions and selection list options provided to perform a SYSMAIN function on profiles and the syntax that applies when using direct commands.

**Related Topics:**

*Editors* documentation and [SYSPARM Utility](#) documentation

## Fields in the Profiles Menu

The **Profiles** menu contains all SYSMAIN functions required for the processing of profiles. The fields provided in the menu are described in the following table:

Field	Explanation	
Code	Specifies the function to be performed as described in <a href="#">Description of Functions</a> :	
	C	Copy profile.
	D	Delete profile.
	L	List profile.
	M	Move profile.
R	Rename profile.	
Sel. List	Specifies whether selective processing or automated processing is used:	
	Y	Yes. Selective processing is activated. A selection list is displayed when processing profiles. This is the default setting. For the columns and line commands available on a selection list, see <a href="#">Selection Lists for Profiles</a> .
N	No. Selective processing is deactivated.	
Name	The name of the profile to be processed or a range of names: see also <a href="#">Specifying a Range of Names</a> .	
New Name	The name to be given to a profile when it is renamed with the rename function.	
Type	The type of profile to be processed:	

Field	Explanation					
	D	Only device profiles are processed.				
	E	Only editor profiles are processed.				
	M	Only map profiles are processed.				
	P	Only parameter profiles are processed.				
	* or empty field	All profiles are processed. This is the default setting.				
	Types D, E, M and P can be used in any combination.					
<b>Database</b>	<p>The database ID (DBID) of a source or a target database.</p> <p>The source database contains the system file where the profile to be processed is stored. The target database contains the system file to which the profile is to be copied or moved, or where the profile is renamed.</p> <p>Valid database IDs are 1 to 65535.</p>					
<b>File</b>	<p>The file number (FNR) of a source or a target system file.</p> <p>Valid file numbers are 1 to 65535.</p> <p>The source file contains the profile to be processed. The target file is the file to which the profile is to be copied or moved, or where the profile is renamed.</p>					
<b>Name</b> (below <b>New Name</b> )	<p>Only applies to VSAM files.</p> <p>The DDNAME/FCT entry for the source or target file number.</p>					
<b>Replace</b>	<p>Specifies whether a profile is to be replaced when using the move, copy or rename function:</p> <table border="1" data-bbox="464 1276 1474 1591"> <tbody> <tr> <td data-bbox="464 1318 805 1360">Y</td> <td data-bbox="813 1318 1474 1434">Yes. A profile with the same name which exists in the target environment is replaced.</td> </tr> <tr> <td data-bbox="464 1444 805 1486">N</td> <td data-bbox="813 1444 1474 1539">No. A profile with the same name which exists in the target environment is not replaced. This is the default setting.</td> </tr> </tbody> </table> <p>See also <i>Using the Replace Option</i>.</p>		Y	Yes. A profile with the same name which exists in the target environment is replaced.	N	No. A profile with the same name which exists in the target environment is not replaced. This is the default setting.
Y	Yes. A profile with the same name which exists in the target environment is replaced.					
N	No. A profile with the same name which exists in the target environment is not replaced. This is the default setting.					

## Selection Lists for Profiles

If selective processing has been activated, a selection list of all profiles that meet the specified selection criteria is displayed on a **Selection** screen.

This section describes the columns contained on a **Selection** screen and the line commands provided to further process a profile:

- [Columns](#)
- [Line Commands](#)

### Columns

The following columns are displayed on a **Selection** screen:

Column	Explanation										
C	Input field for line commands (see below).										
Profile	The name of the profile that meets the specified selection criteria.										
Type	The type of profile: <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;"></td> <td></td> </tr> <tr> <td>D</td> <td>Device profile.</td> </tr> <tr> <td>E</td> <td>Editor profile.</td> </tr> <tr> <td>M</td> <td>Map profile.</td> </tr> <tr> <td>P</td> <td>Parameter profile.</td> </tr> </table>			D	Device profile.	E	Editor profile.	M	Map profile.	P	Parameter profile.
D	Device profile.										
E	Editor profile.										
M	Map profile.										
P	Parameter profile.										
Message	The message which indicates the current processing status of a profile. For possible messages, see <a href="#">Status Messages</a> .										

### Line Commands

The line commands that can be entered in the **C** (Command) column of a **Selection** screen are listed in the following table. Line commands E, M, D and P can be entered in any combination.

Line Command	Function
A	Process device profiles, editor profiles, map profiles and/or parameter profiles.
D	Process only device profiles, even if there are editor, map and parameter profiles listed as well. If D is specified for a profile which does not exist as a device profile, an error occurs.
E	Process only editor profiles, even if there are device, map and parameter profiles listed as well. If E is specified for a profile which does not exist as an editor profile, an error is returned.

Line Command	Function
M	Process only map profiles, even if there are device, editor and parameter profiles listed as well. If M is specified for a profile which does not exist as a map profile, an error is returned.
L	Display the contents of a parameter profile.
P	Process only parameter profiles, even if there are device, editor and map profiles listed as well. If P is specified for a profile which does not exist as a parameter profile, an error is returned.

## Direct Command Syntax for Profiles

This section shows the syntax that applies when performing a SYSMAIN function on profiles by using direct commands in either online or batch mode. For general instructions on using direct commands, refer to [Executing Commands](#).

For explanations of the keywords and variable values used in the syntax diagrams below, refer to [Keywords and Variables in Direct Commands](#). The symbols in the syntax diagrams correspond to the syntax symbols used for system commands. These symbols are explained in *System Command Syntax* in the *System Commands* documentation.

The syntax of the *where-clause* and the *with-clause* are identical for each command.

This section covers the following topics:

- COPY and MOVE
- DELETE
- LIST
- RENAME
- *where-clause*
- *with-clause*

### COPY and MOVE

```
{ COPY } PROFILE name [with-clause] FM [where-clause] TO [where-clause]
```

```
{ MOVE } PROFILE name [with-clause] FM [where-clause] TO [where-clause]
```

**Examples:**

```
COPY PRO USER1 TYPE E FM DBID 1 FNR 5 TO DBID 2 FNR 5
```

```
C PROFILE USER TYPE MED REP FM FNR 6 TO FNR 7
```

```
COPY PRO USER1 TYPE P FM DBID 10 FNR 44 TO DBID 3 FNR 7
```

```
MOVE PRO USER1 TYPE E FM DBID 1 FNR 5 TO DBID 2 FNR 5
```

```
M PROFILE USER1 TYPE MED REP FM FNR 6 TO FNR 7
```

**DELETE**

```
DELETE PROFILE name [ IN [where-clause] ] [with-clause]
```

**Examples:**

```
DELETE PRO U* TYPE DM
```

```
D PRO TEST* IN DBID 177 FNR 205
```

**LIST**

```
LIST PROFILE name [ IN [where-clause] ] [with-clause]
```

**Examples:**

```
LIST PRO USER* IN DBID 1 FNR 5
```

```
L PRO DT* TYPE E
```

**RENAME**

```
RENAME PROFILE name AS new-name  
                IN [where-clause]  
                TO [where-clause] [with-clause]
```

**Examples:**

```
RENAME PROFILE USER1 AS USER2 RCOP
```

```
R PRO USER1 AS USER2 DBID 1 FNR 4 TO DBID 1 FNR 5
```

```
R PRO USER1 AS NEWUSER IN FNR 4 TO FNR 5 REPLACE RCOP
```

**where-clause**

```
[WHERE] [DBID dbid] [FNR fnr] [NAME vsam-name]  
[CIPHER cipher] [ { PASSWORD } password ]  
                  [ PSW ]
```

**with-clause**

```
[WITH] [TYPE type] [REPLACE] [RCOP] [ PROMPT ] [ MON ] [HELP]  
                  [ NOPROMPT ] [ NOMON ]
```



# 93 Processing Rules

---

- Fields in the Rules Menu ..... 564
- Selection Lists for Rules ..... 565
- Direct Command Syntax for Rules ..... 567

All SYSMAIN functions except the find function can be performed on automatic and free rules.

If Predict is installed, we recommend that you use Predict instead of SYSMAIN for the processing of rules; Natural does not process all information on rules (for example, format and verification type) provided by Predict.

The rule environment specification must always correspond to the database ID (DBID) and file number (FNR) of the relevant FDIC system file.

This section describes menu functions and selection list options provided to perform a SYSMAIN function on rules and the syntax that applies when using direct commands.

## Fields in the Rules Menu

The **Rules** menu contains all SYSMAIN functions required for the processing of rules. The fields provided in the menu are described in the following table:

Field	Explanation	
<b>Code</b>	Specifies the function to be performed as described in <i>Description of Functions</i> :	
	C	Copy rule.
	D	Delete rule.
	L	List rule.
	M	Move rule.
	R	Rename rule.
<b>Sel. List</b>	Specifies whether selective processing or automated processing is used:	
	Y	Yes. Selective processing is activated. A selection list is displayed when processing rules. This is the default setting. For the columns and line commands available on a selection list, see <i>Selection Lists for Rules</i> .
	N	No. Selective processing is deactivated.
<b>Name</b>	The name of the rule to be processed or a range of names: see also <i>Specifying a Range of Names</i> .	
<b>New Name</b>	The name to be given to a rule when it is renamed with the rename function.	
<b>Type</b>	The type of rule to be processed:	

Field	Explanation	
	A	Only automatic rules are processed.
	F	Only free rules are processed.
	AF * or empty field	All rules are processed: automatic and/or free rules. This is the default setting.
<b>Database</b>	The database ID (DBID) of a source or a target database.  The source database contains the FDIC system file where the rule to be processed is stored. The target database contains the FDIC system file to which the rule is to be copied or moved, or where the rule is renamed.  Valid database IDs are 1 to 65535.	
<b>File</b>	The file number (FNR) of a source or a target FDIC system file.  Valid file numbers are 1 to 65535.  The source file contains the rule to be processed. The target file is the file to which the rule is to be copied or moved, or where the rule is renamed.	
<b>Name</b> (next to <b>File</b> )	Only applies to VSAM files.  The DDNAME/FCT entry for the source or target file number.	
<b>Replace</b>	Specifies whether a rule is to be replaced when using the move, copy or rename function:	
	Y	Yes. A rule with the same name which exists in the target environment is replaced.
	N	No. A rule with the same name which exists in the target environment is not replaced. This is the default setting.
	See also <a href="#">Using the Replace Option</a> .	

## Selection Lists for Rules

If selective processing has been activated, a selection list of all rules that meet the specified selection criteria is displayed on a **Selection** screen.

This section describes the columns contained on a **Selection** screen and the line commands provided to further process a rule:

- [Columns](#)

- [Line Commands](#)

## Columns

The following columns are displayed on a **Selection** screen:

Column	Explanation
<b>C</b>	Input field for line commands (see below).
<b>Rule Name</b>	The name of the rule that meets the specified selection criteria.
<b>Type</b>	The type of rule: Free or Auto (Automatic) rule.
<b>Ver. Type</b>	The verification type: an attribute of the rule which is directly taken from the verification as defined in Predict. A value of <i>Unknown</i> indicates that there is no corresponding verification information in Predict available. For valid values, see the <i>Predict</i> documentation.
<b>Format</b>	The format type: an attribute of the rule which is directly taken from the verification as defined in Predict. A value of <i>Unknown</i> indicates that there is no corresponding verification information in Predict available. For valid values, see the relevant <i>Predict</i> documentation.
<b>Message</b>	The message which indicates the current processing status of a rule. For possible messages, see <a href="#">Status Messages</a> .

## Line Commands

One of the following line commands can be entered in the **C** (Command) column of a **Selection** screen:

Line Command	Function
A	Process the specified rule.
F	Only applies to automatic rules.  All DDMs and fields which have the specified rule assigned are displayed.
H	Produce a hardcopy of the saved (source) object of a rule.  The source code of the specified rule is printed and listed on the screen.
I	Display directory information of the rule.  This command is similar to the system command <code>LIST DIR</code> (see <i>Displaying Directory Information</i> in the <i>System Commands</i> documentation).
L	Display the source code of a rule.

## Direct Command Syntax for Rules

This section shows the syntax that applies when performing a SYSMAIN function on rules by using direct commands in either online or batch mode. For general instructions on using direct commands, refer to *Executing Commands*.

For explanations of the keywords and variable values used in the syntax diagrams below, refer to *Keywords and Variables in Direct Commands*. The symbols in the syntax diagrams correspond to the syntax symbols used for system commands. These symbols are explained in *System Command Syntax* in the *System Commands* documentation.

The syntax of the *where-clause* and the *with-clause* are identical for each command.

This section covers the following topics:

- COPY and MOVE
- DELETE
- LIST
- RENAME
- where-clause
- with-clause

### COPY and MOVE

```
{ COPY } RULE name [ FM where-clause ] TO where-clause [with-clause]
{ MOVE }
```

#### Examples:

```
C RULE TESTRULE FM FNR 20 TO FNR 24 REPLACE
```

```
COPY R C< FM FNR 20 TO FNR 24
```

```
M RULE TESTRULE FM FNR 20 TO FNR 24 REPLACE
```

```
MOVE RULE C< FM FNR 20 TO FNR 24
```

**DELETE**

```
DELETE RULE name [with-clause] [ IN where-clause ]
```

**Example:**

```
D RULE DEMORULE IN DBID 12 FNR 27
```

**LIST**

```
LIST RULE name [ IN where-clause ] [with-clause]
```

**Example:**

```
L RULE * DBID 1 FNR 5
```

**RENAME**

```
RENAME RULE name AS new-name [ FM where-clause ] [ TO where-clause ] [with-clause]
```

**Examples:**

```
RENAME R OLDRULE AS NEWRULE FM DBID 1 FNR 4 TO DBID 1 FNR 5
```

```
R RULE OLDRULE AS NEWRULE FM FNR 4 TO FNR 5 REPLACE RCOP
```

**where-clause**

```
[WHERE] [DBID dbid] [FNR fnr] [NAME vsam-name] [CIPHER cipher]
  [ { PASSWORD } password ]
  [ { PSW } ]
  [DIC (dbid,fnr,password,cipher)]
  [SEC (dbid,fnr,password,cipher)]
```

**Separators**

Commas must be used as separators between the values following the DIC and SEC keywords, or if a value is missing. For example: DIC (10, ,secret,2a). If the session parameter ID (see *ID - Input Delimiter Character* in the *Parameter Reference* documentation) has been set to a comma, use a slash (/) as the separator between values.

**with-clause**

```
[WITH] [TYPE type] [REPLACE] [RCOP] [ PROMPT ] [ MON ] [HELP]  
[NOPROMPT] [NOMON]
```



# 94 Processing DL/I Subfiles

---

- Fields in the DL/I Subfiles Menu ..... 572
- Selection Lists for DL/I Subfiles ..... 573
- Direct Command Syntax for DL/I Subfiles ..... 574

All SYSMAIN functions except the find and rename functions can be performed on DL/I subfiles if Natural for DL/I is installed.

DL/I subfiles include the following:

- NDBs = DL/I DBDs (Database Descriptions) defined to Natural.
- NSBs = DL/I PSBs (Program Specification Blocks) defined to Natural.
- UDFs = DL/I User-Defined Fields defined to Natural.

For more information on NDBs, NSBs, DBDs, PSBs and UDFs, see the *Natural for DL/I* documentation.

This section describes menu functions and selection list options provided to perform a SYSMAIN function on DL/I subfiles and the syntax that applies when using direct commands.

## Fields in the DL/I Subfiles Menu

The **DL/I Subfiles** menu contains all SYSMAIN functions required for the processing of DL/I subfiles. The fields provided in the menu are described in the following table:

Field	Explanation	
Code	Specifies the function to be performed as described in <i>Description of Functions</i> :	
	C	Copy subfile.
	D	Delete subfile.
	L	List subfile.
M	Move subfile.	
Sel. List	Specifies whether selective processing or automated processing is used:	
	Y	Yes. Selective processing is activated. A selection list is displayed when processing DL/I subfiles. This is the default setting. For the columns, fields and line commands available on a selection list, see <i>Selection Lists for DL/I Subfiles</i> .
N	No. Selective processing is deactivated.	
Name	The name of the DL/I subfile to be processed or a range of names: see also <i>Specifying a Range of Names</i> .	
Type	The type of DL/I subfile to be processed. You must specify either of the following types:	

Field	Explanation
	D Only NDBs and UDFs are processed. This is the default setting.
	P Only NSBs are processed.
<b>Database</b>	The database ID (DBID) of a source or a target database.  The source database contains the FDIC system file where the DL/I subfile to be processed is stored. The target database contains the FDIC system file to which the DL/I subfile is to be copied or moved.  Valid database IDs are 1 to 65535.
<b>File</b>	The file number (FNR) of a source or a target FDIC system file.  Valid file numbers are 1 to 65535.  The source file contains the DL/I subfile to be processed. The target file is the file to which the DL/I subfile is to be copied or moved.
<b>Name</b> (next to <b>File</b> )	Only applies to VSAM files.  The DDNAME/FCT entry for the source or target file number.
<b>Replace</b>	Specifies whether a DL/I subfile is to be replaced when using the move or copy function:  Y Yes. A DL/I subfile with the same name which exists in the target environment is replaced.  N No. A DL/I subfile with the same name which exists in the target environment is not replaced. This is the default setting.  See also <a href="#">Using the Replace Option</a> .

## Selection Lists for DL/I Subfiles

If selective processing has been activated, a selection list of all DL/I subfiles that meet the specified selection criteria is displayed on a **Selection** screen.

This section describes the columns and fields contained on a **Selection** screen and the line commands provided to further process a DL/I subfile:

- [Columns and Fields](#)

- [Line Commands](#)

## Columns and Fields

The following columns and fields are displayed on a **Selection** screen:

Column/Field	Explanation
C	Input field for line commands (see below).
Subfile	The name of the DL/I subfile that meets the specified selection criteria.
Message	The message that indicates the processing status of a DL/I subfile. For possible messages, see <a href="#">Status Messages</a> .
Listed Subfiles	The type of DL/I subfile: NDB (including UDFs) or NSB.

## Line Commands

One of the following line commands can be entered in the C (Command) column of a **Selection** screen:

Line Command	Function
A	Process NSB subfile or NDB and UDF subfiles.  Line command A is not available with the list function.
L	List NSB subfile or NDB and UDF subfiles.

## Direct Command Syntax for DL/I Subfiles

This section shows the syntax that applies when performing a SYSMAIN function on a DL/I subfile by using direct commands in either online or batch mode. For general instructions on using direct commands, refer to [Executing Commands](#).

For explanations of the keywords and variable values used in the syntax diagrams below, refer to [Keywords and Variables in Direct Commands](#). The symbols in the syntax diagrams correspond to the syntax symbols used for system commands. These symbols are explained in *System Command Syntax* in the *System Commands* documentation.

The syntax of the *where-clause* and the *with-clause* are identical for each command.

This section covers the following topics:

- [COPY and MOVE](#)
- [DELETE and LIST](#)
- [where-clause](#)

- `with-clause`

## COPY and MOVE

```
{ COPY } DL1 name [with-clause] [ FM where-clause ] TO where-clause
{ MOVE }
```

### Examples:

```
COPY DL1 SUBFILE1 TYPE D FM DBID 1 FNR 5 TO DBID 2 FNR 5
```

```
C DL1 SUBFILE REP FM FNR 6 TO FNR 7 TYPE D
```

```
COPY DL1 SUBFILE1 TYPE P TO DBID 3 FNR 7
```

```
MOVE DL1 SUBFILE1 TYPE D FM DBID 1 FNR 5 TO DBID 2 FNR 5
```

```
M DL1 SUBFILE1 REP FM FNR 6 TO FNR 7 TYPE D
```

## DELETE and LIST

```
{ DELETE } DL1 name [ IN where-clause ] [with-clause]
{ LIST }
```

### Examples:

```
DELETE DL1 S* TYPE D
```

```
D DL1 TEST* IN DBID 177 FNR 205 TYPE D
```

```
LIST DL1 SUBFILE* IN DBID 1 FNR 5 TYPE D
```

```
L DL1 SF* TYPE P
```

## where-clause

```
[WHERE] [DBID dbid] [FNR fnr] [NAME vsam-name]
[CIPHER cipher] [ { PASSWORD } password ]
                  [ PSW ]
```

**with-clause**

[WITH] [TYPE <i>type</i> ] [REPLACE] [RCOP] [ PROMPT NOPROMPT ] [ MON NOMON ] [HELP]
--

# 95 Processing DDMs

---

- Fields in the DDMs Menu ..... 578
- Selection Lists for DDMs ..... 580
- Direct Command Syntax for DDMs ..... 581

All SYSMAIN functions except the find and rename functions can be performed on DDMs.

The DDM environment specification must always correspond to the database ID (DBID) and file number (FNR) of the relevant FDIC system file.

This section describes menu functions and selection list options provided to perform a SYSMAIN function on DDMs and the syntax that applies when using direct commands.

## Fields in the DDMs Menu

The **DDMs** menu contains all SYSMAIN functions required for the processing of DDMs. The fields provided in the menu are described in the following table:

Field	Explanation	
<b>Code</b>	Specifies the function to be performed as described in <i>Description of Functions</i> :	
	C	Copy DDM.
	D	Delete DDM.
	L	List DDM.
	M	Move DDM.
<b>Sel. List</b>	Specifies whether selective processing or automated processing is used:	
	Y	Yes. Selective processing is activated. A selection list is displayed when processing DDMs. This is the default setting.  For the columns and line commands available on a selection list, see <i>Selection Lists for DDMs</i> .
	N	No. Selective processing is deactivated.
<b>Name</b>	The name of the DDM to be processed or a range of names: see also <i>Specifying a Range of Names</i> .	
<b>DDM DBID</b>	This field can be used to select only DDMs which were cataloged under the specified database ID (DBID).  Valid database IDs are 1 to 65535.  If this field is empty or set to 0 (zero), the DBID is not used as a selection criterion.	
<b>DDM FNR</b>	This field can be used to select only DDMs which were cataloged under the specified file number (FNR).  Valid file numbers are 1 to 65535.	

Field	Explanation								
	If this field is empty or set to 0 (zero), the FNR is not used as a selection criterion.								
<b>Database</b>	<p>The database ID (DBID) of a source or a target database.</p> <p>The source database contains the FDIC system file where the DDM to be processed is stored. The target database contains the FDIC system file to which the DDM is to be copied or moved.</p> <p>Valid database IDs are 1 to 65535.</p>								
<b>File</b>	<p>The file number (FNR) of a source or a target FDIC system file.</p> <p>Valid file numbers are 1 to 65535.</p> <p>The source file contains the DDM to be processed. The target file is the file to which the DDM is to be copied or moved.</p>								
<b>Name</b> (next to <b>File</b> )	<p>Only applies to VSAM files.</p> <p>The DDNAME/FCT entry for the source or target file number.</p>								
<b>Replace</b>	<p>Specifies whether a DDM is to be replaced when using the move or copy function:</p> <table border="1"> <tbody> <tr> <td></td> <td></td> </tr> <tr> <td>Y</td> <td>Yes. A DDM with the same name which exists in the target environment is replaced.</td> </tr> <tr> <td>N</td> <td>No. A DDM with the same name which exists in the target environment is not replaced. This is the default setting.</td> </tr> <tr> <td></td> <td></td> </tr> </tbody> </table> <p>See also <a href="#">Using the Replace Option</a>.</p>			Y	Yes. A DDM with the same name which exists in the target environment is replaced.	N	No. A DDM with the same name which exists in the target environment is not replaced. This is the default setting.		
Y	Yes. A DDM with the same name which exists in the target environment is replaced.								
N	No. A DDM with the same name which exists in the target environment is not replaced. This is the default setting.								
<b>Del.NSC-Def.</b>	<p>This field only appears in a Natural Security environment.</p> <p>Indicates deletion of Natural Security definitions.</p> <p>If a DDM is deleted from a source environment or moved to a new environment and different FSEC system files have been specified, you can use this field to specify whether or not to delete the DDM definition in the source FSEC.</p> <p>Possible values are:</p> <table border="1"> <tbody> <tr> <td></td> <td></td> </tr> <tr> <td>Y</td> <td>Yes. The DDM definition in the source FSEC is deleted.</td> </tr> <tr> <td>N</td> <td>No. The DDM definition in the source FSEC is kept. This is the default setting.</td> </tr> </tbody> </table>			Y	Yes. The DDM definition in the source FSEC is deleted.	N	No. The DDM definition in the source FSEC is kept. This is the default setting.		
Y	Yes. The DDM definition in the source FSEC is deleted.								
N	No. The DDM definition in the source FSEC is kept. This is the default setting.								

## Selection Lists for DDMs

If selective processing has been activated, a selection list of all DDMs that meet the specified selection criteria is displayed on a **Selection** screen.

This section describes the columns contained on a **Selection** screen and the line commands provided to further process a DDM:

- [Columns](#)
- [Line Commands](#)

### Columns

The following columns are displayed on a **Selection** screen:

Column	Explanation
C	Input field for line commands (see below).
DDM Name	The name of the DDM that meets the specified selection criteria.
DBID	The database ID (DBID) under which the DDM was cataloged. If no DBID or 0 (zero) is listed, the DDM was cataloged under the current DBID.
FNR	The file number (FNR) under which the DDM was cataloged.
Cat Date	The date on which the DDM was cataloged.
Time	The time when the DDM was cataloged.
User ID	The ID of the user who cataloged the DDM.
Message	The message that indicates the processing status of a DDM. This column only appears after the line command A was executed on a DDM. For possible messages, see <a href="#">Status Messages</a> .

The columns **Cat Date**, **Time** and **User ID** are empty if the DDM was cataloged under an older version of Natural which does not supply this type of information.

## Line Commands

One of the following line commands can be entered in the **C** (Command) column of a **Selection** screen:

Line Command	Function
A	Process the specified DDM.  This line command is not available with the list function.
L	List the specified DDM.  For detailed information on the field definitions displayed, see <i>Columns of Field Attributes</i> in the <i>SYSDDM Utility</i> documentation.
R	List all automatic rules linked to the specified DDM.  Line command R invokes the <b>Rule Usage for DDM/Fields</b> screen, which indicates whether a DDM uses a rule or not. If a rule is used, the name of the rule and the DDM field referenced are listed on the screen.
X	Only applies if Natural Connection and Entire Connection are installed.  Download DDM(s) to a PC. For detailed instructions, refer to <a href="#">Downloading Objects to a PC</a> .

## Direct Command Syntax for DDMs

This section shows the syntax that applies when performing a SYSMAIN function on DDMs by using direct commands in either online or batch mode. For general instructions on using direct commands, refer to [Executing Commands](#).

For explanations of the keywords and variable values used in the syntax diagrams below, refer to [Keywords and Variables in Direct Commands](#). The symbols in the syntax diagrams correspond to the syntax symbols used for system commands. These symbols are explained in *System Command Syntax* in the *System Commands* documentation.

The syntax of the *where-clause* and the *with-clause* are identical for each command.



**Note:** For compatibility reasons, instead of the keyword DDM you can use the keyword VIEW (or its short form V) in direct commands for DDMs.

This section covers the following topics:

- COPY and MOVE
- DELETE
- LIST

- *where-clause*
- *with-clause*

## COPY and MOVE

```
{ COPY } DDM name [ FM where-clause ] TO where-clause [with-clause]
{ MOVE }
```

### Examples:

```
C DDM PERSONNEL FM FNR 20 TO FNR 24 REPLACE
```

```
COPY DDM C< FM FNR 20 TO FNR 24
```

```
M DDM PERSONNEL FM FNR 20 TO FNR 24 REPLACE
```

```
MOVE DDM C< FM FNR 20 TO FNR 24
```

## DELETE

```
DELETE DDM name [with-clause] [ IN where-clause ]
```

### Example:

```
D DDM FINANCE IN DBID 12 FNR 27
```

## LIST

```
LIST DDM name [ IN where-clause ] [with-clause]
```

### Example:

```
L DDM * IN DBID 1 FNR 5
```

## where-clause

```
[WHERE] [DBID dbid] [FNR fnr] [NAME vsam-name] [CIPHER cipher]
[ { PASSWORD } password ]
[ { PSW } ]
[DIC (dbid,fnr,password,cipher)]
[SEC (dbid,fnr,password,cipher)]
```

## Separators

Commas must be used as separators between the values following the DIC and SEC keywords, or if a value is missing. For example: DIC (10, ,secret,2a). If the session parameter ID (see *ID - Input Delimiter Character* in the *Parameter Reference* documentation) has been set to a comma, use a slash (/) as the separator between values.

## with-clause

```
[WITH] [ { DDMDBID } DDM-dbid ] [ { DDMFNR } DDM-fnr ] [REPLACE] [ PROMPT  
NOPROMPT ] [ MON  
NOMON ] [HELP]
```



# 96

## Keywords and Variables in Direct Commands

---

- Description of Keywords ..... 586
- Specifying a Range of Names ..... 596

This section describes all keywords and variables that are relevant when using direct commands in online or batch mode. Each keyword represents a parameter that is used to specify object selection criteria or set an option for the command being executed. If indicated, a variable value must be supplied with a keyword.

The symbols used in the syntax diagrams shown below correspond to the syntax symbols used for system commands which are explained in *System Command Syntax* in the *System Commands* documentation.

For the direct command syntax to which the keywords refer, including details on the *where-clause* and the *with-clause* mentioned in this section, refer to the object-type specific sections of the SYSMAIN Utility documentation.

## Description of Keywords

This section explains the keywords and corresponding variable values (if required) used in a direct command.

Keywords are listed alphabetically. Letters in italics represent variable values that must be supplied with a keyword. For each variable value, the Natural data format and length is indicated.

Keyword	Value	Natural Data Format/Length	Explanation
ALL or A	<i>name</i>	A9	Only applies to programming objects.  The name of the object to be processed or a range of names (see also <i>Specifying a Range of Names</i> ). Any saved (source) objects and/or cataloged objects are processed.
AS	<i>new-name</i>	A8 or A12	Not applicable to DL/I subfiles and DDMS.  The new name to be given to an object when it is renamed with the <b>RENAME</b> command.  Format/length A12 only applies to debug environments.
	<i>new-number</i>	N4	For error messages:  The new number to be given to an error message, or the start number of a range of new numbers to be given to a range of existing error messages when using the <b>RENAME</b> command.
CATALOGED	<i>name</i>	A9	Only applies to programming objects.  The name of the cataloged object to be processed or a range of names (see also <i>Specifying a Range of Names</i> ).

Keyword	Value	Natural Data Format/Length	Explanation
CIPHER	<i>cipher</i>	A8	<p>The Adabas cipher code of a source file and/or target system file which is used in a <i>where-clause</i>.</p> <p>For rules and DDMs: The corresponding <b>DIC</b> specification can be used instead of CIPHER. If <i>cipher</i> is specified twice, the one specified last will be used.</p>
DBID	<i>dbid</i>	N5	<p>The database ID (DBID) of a source or a target database.</p> <p>The source database contains the system file where the object to be processed is stored. The target database contains the system file to which the object is to be copied or moved, or where the object is renamed (or in the case of an error message, renumbered) if relevant.</p> <p>Valid DBIDs are 1 to 65535.</p> <p>If no DBID or file number (<b>FNR</b>) is specified and SYSMAIN is called with the system command SYSMAIN or the subprogram MAINUSER (see also <i>Invoking SYSMAIN with Appl. Programming Interface</i>), the following applies: The DBID and FNR of the system file from which SYSMAIN was called are always used. For example: if you enter SYSMAIN from a library contained in the FUSER system file, the DBID and FNR of this file are used.</p> <p>For rules and DDMs: The corresponding <b>DIC</b> specification can be used instead of DBID. If <i>dbid</i> is specified twice, the one specified last will be used.</p>
DDM or VIEW	<i>name</i>	A32	<p>Only applies to DDMs.</p> <p>The name of the DDM to be processed or a range of names. See also <i>Specifying a Range of Names</i>.</p>
DDMDBID or DDBID	<i>dgm-dbid</i>	N5	<p>Only applies to DDMs.</p> <p>The DDM database ID (DBID): All DDMs that were cataloged under the specified DBID are processed.</p> <p>Valid DDM DBIDs are 0 to 65535. If no value or 0 (zero) is specified, the DDM DBID is not checked.</p>
DDMFNR or DFNR	<i>dgm-fnr</i>	N5	<p>Only applies to DDMs.</p> <p>The DDM file number (FNR): All DDMs that were cataloged under the specified FNR are processed.</p> <p>Valid DDM FNRs are 0 to 65535. If no value or 0 (zero) is specified, the DDM FNR is not checked.</p>

Keyword	Value	Natural Data Format/Length	Explanation
DEBUG	<i>name</i>	A12	<p>Only applies to debug environments.</p> <p>The name of the debug environment to be processed or a range of names. See also <a href="#">Specifying a Range of Names</a>.</p>
DIC	<i>dbid</i> <i>fnr</i> <i>password</i> <i>cipher</i>	A80	<p>Not applicable to error messages, profiles and DL/I subfiles.</p> <p>Specifies the environment of the FDIC source and/or target system file: database ID (<i>dbid</i>), file number (<i>fnr</i>), Adabas password (<i>password</i>) and Adabas cipher code (<i>cipher</i>).</p> <p>For rules and DDMs: DBID, FNR, CIPHER and PASSWORD specifications can be used instead of the corresponding DIC specifications, or vice versa. If an item is specified twice, the one specified last will be used.</p>
DL1 or SUBFILES or S	<i>name</i>	A8	<p>Only applies to DL/I subfiles.</p> <p>The name of the DL/I subfile to be processed or a range of names. See also <a href="#">Specifying a Range of Names</a>.</p>
ERROR	<i>number</i>	N4	<p>Only applies to error messages.</p> <p>The number of the error message to be processed or the start number of a range of numbers if <b>THRU</b> is specified.</p>
EXTEND	-	-	<p>Only applies to programming objects and the <b>LIST</b> or <b>FIND</b> command when being used in batch mode.</p> <p>If <b>EXTEND</b> is <i>not</i> specified, a short list of the objects contained in the specified library is displayed. The short list contains the name and the type of the object and indicates whether a source object and/or a cataloged object exists.</p> <p>If <b>EXTEND</b> is specified, an extended list of the objects contained in the specified library is displayed. In addition to the information displayed if <b>EXTEND</b> is not specified, the extended list provides information from the object directory: programming mode, Natural version, user ID, saved/cataloged date and time, and the source of the object (if any).</p>
FROM or FM or IN	<i>lib-name</i>	A8	<p>For programming objects, debug environments and error messages: Specifies a source library.</p> <p>For profiles, rules, DDMs and DL/I subfiles: Introduces a <i>where-clause</i>.</p>
FMDATE or FMDD	<i>date-from</i>	A10	<p>Only applies to programming objects.</p> <p>The start date of a time period: All objects which were saved or cataloged on or after the specified date are processed.</p>

Keyword	Value	Natural Data Format/Length	Explanation
			<p>If no end date is specified with <b>TODATE</b>, all objects from the specified date are selected for processing.</p> <p>A date must be specified according to the setting of the <b>DTFORM</b> profile parameter (see <i>DTFORM - Date Format</i> in the <i>Parameter Reference</i> documentation) as indicated in the upper right corner of a <b>SYSMAIN</b> menu screen. The default setting is the international format <i>YYYY-MM-DD</i> (<i>YYYY</i> = year, <i>MM</i> = month, <i>DD</i> = day), for example, 2005-08-20.</p>
FMTIME or FMTM or FMTT	<i>time-from</i>	A5	<p>Only applies to programming objects and if <b>FMDATE</b> is specified.</p> <p>Specifies a start time: All objects which were saved or cataloged at or after the specified time (and date) are processed.</p> <p>A time must be specified in the format <i>HH:II</i> (<i>HH</i> = hours, <i>II</i> = minutes), for example, 11:33.</p>
FNR	<i>fnr</i>	N5	<p>The file number (FNR) of a source or a target system file (FNAT, FDIC or FUSER).</p> <p>The source system file contains the object to be processed. The target system file is the system file to which the object is to be copied or moved, or where the object is renamed (or in the case of an error message, renumbered) if relevant.</p> <p>Valid FNRs are 1 to 65535.</p> <p>If no database ID (DBID) or FNR is specified and <b>SYSMAIN</b> is called with the system command <b>SYSMAIN</b> or the subprogram <b>MAINUSER</b> (see also <i>Invoking SYSMAIN with Appl. Programming Interface</i>), the following applies: The DBID and FNR of the system file from which <b>SYSMAIN</b> was called are always used. For example: if you enter <b>SYSMAIN</b> from a library contained in the <b>FUSER</b> system file, the DBID and FNR of this file are used.</p> <p>For rules and DDMs: The corresponding <b>DIC</b> specification can be used instead of FNR. If <i>fnr</i> is specified twice, the one specified last will be used.</p>
HELP or ?	-	-	<p>Activates online <b>selective processing</b>.</p> <p>You can either include the keyword <b>HELP</b> in the <i>with-clause</i> or enter a question mark (?) as the final character of an object name.</p>

Keyword	Value	Natural Data Format/Length	Explanation
LANGUAGE	<i>language</i>	A9	<p>Only applies to error messages.</p> <p>The code of the language of the error message to be processed.</p> <p>The languages can be specified using any combination of language codes. For information on which language code is assigned to which language, see <i>Language Code Assignments</i> in *LANGUAGE in the <i>System Variables</i> documentation. See also <i>Specifying Languages</i>.</p> <p>You can use an asterisk (*) to select all existing languages of the error messages to be processed.</p>
LIBRARY or APPLIC or APL	<i>lib-name</i>	A8	<p>Only applies to programming objects, debug environments and error messages.</p> <p>An optional keyword that indicates the name of a source or a target library. If you omit the keyword and respective value, the library where you logged on before you invoked SYSMAIN is used for processing.</p> <p>The source library contains the object to be processed. The target library is the library to which the object is to be copied or moved, or where the object is renamed (or in the case of an error message, renumbered).</p> <p>For system error messages, specify NATURAL - SYSTEM or NATURAL - SYS as <i>lib-name</i>.</p> <p><i>lib-name</i> must be specified immediately after the FROM and TO keywords. If the optional keyword LIBRARY is used, it must be entered between FROM or TO and <i>lib-name</i>.</p>
MON or NOMON or MONOFF	-	-	<p>Not applicable in batch mode.</p> <p>Activates (MON) or deactivates (NOMON or MONOFF) tracing of the current activity in SYSMAIN. During processing, you are informed as to which object is being read, deleted, updated, added, and whether an error occurs. With programming objects, you are also informed about the action taken with the XRef data. This function is effective only with TP environments which can run in non-conversational mode.</p>
NAME	<i>vsam-name</i>	A8	The DDNAME/FCT entry for the source or target file number.
PROMPT or NOPROMPT	-	-	<p>Not applicable in batch mode.</p> <p>Enables (PROMPT) or disables (NOPROMPT) the SYSMAIN prompts. With NOPROMPT, no confirmation screen is displayed. For example, before any deletion, SYSMAIN prompts you for confirmation.</p>

Keyword	Value	Natural Data Format/Length	Explanation
PASSWORD or PSW	<i>password</i>	A8	The Adabas password of a source file and/or target system file which is used in a <i>where-clause</i> .  For rules and DDMs: The corresponding <b>DIC</b> specification can be used instead of PASSWORD. If <i>password</i> is specified twice, the one specified last will be used.
PROFILE	<i>name</i>	A8	Only applies to profiles.  The name of the profile to be processed or a range of names. See also <i>Specifying a Range of Names</i> .
RCOP	-	-	Specifies that a copy of the object being renamed is to be made.
REPLACE	-	-	Activates the replace option used in a <i>with-clause</i> .  An object with the same name in the target environment is replaced by the object to be processed.  <b>Note:</b> If a programming object is replaced it is also deleted from the Natural buffer pool; any existing cross-reference records are also deleted if Predict is installed.
RULE	<i>name</i>	A32	Only applies to rules.  The name of the rule to be processed or a range of names. See also <i>Specifying a Range of Names</i> .
SAVED	<i>name</i>	A9	Only applies to programming objects.  The name of the saved (source) object to be processed or a range of names. See also <i>Specifying a Range of Names</i> .
SEC	<i>dbid</i> <i>fnr</i> <i>password</i> <i>cipher</i>	A80	Not applicable to profiles and DL/I subfiles.  Specifies the environment of the FSEC source and/or target system file: database ID ( <i>dbid</i> ), file number ( <i>fnr</i> ), Adabas password ( <i>password</i> ) and Adabas cipher code ( <i>cipher</i> ).
SETNO	<i>set-number</i>	N2	The number of the retained Predict set created with the Predict XRef save set option of the LIST XREF command. You can apply all SYSMAIN processing functions to the objects included in this set.  If any valid number is specified, SYSMAIN assumes a Predict set. If no number is specified, normal object processing is assumed.
SETLIBRARY	<i>set-library</i>	A8	Activates the option to overwrite the library specification for a Predict set as a part of the security for Predict files.  SETLIBRARY is only evaluated if a valid number has been specified for SETNO.

Keyword	Value	Natural Data Format/Length	Explanation
SETUSER	<i>set-user</i>	A8	<p>Activates the option to overwrite the user-ID specification for a Predict set as a part of the security for Predict files.</p> <p>SETUSER is only evaluated if a valid number has been specified for SETNO.</p>
STOWED or BOTH	<i>name</i>	A9	<p>Only applies to programming objects.</p> <p>The name of an object (or a range of names) for which the saved (source) <i>and</i> the cataloged object are to be processed (see also <a href="#">Specifying a Range of Names</a>). Only an object that exists as both a saved object <i>and</i> a cataloged object is processed.</p> <p>The exceptions to this are copycode, text and recording, neither of which can be cataloged. However, they are included in processing when this option is specified.</p>
THRU	<i>number</i> or <i>new-number</i>	N4	<p>Only applies to error messages.</p> <p>The end number of a range of error message numbers to be processed if a start number is specified with <a href="#">AS</a> .</p>
TID	<i>terminal-ID</i>	A8	<p>Only applies to programming object.</p> <p>A terminal ID: All objects that were saved or cataloged on the specified terminal are processed.</p>
TO	<i>lib-name</i>	A8	<p>For programming objects, debug environments and error messages: Specifies a target library.</p> <p>For profiles, rules, DDMs and DL/I subfiles: Introduces a <i>where-clause</i>.</p>
TODATE or TODD	<i>date-to</i>	A10	<p>Only applies to programming objects.</p> <p>The end date of a time period: All objects which were saved or cataloged on or before the specified date are processed. A start date can be specified with <a href="#">FMDATE</a> .</p> <p>A date must be specified according to the setting of the <code>DTFORM</code> profile parameter (see <a href="#">DTFORM - Date Format</a> in the <a href="#">Parameter Reference</a> documentation) as indicated in the upper right corner of a SYSMAIN menu screen. The default setting is the international format <code>YYYY-MM-DD</code> (<code>YYYY</code> = year, <code>MM</code> = month, <code>DD</code> = day), for example, <code>2005-08-20</code>.</p>

Keyword	Value	Natural Data Format/Length	Explanation
TOTIME or TOTT or TOTM	<i>time-to</i>	A5	<p>Only applies to programming objects and if <b>TODATE</b> is specified.</p> <p>The end time of a time period: All objects which were cataloged or saved at or before the specified time (and date) are processed.</p> <p>The time must be specified in the format <i>HH:II</i> (<i>HH</i> = hours, <i>II</i> = minutes), for example, 11:33.</p>
TYPE	<i>type</i>	-	The type of <b>programming object, error message, profile, rule</b> or <b>DL/I subfile</b> to be processed as listed in <i>TYPE Specification</i> below.
USER or USR	<i>user-id</i>	A8	<p>Only applies to programming objects.</p> <p>A user ID: All objects that were saved or cataloged by the specified user are processed.</p>
WHERE	<i>where-clause</i>	-	<p>An optional keyword that indicates the start of a <i>where-clause</i>.</p> <p>The <i>where-clause</i> must always follow the FROM or TO keyword and <i>lib-name</i> (if relevant); the sequence of the keywords and values within the clause can be specified in any order.</p> <p>For details, see the direct command syntax in the object-type specific sections of the <i>SYSMAIN Utility</i> documentation.</p>
WITH	<i>with-clause</i>	-	<p>An optional keyword that indicates the start of a <i>with-clause</i>.</p> <p>The keywords and values of the <i>with-clause</i> can be specified in any order, and the <i>with-clause</i> can be placed in any location within the direct command string, except in the first three positions.</p> <p>For details, see the direct command syntax in the object-type specific sections of the <i>SYSMAIN Utility</i> documentation.</p>
XREF	F or N or S or Y	A1	<p>Only applies to programming objects.</p> <hr/> <p>Indicates whether cross-reference (XRef) data stored on Predict system files is to be processed.</p> <hr/> <p>You can specify one of the following values:</p> <hr/>

Keyword	Value	Natural Data Format/Length	Explanation
			F All XRef data is processed and the object must be documented in Predict.
			N XRef data is not processed, except when using the DELETE command. If a cataloged object is deleted or replaced, SYSMAIN always deletes any existing XRef data for this object.
			S A specified object is processed regardless of whether it has XRef data or not.
			Y All XRef data is processed.
			For further details, see <a href="#">XRef Considerations</a> .

### TYPE Specification - Programming Objects

#### Natural Data Format/Length: A20

The following table lists all valid object-type codes for programming objects:

Code	Object Type
P	Program
N	Subprogram
S	Subroutine
M	Map
H	Helproutine
0	ISPF macro
3	Dialog
5	Processor
A	Parameter data area
G	Global data area
L	Local data area
C	Copycode
T	Text
R	Report
Z	Recording
4	Class
7	Function

Code	Object Type
8	Adapter
9	Resource
*	All programming object types

**TYPE Specification - Error Messages**

**Natural Data Format/Length: A1**

The following table lists all valid type codes for error messages:

Code	Type
S	Short error message
E	Extended (long) error message
A	All error message types: short and/or extended messages

**TYPE Specification - Profiles**

**Natural Data Format/Length: A3**

The following table lists all valid type codes for profiles:

Code	Type
E	Editor profile
D	Device profile
M	Map profile
P	Parameter profile
*	All profile types.

## TYPE Specification - Rules

### Natural Data Format/Length: A2

The following table lists all valid type codes for rules:

Code	Type
A	Automatic rule
F	Free rule
AF	All rule types: automatic and/or free rules.

## TYPE Specification - DL/I Subfiles

### Natural Data Format/Length: A1

The following table lists the valid type codes for DL/I subfiles:

Code	Type
D	NDBs and UDFs
P	NSBs

## Specifying a Range of Names

---

All SYSMAIN functions provide the option to specify either a name or a range of names for the objects to be processed. In addition, in menu mode, on a **Find Selection** or **List Selection** screen, you can specify a name or a range of names to limit the number of objects displayed. See also [To shorten a selection list](#) in *Using a Selection List*.

When using the find or the list function with programming objects, you can also specify a range of library names. The same applies when using the list function with debug environments or the find function with error messages. However, specifying library ranges may have a negative effect on the response time depending on how often the selection criteria occur.

The valid notations for name ranges are listed below where *value* denotes any combination of one or more characters:

Input	Items Selected
*	All items.
<i>value</i> *	All items with names that start with <i>value</i> . Example: AB* Selected: AB, AB1, ABC, ABEZ Not selected: AA1, ACB
<i>value</i> >	All items with names greater than or equal to <i>value</i> . Example: AB> Selected: AB, AB1, BBB, ZZZZZZZ Not selected: AA1, AAB
<i>value</i> <	All items with names less than or equal to <i>value</i> . Example: AX< Selected: AB, AWW, AX Not selected: AXA, AY



# 97

## Special Commands Issued to SYSMAIN

There are commands that can be issued to the SYSMAIN utility to perform functions related to the operation of the utility itself or to define the security for Natural system files.

Command	Function
ADAON or NOADA or ADAOFF	Activate (ADAON) or deactivate (NOADA or ADAOFF) error trapping.  Trap abnormal database errors (only applicable online with programming objects) for debugging purposes.
BATCH or NOBATCH	Switch the SYSMAIN utility into batch mode (BATCH), whereby all processing is done as if SYSMAIN was running in batch. If online automated processing is used, a batch report is then displayed: see also <i>Online Report Mode</i> .  NOBATCH switches the SYSMAIN utility back to online mode.
CLEAR	Clear the current work area. This function can be useful if the source code of a large object is contained in the work area and the SYSMAIN utility therefore requires a larger <code>ESIZE</code> .
DISPLAY	Display the extended (long) message text for the error which has occurred.
FINDFIRST	Stop the FIND command for programming objects when the first library is found that contains the specified object.
FINDALL	Cause the FIND command for programming objects to search in all libraries.
MON or NOMON or MONOFF	Activate (MON) or deactivate (NOMON or MONOFF) tracing of the current activity in SYSMAIN. See also <b>MON</b> in <i>Keywords and Variables in Direct Commands</i> .
PROMPT or NOPROMPT	Enable (PROMPT) or disable (NOPROMPT) the SYSMAIN prompts. See also <b>PROMPT</b> in <i>Keywords and Variables in Direct Commands</i> .
SET	Invoke the <b>Command Help</b> window where all special SYSMAIN commands are explained.

Command	Function																
SET FDIC	<p>Invoke a window to specify Adabas security information for the Predict system file. This refers to the profile parameter FDIC (see <i>FDIC - Predict System File</i> in the <i>Parameter Reference</i> documentation). In batch mode, you can specify security information by using the keyword DIC in a <i>where-clause</i> as indicated in the direct command syntax in the object-type specific sections of the <i>SYSMAIN Utility</i> documentation.</p> <p>See also <a href="#">Special Considerations for Administrators</a>.</p>																
SET FNAT	<p>Invoke a window to specify Adabas security information for the SYSMAIN source and target system files. In batch mode, you can specify security information by using the keyword SEC in a <i>where-clause</i> as indicated in the direct command syntax in the object-type specific sections of the <i>SYSMAIN Utility</i> documentation.</p> <p>See also <a href="#">Special Considerations for Administrators</a>.</p>																
SET FSEC	<p>Invoke a window to specify Adabas security information of the FSEC system file if Natural Security is installed. This refers to the profile parameter FSEC (see <i>FSEC - Natural Security System File</i> in the <i>Parameter Reference</i> documentation).</p> <p>See also <a href="#">Special Considerations for Administrators</a>.</p>																
SET PC	<p>Only applies if Natural Connection and Entire Connection are installed.</p> <p>Activate the PC connection. This setting can be intermittently changed with the %+ and %- terminal commands (see also <i>Enable/Disable Use of Natural Connection</i> in the <i>Terminal Commands</i> documentation). SET PC then results in SYSMAIN re-verifying the status of the PC connection.</p>																
STATUS	<p>Display the current values of SYSMAIN variables that are important for Software AG technical support.</p>																
TOTAL	<p>Invoke the <b>Results of Function</b> window which verifies the processing of the last SYSMAIN function executed. The following information is displayed for saved (source) and cataloged objects:</p> <table border="1"> <tbody> <tr> <td></td> <td></td> </tr> <tr> <td><b>Read</b></td> <td>Total number of objects which were actually read, based on the object name specification.</td> </tr> <tr> <td><b>Rejected</b></td> <td>Total number of objects read which were then rejected, based on the selection criteria specified. See also <a href="#">Object Rejection and Reasons</a>.</td> </tr> <tr> <td><b>Processed</b></td> <td>Total number of objects which were processed.</td> </tr> <tr> <td><b>Added</b></td> <td>Total number of new objects added to the target environment.</td> </tr> <tr> <td><b>Updated</b></td> <td>Total number of existing objects updated. (Where possible, SYSMAIN attempts to update existing objects instead of deleting and adding new ones.)</td> </tr> <tr> <td><b>Deleted</b></td> <td>Total number of objects deleted from either the source or target environment, depending on the function and the setting of the replace option.</td> </tr> <tr> <td><b>Replaced</b></td> <td>Total number of objects which were replaced in the target environment.</td> </tr> </tbody> </table>			<b>Read</b>	Total number of objects which were actually read, based on the object name specification.	<b>Rejected</b>	Total number of objects read which were then rejected, based on the selection criteria specified. See also <a href="#">Object Rejection and Reasons</a> .	<b>Processed</b>	Total number of objects which were processed.	<b>Added</b>	Total number of new objects added to the target environment.	<b>Updated</b>	Total number of existing objects updated. (Where possible, SYSMAIN attempts to update existing objects instead of deleting and adding new ones.)	<b>Deleted</b>	Total number of objects deleted from either the source or target environment, depending on the function and the setting of the replace option.	<b>Replaced</b>	Total number of objects which were replaced in the target environment.
<b>Read</b>	Total number of objects which were actually read, based on the object name specification.																
<b>Rejected</b>	Total number of objects read which were then rejected, based on the selection criteria specified. See also <a href="#">Object Rejection and Reasons</a> .																
<b>Processed</b>	Total number of objects which were processed.																
<b>Added</b>	Total number of new objects added to the target environment.																
<b>Updated</b>	Total number of existing objects updated. (Where possible, SYSMAIN attempts to update existing objects instead of deleting and adding new ones.)																
<b>Deleted</b>	Total number of objects deleted from either the source or target environment, depending on the function and the setting of the replace option.																
<b>Replaced</b>	Total number of objects which were replaced in the target environment.																

Command	Function	
	<b>Not Repl.</b>	Total number of objects which were <i>not</i> replaced in the target environment.
	<b>Recs.Read:</b>	Total number of records which were read.
. or END  or QUIT	Terminate SYSMAIN.	



# 98 Processing Status and Error Notification

---

- Object Rejection and Reasons ..... 604
- Status Messages ..... 605
- SYSMAIN Error Notification ..... 609

This section describes possible reasons for object rejection during function processing, the status messages displayed after processing, and error notification during processing.

## Object Rejection and Reasons

---

If, during the execution of a SYSMAIN function, one or more objects were found to satisfy the specified selection criteria, but some or all of these objects were then rejected for further processing, any of the following Natural system errors occurs:

4867:Nothing found for this request.  
4810:All data rejected by these selection criteria.  
4893:Normal completion, but some data were rejected.

You can use the SYSMAIN command **TOTAL** (see *Special Commands issued to SYSMAIN*) to review the specific status of a request.

Possible reasons for object rejection are described in the following section:

- Invalid Object Type, Date/Time, User or Terminal
- Identical Target Name
- XREF and User Exits
- No Short or Extended Error Message
- Library Restrictions

### Invalid Object Type, Date/Time, User or Terminal

- An object was selected and then rejected because the object type was not valid for the type of processing specified. For example, when processing programming objects, all maps are rejected if processing has been restricted to objects of the type program or subroutine.
- An object was selected and then rejected because the date or time on which the specified object was saved or cataloged did not fall within the date or time range specified.
- An object was selected and then rejected because the user ID or terminal ID by which the specified object was saved or cataloged did not match with the user ID or terminal ID specified.
- A cataloged object was selected and then rejected because the profile parameter `RECAT` was set to `ON` and there was no saved (source) object corresponding to the cataloged object. See also: [Using Profile Parameter RECAT](#).
- A saved (only) object was selected and then rejected because the profile parameter `RECAT` was set to `ON` and the target environment already contained a cataloged object with the same name. See also: [Using Profile Parameter RECAT](#).

### Identical Target Name

- An object was selected and then rejected because the target environment already contained an object identified by the same name, and the replace option was not activated (**Replace** field set to N or keyword REPLACE not specified).
- A cataloged programming object of type S (subroutine) was selected and then rejected because the name of the external subroutine was identical to the name of another subroutine in the target library.
- A cataloged programming object of type 4 (class) was selected and then rejected because the name of the external class was identical to the name of another class in the target library.

### XREF and User Exits

- The XREF option was activated (**XREF** field *not* set to N or keyword XREF specified) and there were no XRef data for the programming object specified.
- A user exit routine was active and a non-zero return code was returned during processing of the object.

### No Short or Extended Error Message

- An extended (long) error message was selected and then rejected because there was no corresponding short error message in the source library.
- An extended (long) error message was selected but could not be processed because there was no corresponding short error message in the target environment.
- A short error message was selected to be moved, deleted or renumbered, but could not be processed because the corresponding extended error text was not included in the selection criteria. An extended error message must always have a corresponding short error message.

### Library Restrictions

- A library was specified which is controlled by Predict Application Control/Predict Application Audit, and the object cannot be handled by SYSMAIN.

## Status Messages

---

The table below describes the status messages that can be displayed in the **Message** column of a **Selection** screen or in a batch report (see also [Online Report Mode](#)) after a SYSMAIN function was performed on an object. For the line commands mentioned, refer to *Selection Lists* in the object-type specific sections of the SYSMAIN documentation.

Message	Explanation	Function completed?
Class Exists	Only applies to programming objects.  The cataloged object of a class had an external class name already used by another cataloged object of a class in the target environment.	No
Copied	One of the following line commands was executed from a <b>Copy/Rename Selection</b> screen: A, or A, C or S for programming objects, or A, E or S for error messages, or A, D, E, M or P for profiles.	Yes
DB Error: <i>nnn</i>	A database error was returned for the object during processing.	No
Deleted	One of the following line commands was executed from a <b>Delete Selection</b> screen: A, or A, C or S for programming objects, or A, E or S for error messages, or A, D, E, M or P for profiles.	Yes
Dev exists	Only applies to profiles.  The replace option was set to N and line command A or D was executed from a <b>Copy/Move/Rename Selection</b> screen. The device profile with the same name in the target environment was <i>not</i> replaced.	No
Directory	Line command I (display directory) was executed from a <b>Selection</b> screen.	Yes
Edt exists	Only applies to profiles.  The replace option was set to N and line command A or E was executed from a <b>Copy/Move/Rename Selection</b> screen. The editor profile with the same name in the target environment was <i>not</i> replaced.	No
Err: NAT2999	Only applies to programming objects.  A cataloged object was being processed with the XREF option set to F. No Predict entry exists on the FDIC system file specified for the object.	No
Err: NAT4852	A Natural Security violation occurred.	No
Exit: <i>nnn</i>	A user exit routine was active and a non-zero return code was returned by the exit ( <i>nnn</i> = the return code). See also <a href="#">User Exit Routines</a> .	No
Exported	Only applies to programming objects and DDMs.  Line command X (PC download) was executed on a source object from a <b>Selection</b> screen.	Yes
Ext Exists	Only applies to error messages.  An attempt has been made to delete only the short error message of an error message for which an extended (long) error exists. This would have resulted	No

Message	Explanation	Function completed?
	in an extended (long) error message with no corresponding short message. This is not permitted in Natural.	
File Listed	Only applies to rules.  Line command F (display DDMs/fields) was executed on an automatic rule from a <b>Selection</b> screen.	Yes
In Use	An Adabas response code 145 was returned during Natural UPDATE/READ processing of an object.	No
Invalid	Only applies to debug environments.  An invalid line command was entered for one of the debug environments listed on a <b>Selection</b> screen.	No
Listed	Not applicable to DDMs and debug environments.  Line command L (display source code) was executed from a <b>Selection</b> screen.	Yes
Map exists	Only applies to profiles.  The replace option was set to N and line command A or M was executed from a <b>Copy/Move/Rename Selection</b> screen. The map profile with the same name in the target environment was <i>not</i> replaced.	No
Moved	One of the following line commands was executed from a <b>Move Selection</b> screen: A, or A, C or S for programming objects, or A, E or S for error messages, or A, D, E, M or P for profiles.	Yes
Must be Auto	Only applies to rules.  Line command F (display DDMs/fields) was executed on a free rule from a <b>Selection</b> screen. Line command F can only be executed on automatic rules.	No
Name Erro	Only applies to debug environments.  The rename function was used, but the new name specified was found to be invalid. Either no new name was specified for the selection, or the specified name either contained invalid special characters or did not start with an alphabetic character.	No
NBP Deleted or Ignored	Only applies to programming objects.  Line command B (delete object from buffer pool) was executed from a <b>Selection</b> screen. The message returned upon completion of processing is either <b>NBP Deleted</b> or <b>Ignored</b> , depending on whether deletion from the buffer pool for the specified object(s) has been confirmed or not.	Yes/No

Message	Explanation	Function completed?
No Lang 1	<p>Only applies to error messages.</p> <p>Only language code 1 (English) is available for Natural system extended (long) error messages. An attempt has been made to copy a Natural system extended error message, and language code 1 has not been specified as a language.</p>	No
no Short Err	<p>Only applies to error messages.</p> <p>An extended (long) error message was selected for further processing, but the target error message number has no corresponding short error message.</p>	No
No Xref	<p>Only applies to programming objects.</p> <p>A cataloged object was being processed and the XREF option was set to Y or F. No XRef data exists on the FDIC system file specified for the object.</p>	No
Not Found	<p>Only applies to error messages and profiles.</p> <p>An error in the update logic occurred during processing and the requested error message or profile could not be found. This implies that the specified error message or profile was deleted in the time between selection and update.</p>	No
Not Replaced or Not Repld	<p>The replace option was set to N and one of the following line commands was executed from a <b>Copy/Move/Rename Selection</b> screen:</p> <p>A, or A, C or S for programming objects, or A, E or S for error messages, or A, D, E, M or P for profiles.</p> <p>The object with the same name in the target environment was <i>not</i> replaced.</p>	No
Parm exists	<p>Only applies to profiles.</p> <p>The replace option was set to N and line command A or P was executed from a <b>Copy/Move/Rename Selection</b> screen. The parameter profile with the same name in the target environment was <i>not</i> replaced.</p>	No
Printed	<p>Only applies to programming objects and rules.</p> <p>Line command H (produce hardcopy) was executed from a <b>Selection</b> screen.</p>	Yes
Renamed or Copied	<p>Only applies to debug environments and error messages.</p> <p>One of the following line commands was executed from a <b>Rename Selection</b> screen: A for debug environments, or A, E or S for error messages.</p> <p>The message returned upon completion of processing is either <b>Renamed</b> or <b>Copied</b> depending on whether the option to retain the original object was specified.</p>	Yes

Message	Explanation	Function completed?
Renamed as or Copied as	Only applies to programming objects.  Line command A, C or S was executed from a <b>Rename Selection</b> screen. The message returned upon completion of processing is either <b>Renamed as</b> or <b>Copied as</b> , depending on whether the option to retain the original object was specified.	Yes
Replaced	The replace option was set to Y and one of the following line commands was executed from a <b>Copy/Move/Rename Selection</b> screen:  A, or A, C or S for programming objects, or A, E or S for error messages, or A, D, E, M or P for profiles.  The object with the same name in the target environment was replaced.	Yes
Sized	Only applies to programming objects.  Line command Z (calculate size) was executed from a <b>Selection</b> screen.	Yes
Subrtn Exists	Only applies to programming objects.  The cataloged object of a subroutine had an external subroutine name already used by another cataloged subroutine in the target environment.	No
Updated	Only applies to error messages.  The text in the specified language did not previously exist for the error message selected, and SYSMAIN has updated the error message with the new language text.	Yes
Src Locked	Only applies to programming objects.  Line command A or S was executed from a <b>Copy/Move/Rename/Delete Selection</b> screen.  The object was <i>not</i> processed because the source of the respective object was being locked.	No

## SYSMAIN Error Notification

SYSMAIN always attempts to recover in the event of a runtime error during processing. This feature is automatically activated and uses the system variable \*ERROR-TA (see also \*ERROR-TA in the *System Variables* documentation). This feature is deactivated when SYSMAIN is terminated normally.

If a terminal command (see also %% and % . - *Interrupt Current Operation* in the *Terminal Commands* documentation) is used to terminate SYSMAIN, this is considered an abnormal termination, and

the \*ERROR-TA system variable is not reset. It can be reset by re-invoking SYSMAIN and terminating it normally. In the event that you have set the \*ERROR-TA system variable, SYSMAIN resets it to its previously assigned value upon termination.

If invalid data has been specified with respect to the selection criteria, an error message is displayed in the message line. If you are uncertain as to the meaning of the short error message, the special command `DISPLAY` (see also *Special Commands Issued to SYSMAIN*) can be entered to activate a display of the corresponding extended (long) error message text.

This section covers the following topics:

- [Data Entry Errors](#)
- [Processing Errors](#)

### Data Entry Errors

If invalid data has been specified with respect to the selection criteria, an error message is displayed in the message line. In some situations the online help facility for particular entries is invoked. This feature provides you with more detailed information on the error.

If an error occurs in batch mode, an error message and corresponding error number are printed and the SYSMAIN utility is terminated.

### Processing Errors

If you make a request which causes a processing error, SYSMAIN issues an error report as shown in the example below:

```
16:51:08          *** SYSMAIN Error Report ***          2005-08-10

The following internal error occurred while processing the
SYSMAIN function xxxxxx (cc):

    Error in field specification for IF SELECTION statement.

Error Number .. eeee
Program ..... pppppppp
Status Code ... s          Status ..... tttttttt
Line ..... 1111          Level ..... vv
Device ..... dddddddd
User ID ..... uuuuuuuu   User Name ... nn...nn
```

The information contained in the window is useful for analyzing the cause of the error.

The values in the window above have the following meanings:

Field	Value	Explanation
<b>SYSMAIN function</b>	xxxxxx	The SYSMAIN function that is being performed.
	cc	An internal status code useful for Software AG technical support. The following codes can be displayed:  A Automatic processing. D XRef data is being deleted. E Error in processing (flag for SYSMAIN). F Status setting when XRef data is being processed. G Status setting when XRef data is being processed. H Selection list processing. I Option is being processed. S Single object processing. T SYSMAIN termination by command processor. V Status setting when XRef data is being processed. X SYSMAIN termination by command processor. Y Validation error has occurred, redisplay should follow. Z Validation error has occurred, redisplay should follow.
<b>Error Number</b>	eeee	Corresponds to the system variable *ERROR-NR (see the <i>System Variables</i> documentation).
<b>Program</b>	pppppppp	Corresponds to the system variable *PROGRAM (see the <i>System Variables</i> documentation).
<b>Status Code</b>	s	The kind of error. Possible codes are:  C Command processing error. L Logon error. O Object time error. S Non-correctable syntax error.
<b>Line</b>	1111	Corresponds to the system variable *ERROR-LINE (see the <i>System Variables</i> documentation).
<b>Device</b>	ddddddd	Corresponds to the system variable *DEVICE (see the <i>System Variables</i> documentation).
<b>User ID</b>	uuuuuuuu	Corresponds to the system variable *USER (see the <i>System Variables</i> documentation).
<b>Status</b>	tttttttt	Corresponds to the system variable *ERROR-TA (see the <i>System Variables</i> documentation).
<b>Level</b>	vv	Corresponds to the system variable *LEVEL (see the <i>System Variables</i> documentation).

Field	Value	Explanation
User Name	nn . . . nn	Corresponds to the system variable *USER-NAME (see the <i>System Variables</i> documentation).

If a processing error occurs, note the information in the window and press ENTER. The SYSMAIN utility attempts to recover to the last active menu screen, leaving the data values of the input fields unchanged.

If the special command DISPLAY is entered in the window (see also *Special Commands Issued to SYSMAIN*), the extended (long) message text for the error incurred is displayed.

If a processing error occurs during batch processing, the SYSMAIN utility prints the relevant error message and terminates.

Certain user errors can also cause the window to be displayed. Although SYSMAIN attempts to trap all errors during evaluation, this may not always be entirely successful. For example, if a user requests that a DDM be copied from one environment to another, but specifies an invalid database ID (DBID), SYSMAIN attempts to access this database. An Adabas response code of 148 is returned, and the SYSMAIN ERROR transaction is invoked and the window displayed. Similarly, an invalid file can result in a number of errors being sent from the database.

In situations in which an Adabas response code 9 is returned, SYSMAIN writes a message informing you of the error and restarts processing from the last function or menu. If a particular request had not been completed, you can assume that the response code 9 resulted in a BACKOUT TRANSACTION to the last non-completed transaction.

# 99

## Special Considerations for Administrators

---

▪ File Security .....	614
▪ Natural Security .....	615
▪ User Exit Routines .....	616

This section describes the security aspects of the SYSMAIN utility and the user exit routines supplied for SYSMAIN.

## File Security

The file security (that is, passwords and cipher codes) relates to the security that has been defined for a system file in an Adabas or a VSAM environment. If file security has been defined for a system file, you need to specify a password, cipher code and/or VSAM name for the source and/or target system file required before you perform a SYSMAIN function. Otherwise, Adabas or VSAM will issue an appropriate error message. You do not have to provide security information for the default system files assigned to the Natural session at the start of the SYSMAIN utility.

► **To specify passwords and cipher codes**

- 1 From any SYSMAIN utility menu, invoke a security window for the required system file by using either a PF key or a special command as indicated in the table below:

System File	Command	PF Key	Objects/Data Affected
FUSER, FNAT	SET FNAT	PF12	- programming objects - debug environments - error messages - profiles
FDIC	SET FDIC	PF11	- rules - DL/I subfiles - DDMs - XRef information
FSEC	SET FSEC	PF10	- Natural Security profile

The security window that appears for the specified system file looks similar to the example of the FUSER and FNAT system files below:

```

+-----+
! --- Security for the Natural System Files --- !
!                                             !
! Specify the password(s), cipher(s) and VSAM FCT !
! name(s) for the source/target file(s) below: !
!                                             !
!   - Source -                               - Target - !
! Library .... OLDLIB                       Library ... NEWLIB !
! Database ... 10                           Database ... 10    !
! File ..... 32                             File ..... 32    !
!                                             !
! Password ...                               Password ...    !

```

```
! Cipher ..... Cipher ..... !
! VSAM Name .. _____ VSAM Name .. _____ !
+-----+-----+-----+
```

- 2 In the window, enter the appropriate password(s), cipher code(s) and/or VSAM name for the required source and/or target system file.



**Note:** The **Library** field is applicable only when processing programming objects, debug environments or error messages.

Once file security is defined, the SYSMAIN utility uses this security information for all subsequent processing. If you then require that the default security information (obtained at the initialization of the session) be used, you must re-invoke the corresponding security window and clear the password, cipher code and/or VSAM name fields. The passwords and cipher codes are non-display, so even though the fields *appear* to be empty, they should be cleared again.

## Natural Security

Two aspects must be considered when using the SYSMAIN utility within a Natural Security environment:

- [Defining the Natural Security Environment](#)
- [Restricting Use of SYSMAIN under Natural Security](#)

### Defining the Natural Security Environment

The source and target libraries can be within one Natural Security environment or within two different Natural Security environments. These environments must be defined to the SYSMAIN utility.

The definition of the Natural Security environment(s) to be used is specified with the special command `SET FSEC`.

By default, the current FSEC settings assigned at the start of the Natural session are used. If you change these settings (in the window **Security for Natural Security (FSEC) Files**), they remain in effect until they are changed by the next `SET FSEC` process. In batch mode or direct command mode, the `SEC` keyword should be used to specify the file security and assignments of the request.

Once the source and target environments have been determined, SYSMAIN verifies both the source and target libraries with Natural Security. (The source and/or target database and file must correspond to the database ID (DBID) and file number (FNR) specified in the library security profile; if these values are not specified, default values are taken from the security profile.)

## Restricting Use of SYSMAIN under Natural Security

The use of the SYSMAIN utility itself can be restricted, or the use of the source and target libraries to be handled with the SYSMAIN utility can be restricted. The use of SYSMAIN functions when invoked via the application programming interface MAINUSER can be controlled separately. See *Protecting Utilities* in the *Natural Security* documentation for details.

## User Exit Routines

---

The user exit routines of the SYSMAIN utility are used to provide information on each object being processed or control function processing. A user exit routine is a Natural subprogram, which is invoked with a `CALLNAT` statement.

The source codes of the subprograms and the data areas they use are stored as source objects under the names `SM-UX-nn` (*nn* = 01 to 11) in the library SYSMAIN. To make a user exit routine available, you have to catalog the corresponding source object under the name `MAINEXnn`, either in the library SYSMAIN or in one of its steplibs.



**Note:** The names of source objects and cataloged objects of user exit routines are different to ensure that the overwriting of the source objects by an update installation does not affect the cataloged objects.

You can change or expand any of the user exit routines as necessary.

Use of these exits results in additional overhead to the SYSMAIN utility, depending on the code logic. It is necessary, however, always to return control to SYSMAIN when exit processing is completed.

As the SYSMAIN utility uses ET logic with Adabas files, the use of user exit routines can lengthen the transaction time limit (Adabas parameter TT). Furthermore, the definition of the Adabas transaction should not be altered, which means that you should not issue any ET/BT commands or `END/BACKOUT TRANSACTION` statements. SYSMAIN is responsible for the issuing of all `END TRANSACTION` statements. The exception to this rule is in a situation where a user terminates the normal completion of any SYSMAIN function with the user exit routines. If this is the case, you must issue a `BACKOUT TRANSACTION` before terminating.

If the return code is set to a non-zero value, this overrides any error given by SYSMAIN. When an error is received from an exit, it is placed in the message field and displayed or printed as appropriate. The exception is automated processing, because processing is completed with minimum terminal I/O.

The individual user exit routines are described in the following section:

- [MAINEX01 - First User Exit Routine for Object Interrogation](#)
- [MAINEX02 - Second User Exit Routine for Object Interrogation](#)

- MAINEX03 - User Exit Routine for Request Interrogation
- MAINEX04 - User Exit Routine for Modification of File Assignments
- MAINEX05 - User Exit Routine for Verification of Direct Commands
- MAINEX06 - User Exit Routine for SYSMAIN Initialization
- MAINEX07 - User Exit Routine for SYSMAIN Termination
- MAINEX08 - User Exit Routine for Nothing Found in Batch Mode
- MAINEX09 - User Exit Routine for Abnormal Termination in Batch Mode
- MAINEX10 - User Exit Routine for Command Errors in Batch Mode
- MAINEX11 - User Exit Routine for Setting Special Flags to SYSMAIN

### MAINEX01 - First User Exit Routine for Object Interrogation

Function	Interrogate the current value settings of the data elements associated with an object <i>before</i> the object is processed by SYSMAIN.
Remarks	<p>Any object passed to MAINEX01 can be rejected by setting the RESP-CODE parameter to a non-zero value.</p> <p>If any additional logic is to be performed, the transaction may <i>not</i> be at end-of-transaction status and so no END TRANSACTION or BACKOUT TRANSACTION statement should be issued.</p> <p>Control must be returned to SYSMAIN.</p>
Parameters	<p>PARM-AREA1 (A250) SYSMAIN parameter area (fixed values).</p> <p>PARM-AREA2 (A250) SYSMAIN parameter area (variable values).</p> <p>RESP-CODE (B1) Response code to be returned to SYSMAIN.</p> <p><b>Note:</b> Only the RESP-CODE parameter can be modified.</p>
Local Data Area	SM-UX-L

### MAINEX02 - Second User Exit Routine for Object Interrogation

Function	Interrogate the current value settings of the data elements associated with an object <i>after</i> the object has been processed by SYSMAIN.
Remarks	<p>Any object passed to MAINEX02 can be rejected by setting the RESP-CODE parameter to a non-zero value.</p> <p>If any additional logic is to be done, the transaction may <i>not</i> be at end-of-transaction status and so no END TRANSACTION or BACKOUT TRANSACTION statement should be issued.</p> <p>Control must be returned to SYSMAIN.</p>
Parameters	<p>PARM-AREA1 (A250) SYSMAIN parameter area (fixed values).</p> <p>PARM-AREA2 (A250) SYSMAIN parameter area (variable values).</p>

	<p>RESP-CODE (B1)      Response code to be returned to SYSMAIN.</p> <p><b>Note:</b> Only the RESP-CODE parameter can be modified.</p>
Local Data Area	SM-UX-L

**MAINEX03 - User Exit Routine for Request Interrogation**

Function	Interrogate any request made to SYSMAIN in terms of a direct command or information entered online in menu mode. MAINEX03 obtains control <i>before</i> SYSMAIN processes the command.
Remarks	<p>Any command passed to MAINEX03 can be rejected by setting the RESP-CODE parameter to a non-zero value.</p> <p>Additional logic can be added, but it is your responsibility to issue any necessary END TRANSACTION requests to the database.</p> <p>Control must be returned to SYSMAIN.</p>
Parameters	<p>PARM-AREA (A250) Command string.</p> <p>RESP-CODE (B1)      Response code to be returned to SYSMAIN.</p> <p><b>Note:</b> Only the RESP-CODE parameter can be modified.</p>

**MAINEX04 - User Exit Routine for Modification of File Assignments**

Function	Override the database, file, password and cipher codes for the Natural system file(s).
Remarks	<p>MAINEX04 is invoked <i>before</i> any request is processed or validated by SYSMAIN. When control is passed to MAINEX04, you are at end-of-transaction status; therefore you have to set the RESP-CODE parameter to a non-zero value if you wish to reject the request.</p> <p>Control must be returned to SYSMAIN.</p>
Parameters	<p>PARM-AREA (A250) SYSMAIN parameter area.</p> <p>RESP-CODE (B1)      Response code to be returned to SYSMAIN.</p>
Local Data Area	SM-UX-L4

**MAINEX05 - User Exit Routine for Verification of Direct Commands**

Function	Verify any direct command entered during online processing of SYSMAIN. In addition, the special characters used to indicate a system command can be overwritten.
Remarks	<p>MAINEX05 is invoked <i>before</i> any direct command issued within SYSMAIN is processed. For example, MAINEX05 enables you to interrogate any of the <b>SET commands</b> (see <i>Special Commands Issued to SYSMAIN</i>) and also prevent them from being issued. You can verify these commands and reject them by returning a non-zero value in the RESP-CODE parameter. You are at end-of-transaction status when control is passed to MAINEX05.</p> <p>A system command entered within SYSMAIN has to be preceded by two slashes (/); see also <i>Using the SYSMAIN Command Line</i>. With MAINEX05, you can define two other special characters for this purpose; to do so, you assign the desired characters to the CMD-DEL parameter. If CMD-DEL is set to blanks, SYSMAIN uses the default value of two slashes (/). Control must be returned to SYSMAIN.</p>
Parameters	<p>COMMAND (A68) Current command issued in SYSMAIN.</p> <p>CMD-DEL (A3) Special character for system commands.</p> <p>RESP-CODE (B1) Response code to be returned to SYSMAIN.</p>

**MAINEX06 - User Exit Routine for SYSMAIN Initialization**

Function	Obtain control at initialization of a SYSMAIN session.
Remarks	<p>MAINEX06 is invoked at the start of the SYSMAIN session, where you can override some of the SYSMAIN default settings, as for example, prompts for confirmation of a request like deleting, moving or replacing an object.</p> <p>All parameters are verified. If they are invalid, the default settings are used.</p> <p>Control must be returned to SYSMAIN.</p>
Parameter Data Area	SM-UX-L6

**MAINEX07 - User Exit Routine for SYSMAIN Termination**

Function	Obtain control at termination of a SYSMAIN session.
Remarks	MAINEX07 is invoked at termination of a SYSMAIN session to decide whether control is to be kept by SYSMAIN or not.
Parameters	<p>USER-AREA (A50) Area for free usage.</p>

**MAINEX08 - User Exit Routine for Nothing Found in Batch Mode**

Function	Determine further processing if no objects are found for a command in batch mode.
Remarks	MAINEX08 is invoked if no objects are found that meet the specified criteria for a specific command executed in batch mode. If this is the case, control may, but need not, be returned to SYSMAIN. If control is returned to SYSMAIN, SYSMAIN will continue processing with the next command.
Parameters	CMD (A250) Command string.

**MAINEX09 - User Exit Routine for Abnormal Termination in Batch Mode**

Function	Determine action to be taken in case of error in batch mode.
Remarks	MAINEX09 is invoked if SYSMAIN processing in batch mode leads to an error. If this is the case, control may, but need not, be returned to SYSMAIN. If control is returned to SYSMAIN, SYSMAIN will be terminated with condition code 45.  <b>Note:</b> Errors NAT4810, NAT4818, NAT4867, NAT4868 and NAT4893 cannot be handled by this user exit routine.
Parameters	CMD (A250) Command string. ERROR-CODE (N4) Number of error which caused termination.

**MAINEX10 - User Exit Routine for Command Errors in Batch Mode**

Function	Determine action to be taken in case of command error in batch mode.
Remarks	MAINEX10 is invoked if an error is detected in a SYSMAIN command in batch mode. If this is the case, control may, but need not, be returned to SYSMAIN. If control is returned to SYSMAIN, SYSMAIN will continue processing with the next command.
Parameters	CMD (A250) Command string. ERROR-CODE (N4) Number of error which caused termination.

**MAINEX11 - User Exit Routine for Setting Special Flags to SYSMAIN**

Function	Special settings user exit routine.
Remarks	MAINEX11 is invoked at the start of the SYSMAIN session, where you can set some special SYSMAIN flags, as for example, display of MAINUSER messages in batch.  See the source object of the user exit routine (SM-UX-11) for the available flags.  Control must be returned to SYSMAIN.
Parameters	FLAGS (A250) Flag string (redefined).



# 100

## SYSNCP Utility

---

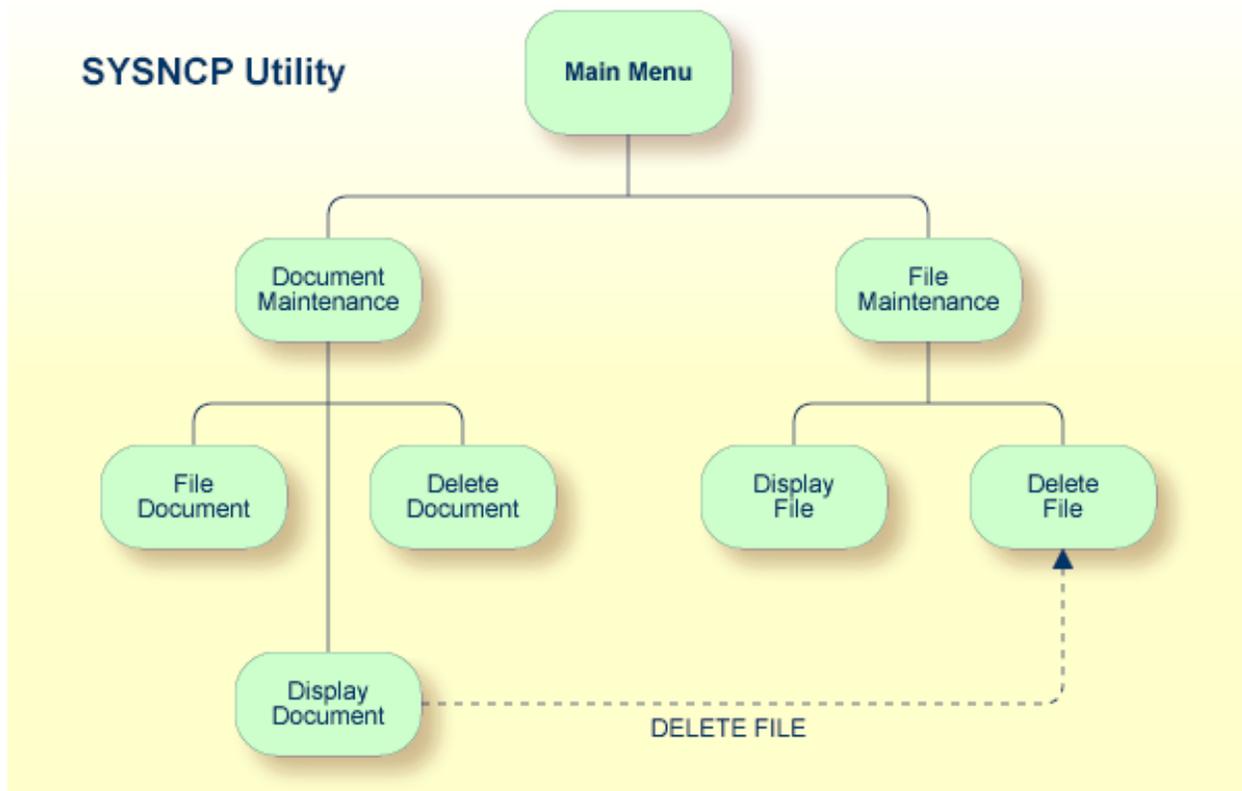
▪ Introducing the SYSNCP Utility .....	624
▪ Invoking SYSNCP .....	631
▪ Processor Selection .....	632
▪ Header Records .....	633
▪ Keyword Maintenance .....	642
▪ Function Maintenance .....	647
▪ Runtime Actions .....	652
▪ Processor Cataloging .....	657
▪ Administrator Services .....	658
▪ Session Profile .....	665

The utility SYSNCP is used to define command-driven navigation systems for Natural applications.

The Natural Command Processor (NCP) consists of two components: maintenance and runtime. The utility SYSNCP is the maintenance part which comprises all facilities used to define and control navigation within an application. The PROCESS COMMAND statement (see the *Statements* documentation) is the runtime part used to invoke Natural programs.

## Introducing the SYSNCP Utility

Applications which enable users to move from one activity to another activity by using direct commands far exceed in usability the ones which force the user to navigate through menu hierarchies to a desired activity.



The figure above illustrates the advantage of using direct commands. In an application in which menu hierarchies form the basis for navigation, a user wishing to advance from the Display Document facility to the Delete File facility would have to return to the Main Menu via the document branch and then enter the file branch. This is clearly less efficient than accessing the Delete File facility directly from the Display Document facility.

Below is information on:

- [Object-Oriented Data Processing](#)
- [Features of the Command Processor](#)
- [Components of the Command Processor](#)
- [What is a Command?](#)
- [Creating a Command Processor](#)

## Object-Oriented Data Processing

The Natural command processor is used to define and control navigation within an application. It could be used, for example, to define a command `DISPLAY DOCUMENT` to provide direct access to the Display Document facility. When a user enters this command string in the Command line of a screen (for which this command is allowed), the Natural command processor processes the input and executes the action(s) assigned to the command.

In contrast to menu-driven applications, the command-driven applications implemented with the Natural command processor take a major step toward object-oriented data processing. This approach has the following advantages:

- The design of an application need not depend on the way in which a certain result can be reached, but only on the desired result itself. Thus, the design of an application is no longer influenced by the process flow within its components.
- The processing units of an application become independent of one another, making application maintenance easier, faster and much more efficient.
- Applications can be easily expanded by adding independent processing units. The resulting applications are, therefore, not only easy to use from an end-user's view, but also easier to create from a programmer's view.

The Natural command processor has the following additional benefits:

- **Less Coding**  
Instead of having to repeatedly program lengthy and identically structured statement blocks to handle the processing of commands, you only have to specify a `PROCESS COMMAND` statement that invokes the command processor; the actual command handling need no longer be specified in the source code. This considerably reduces the amount of coding required.
- **More Efficient Command Handling**  
As the command handling is defined in a standardized way and in one central place, the work involved in creating and maintaining the command-processing part of an application can be done much faster and much more efficiently.
- **Improved Performance**  
The Natural command processor has been designed with particular regard to performance aspects: it enables Natural to process commands as fast as possible and thus contributes to improving the performance of your Natural applications.

## Features of the Command Processor

The Natural command processor provides numerous features for efficient and user-friendly command handling:

### ■ Flexible Handling of Commands

You can define aliases (that is, synonyms for keywords), and abbreviations for frequently used commands.

### ■ Automatic Check for Uniqueness of Abbreviated Keywords

The command processor automatically compares every keyword you specify in SYSNCP with all other keywords and determines the minimum number of characters in each keyword required to uniquely identify the keyword. This means that, when entering commands in an application, users can shorten each keyword to the minimum length required by the command processor to distinguish it from other keywords.

### ■ Local and Global Validity of Commands

You can specify in SYSNCP whether the action to be performed in response to a specific command is to be the same under all conditions or situation-dependent. For example, you can make the action dependent on which program was previously issued. In addition, you can define a command to be valid under one condition but invalid under another.

### ■ Error Handling for Invalid Commands

You can attach your own error-handling routines to commands or have error input handled by Natural.

### ■ Functional Security

With Natural Security, library-specific and user-specific conditions of use can be defined for the tables generated with SYSNCP. Thus, for your Natural applications you can allow or disallow specific functions or keywords for a specific user. This is known as functional security. See also the section *Functional Security* in the *Natural Security* documentation.

### ■ Help Text

In SYSNCP, you can attach help text to a keyword or a command. Then, by specifying a PROCESS COMMAND ACTION TEXT statement, you can return command-specific help text to the program.

### ■ Online Testing of Command Processing

If the execution of a command does not produce the intended result, you can find out why the command was not processed correctly by using the PROCESS COMMAND statement (see the *Statements* documentation) and the EXAM\* sample test programs (source form) provided in the Library SYSNCP. The endings of the EXAM-\* program names appear as abbreviations at the top border line of the relevant action windows (for example, EXAM-C appears as C).

#### ▶ To test a command processor at runtime

- 1 Enter the direct command EXAM to list all test programs. The **Demonstrate PROCESS COMMAND Statement** window is displayed.

- 2 Enter Function Code **O** to open a processor.
- 3 Enter the name of the processor.
- 4 Choose any of the Functions Codes listed (for example, C for CHECK) to apply command actions.
- 5 Enter Function Code **Q** to close the processor.

## Components of the Command Processor

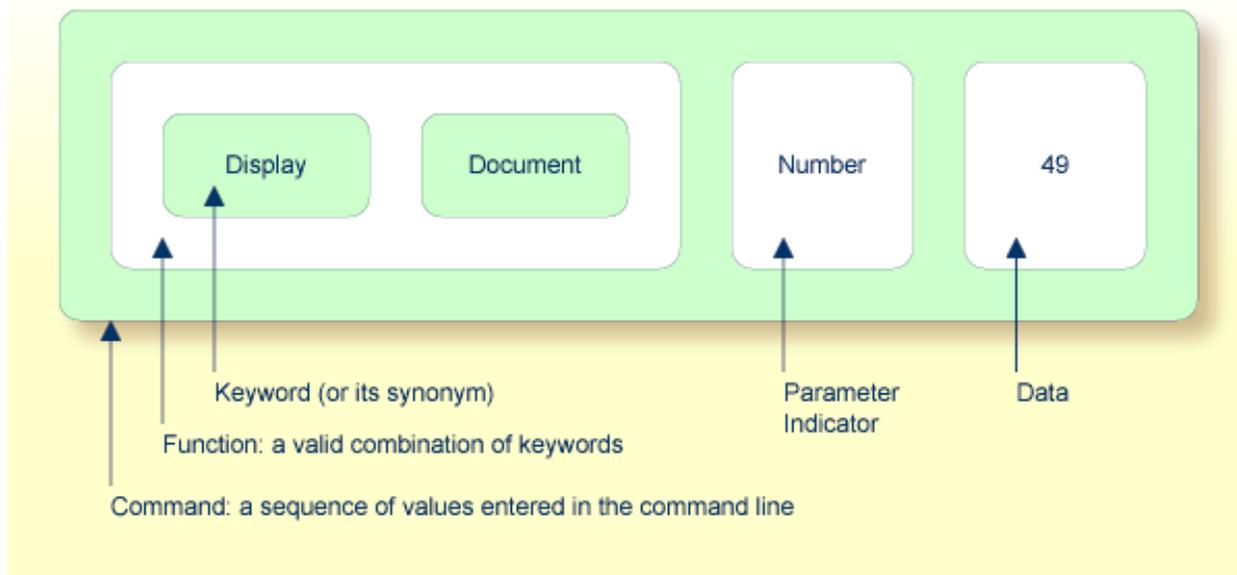
The Natural command processor consists of two parts: a development part and a runtime part:

- The development part is the utility SYSNCP, which is described in this section. With the utility SYSNCP you define commands (as described below) and the actions to be performed in response to the execution of these commands. From your definitions, SYSNCP generates decision tables which determine what happens when a user enters a command. These tables are contained in a Natural member of type Processor.
- The runtime part is the statement PROCESS COMMAND, which is described in the *Statements* documentation. This statement is used to invoke the command processor within a Natural program. In the statement, you specify the name of the processor to be used to handle the command input by a user at this point.

## What is a Command?

A command is any sequence of values entered in the Command line which is recognized and processed by an application. Commands can contain up to three elements:

- **Function:**  
One or more valid keywords. For example, MENU or DISPLAY DOCUMENT.
- **Parameter Indicator:**  
Optional. A keyword which introduces command data.
- **Command Data:**  
Information to be sent to a function. Command data can be alphanumeric or numeric, for example, the name or the number of the file to be displayed.



Commands are always executed from a situation within an application; the position where this situation is reached is referred to as a location. Commands take the user from one location to another location; thus, each command can be viewed as a vector:

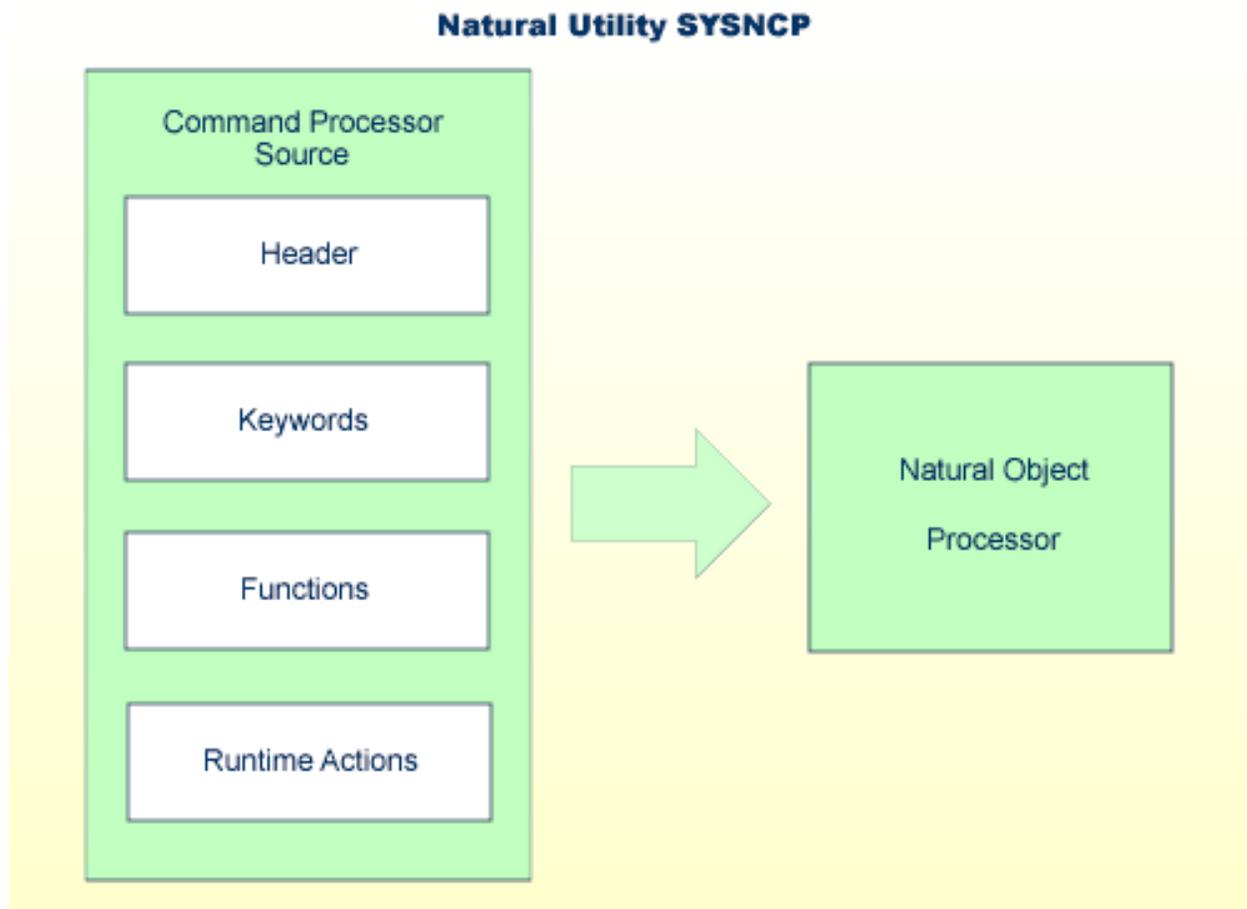


The location from which a certain command can be issued can be restricted on a system-wide or user-specific basis. On a system-wide basis, for example, the functions specified within commands can be local or global. A global function can be issued from *any* location while a local function can only be issued from specified locations. Restrictions can be placed on keywords and functions, however, if Natural Security is active in your environment.

## Creating a Command Processor

The utility SYSNCP is used to create and maintain command processors. A command processor contains decision tables which determine what happens when a user enters a valid command.

The creation of a command processor is a cumulative operation involving several steps, from header definition, which establishes general defaults for the processor, to keyword definition, function definition and the linking of actions to functions. Special editors are provided by SYSNCP for the purpose of specifying keywords, functions and actions.



The end product of command processor development is a complex command processor source, which, when cataloged, generates a Natural object of type Processor. Whenever this object is referenced by the Natural statement `PROCESS COMMAND`, the runtime system of the Natural command processor is triggered.

The following is a summary of the steps necessary to create a command processor.

▶ **To create a command processor**

1 **Verify/Modify the Session Profile.**

SYSNCP itself uses a Session Profile which contains various parameters which control how SYSNCP is to perform certain actions and how information is to be displayed. Desired modifications can be made and the resulting profile can be saved with a given user ID. See the section [Session Profile](#).

2 **Initialize the Command Processor.**

The name of the command processor and the library into which it is to be stored are specified.

3 **Define Global Settings (Header).**

Various global settings for the command processor are defined. For example, descriptive text for keywords during editing, minimum and maximum length for keywords, in which sequence keywords are to be processed at runtime, runtime error-handling, and whether PF keys can be used at runtime to invoke functions. See the section [Header Records](#).

4 **Define Keywords.**

Each keyword which is to be processed by the command processor is defined together with an indication as to whether the keyword is to be entered as the first, second or third entry of a command. Keyword synonyms can also be defined as well as parameter indicators. User text can be defined for each keyword. This text can subsequently be read at runtime using the PROCESS COMMAND ACTION TEXT statement. See the section [Keyword Maintenance](#).

5 **Define Functions.**

Functions are defined by validating keyword combinations. A function can be defined as local (can only be invoked from a specific location within an application) and/or global (can be invoked from anywhere within an application). See the section [Function Maintenance](#).

6 **Define Runtime Actions.**

The actions to be taken by the command processor when a command is issued at runtime are specified. Example actions are: fetch a Natural program, place a command at the top of the Natural stack, place data at the top of the Natural stack, change contents of the Command line. See the section [Runtime Actions](#).

7 **Catalog Command Processor.**

The resulting source is cataloged as a Natural object (type Processor) in the designated Natural library. The command processor can now be invoked by a Natural program using the PROCESS COMMAND statement. See the section [Processor Cataloging](#).

## Invoking SYSNCP

### ▶ To invoke the SYSNCP utility

- Enter the system command SYSNCP.

The Processor Source Maintenance menu is displayed:

```

18:22:53          ***** NATURAL SYSNCP UTILITY *****          2000-05-22
User SAG          - Processor Source Maintenance -

Code  Function

S     Select Processor
N     Create New Processor
H     Modify Header
K     Define Keywords
F     Define Functions
R     Define Runtime Actions
C     Catalog Processor
A     Administrator Services
?     Help
.     Exit

Code .. _      Name .. SAGTEST_  Library .. SYSNCP__

Logon to SYSNCP accepted.
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Cmd  Exit  Last  List  Flip                                Canc

```

From this menu, you can invoke all functions necessary to create and maintain a command processor. To invoke a function, enter the code letter in the Code field.



**Note:** When you invoke the SYSNCP utility or restart SYSNCP, the user exit NCP-USR1 is invoked for dynamic customization purposes: see the program NCP-USR1 delivered in the Natural system library SYSNCP.

## Help

For help on individual input fields (and also on some output fields) in SYSNCP, place the cursor on the field and press PF1.

## Processor Selection

The Select Processor function results in a list of all existing command processor sources with related information. If Natural Security is installed, only those sources are listed which can be cataloged to a library to which you are allowed to log on. These restrictions do not apply to those users who have administrator status.

### ▶ To invoke the Select Processor function

- 1 In the Processor Source Maintenance menu, enter Function Code S.
- 2 Press ENTER.

The following information is provided for each processor:

Name	The name of the command processor.
Library	The name of the Natural library for which a processor is created. When the processor is cataloged, it is stored in this library.
User ID	The ID of the user who created the processor.
Date	The date the processor was created.
Status	The stage of development of the processor. For possible status values, see <a href="#">Current Status</a> in the section <i>Header Records</i> .
Cat	Indicates if the processor has been cataloged.



**Note:** With the user exit NCP-SELX (delivered in the Natural system library SYSNCP), you can limit the display to certain processors.

- 3 In the **Ac** field, enter any character to select a processor.

The Processor Source Maintenance menu is displayed, where the name of the selected processor is automatically placed in the Name field.

If you enter a question mark (?) in the Ac field, a window is displayed, listing other possible options.

The name and library name of a command processor can be one to eight characters long. It can consist of upper-case alphabetical characters (A - Z), numeric characters (0 - 9) and the special characters: "-", "/", "\$", "&", "#", "+" and "\_".

---

## Header Records

---

The header maintenance facility defines various global settings for a command processor. These definitions are collectively referred to as a header. Seven header maintenance screens are provided for creating and modifying headers. Header settings for a command processor can be updated at any stage of development (see the following section). After the settings have been modified, the status of a command processor is always set to Header (see also *Current Status*).

Below is information on:

- [Create New Processor](#)
- [Modify Header - General Explanations](#)
- [Keyword Runtime Options - Header 1](#)
- [Keyword Editor Options - Header 2](#)
- [Miscellaneous Options - Header 3](#)
- [Command Data Handling - Header 4](#)
- [Runtime Error Handling - Header 5](#)
- [Statistics - Header 6](#)
- [Status - Header 7](#)

### Create New Processor

▶ **To create a new command processor**

- 1 In the Processor Source Maintenance menu, enter Function Code **N** (Create New Processor), the name of the command processor to be created, and the name of the Natural library in which the command processor is to be later cataloged.
- 2 Press ENTER.

The first header maintenance screen is displayed.

The first header maintenance screen and the following ones are filled with default values that can be edited.

## Modify Header - General Explanations

The Modify Header function is used to maintain an existing header; that is, to modify the various header settings for a given command processor.

### ▶ To modify an existing header

1 In the Processor Source Maintenance menu, enter Function Code **H** (Modify Header), the name of the corresponding command processor, and the name of the library into which this command processor has been cataloged.

2 Press ENTER.

The first header maintenance screen is displayed.

3 Modify any input field in the header maintenance screens described below.

4 Press ENTER to confirm modifications.

Seven different screens are available for the definition and maintenance of a processor header (for the definition of a header, see the previous section).

### ▶ To navigate between the header maintenance screens

■ Use PF8 (forward) or PF7 (backward).

Each of the screens contains the following information:

Name	The name of the command processor.
Library	The name of the library into which the resulting command processor object is to be placed after being cataloged.
DBID, FNR	The database ID and file in which the specified library is located.
Created by	The user ID of the Natural user who initialized this command processor.
Date	The date the command processor was initially created.
Current Status	The command processor status: <ul style="list-style-type: none"> <li>Init        The command processor has been initialized.</li> <li>Header     The header for the command processor has been created/modified.</li> <li>Keysave    Keywords have been defined and saved.</li> <li>Keystow    Keywords have been checked and stowed.</li> <li>Function    Keyword combinations have been defined.</li> <li>Action     Runtime actions have been defined.</li> <li>Object     An object form of the command processor has been created.</li> <li>Frozen     The command processor has been frozen.</li> </ul>

	Copied     The command processor has been copied.
	Error        An error has been detected.

### Keyword Runtime Options - Header 1

When you select the Modify Header function (as described above), the **Processor Header Maintenance 1** screen is displayed:

```

16:40:19          ***** NATURAL SYSNCP UTILITY *****          2000-05-04
User SAG          - Processor Header Maintenance 1 -

Modify Processor          Name SAGTEST  Library SYSNCP  DBID 10   FNR 32
Created by SAG          Date 2000-04-29          Current Status Init

Keyword Runtime Options:
-----
First Entry used as ..... Action_____
Second Entry used as ..... Object_____
Third Entry used as ..... Addition_____

Minimum Length ..... _1
Maximum Length ..... 16
Dynamic Length Adjustment .. -

Keyword Sequence ..... 123_____
Alternative Sequence ..... _____
Local/Global Sequence ..... LG_____

Processor Header with name SAGTEST for library SYSNCP has been added.
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Cmd  Exit  Last  List  Flip  -    +          Canc

```

Various attributes which are to apply for the keywords defined for the command processor are entered on this screen.

Field	Explanation
First Entry used as	<p>A descriptive text which is to be associated with all keywords which are entered as the first entry (entry type 1) when defining a keyword sequence.</p> <p>For example, if the first keyword of a keyword sequence is to represent the action to be performed (DISPLAY, DELETE, etc.), the descriptive text "Action" could be entered in this field.</p> <p>The first four characters of the text entered in this field appear under the column heading <b>Use</b> in the Keyword Editor as described in the section <i>Keyword Maintenance</i>.</p>
Second Entry used as	<p>A descriptive text which is to be associated with all keywords which are entered as the second entry (entry type 2) when defining a keyword sequence.</p> <p>If, for example, the second keyword of a keyword sequence is to represent the object to be used (DOCUMENT, FILE, etc.), the descriptive text "Object" could be entered in this field.</p> <p>The first four characters of the text entered in this field appear under the column heading <b>Use</b> in the Keyword Editor as described in the section <i>Keyword Maintenance</i>.</p>
Third Entry used as	<p>A descriptive text (TITLE, PARAGRAPH, etc.) which is to be associated with all keywords which are entered as the third entry (entry type 3) when defining a keyword sequence.</p> <p>The first four characters of the text entered in this field appear under the column heading <b>Use</b> in the Keyword Editor as described in the section <i>Keyword Maintenance</i>.</p>
Minimum Length	The minimum length permitted when defining a keyword. Valid values are 1 - 16 characters. The default is one character.
Maximum Length	The maximum length permitted when defining a keyword. Valid values are 1 - 16 characters. The default is 16 characters.
Dynamic Length Adjustment	<p>The following values are permitted:</p> <ul style="list-style-type: none"> <li>+ At runtime, each keyword must be entered in its entirety.</li> <li>- At runtime, each keyword can be abbreviated provided that it retains uniqueness with respect to other keywords.</li> <li>S The number of characters which must be entered for a given keyword is to be specified during keyword definition in the <b>ML field</b> of the Keyword Editor as described in the section <i>Keyword Maintenance</i>.</li> </ul>
Keyword Sequence	The sequence in which keyword entries are to be processed at runtime. Possible values are 1, 2, 3 and P (for parameter indicator); the default sequence is 12, which means first the first keyword entry and then the second keyword entry. See also the <b>field E</b> as described in the section <i>Keyword Maintenance</i> .
Alternative Sequence	An alternative sequence in which keywords are to be processed at runtime in the event that the default sequence (specified above) results in an error during runtime.

Field	Explanation
Local/Global Sequence	<p>This option specifies the order of command validation to be performed at runtime. Possible values are:</p> <p>L Command is to be validated as a local command.</p> <p>G Command is to be validated as a global command.</p> <p>The default validation sequence is LG, which means that the command is to be validated first as a local command and then (if necessary) as a global one.</p>

## Keyword Editor Options - Header 2

Further keyword attributes can be entered on the **Processor Header Maintenance 2** screen:

Field	Explanation
Header 1 for User Text	These two fields are used to enter a descriptive text which appears in the Keyword Editor above the column reserved for user text. This text is also output during runtime when the TEXT option is specified with the PROCESS COMMAND statement as described in the <i>Statements</i> documentation.
Header 2 for User Text	
Prefix Character 1	<p>This field and the next three are used to attach a hexadecimal prefix to keywords. This enables the processing of internal keywords which cannot be represented by a normal keyboard. When the command processor is cataloged, all prefix characters in keywords are replaced by the hexadecimal values specified.</p> <p>If a non-blank character is entered in one of the Prefix Character fields, the specified character is replaced by the hexadecimal value specified in the Hexadecimal Replacement field.</p>
Hex. Replacement 1	The value specified in this field replaces the character specified in the field Prefix Character and is used as a prefix for a keyword at runtime.
Prefix Character 2	See above Prefix Character 1.
Hex. Replacement 2	See above Hex. Replacement 1.
Keywords in Upper Case	<p>This option specifies whether keywords are to be translated to upper case in the Keyword Editor and the application:</p> <p>Y Keywords entered in the Keyword Editor are automatically converted to upper case. In the application, end-users can enter the keywords in upper or lower case.</p> <p>N Keywords entered in the Keyword Editor are not converted to upper case. In the application, end-users must enter the keywords <i>exactly</i> as they appear in the Keyword Editor.</p>
Unique Keywords	<p>This option specifies whether keywords within the processor must be unique.</p> <p>Y Each keyword defined must be unique within this processor, regardless of its type.</p>

Field	Explanation
	N Each keyword defined for a given keyword type (1, 2, 3 or P) must be unique.

### Miscellaneous Options - Header 3

Miscellaneous options can be entered on the **Processor Header Maintenance 3** screen:

Field	Explanation
Invoke Action Editor	<p>This option specifies whether the Runtime Action Editor is to be activated from the Function Editor (see the sections <i>Runtime Action Editor</i> and <i>Define Functions</i>).</p> <p>Y The Runtime Action Editor is invoked whenever a valid keyword combination is defined in the Function Editor.</p> <p>N The Runtime Action Editor is suppressed in the Function Editor.</p> <p><b>Note:</b> If you use the user exit NCP-REDM (delivered in the Natural system library SYSNCP), you should set this option to Y; otherwise, invalid runtime action values cannot be detected in time and can lead to runtime errors.</p>
Catalog User Texts	<p>This option specifies whether user texts are to be cataloged with the command processor.</p> <p>Y Text portions of the edit line (Keyword Editor; see the section <i>Define Keywords</i>) and the user text portion of the action line (Runtime Action Editor) are bound to the associated keyword or function when the command processor is cataloged. This text can then be read at runtime using the TEXT option of the PROCESS COMMAND statement.</p> <p>N Texts are not cataloged with the command processor and cannot be read at runtime.</p>
Security Prefetch	<p>This option specifies whether security checking is to be performed when the command processor is initially invoked during runtime or at each command evaluation.</p> <p>Y If Natural Security is installed, security checking is performed for all keywords when the processor is invoked.</p> <p>N If Natural Security is installed, security checking is performed with the evaluation of each keyword.</p> <p>If option Y is selected, security checking is performed only once for all keywords when the command processor is invoked. Since the checking procedure takes time, evaluation of the first command is comparatively slow at runtime, while the evaluation of all remaining commands is comparatively fast. Conversely, if option N is selected, the evaluation time for each command is always the same because security is checked for each keyword individually before it is evaluated.</p>

Field	Explanation
Command Log Size	<p>Commands processed at runtime can be stored in a command log area by the command processor. Specify in the input field the number of KBs storage space allocated to command logging:</p> <p>0 No storage space is allocated to command logging. Command logging is inactive.</p> <p>1 1 KB of storage space is allocated to command logging. Command logging is active.</p>
Implicit Keyword Entry	<p>This option specifies whether a keyword of type 1 is to be retained as an implicit keyword for all subsequent commands.</p> <p>1 If a command is entered which only contains a keyword of type 2, the command processor assumes the most recently entered keyword of type 1 as implicit keyword.</p> <p>N Option is disabled.</p>
Command Delimiter	<p>This option specifies the character used to separate commands if more than one command is specified in the Command line. At runtime, only the first command will be executed.</p> <p>For example:</p> <p>DISPLAY CUSTOMER; MODIFY CUSTOMER; PRINT.</p>
PF-Key may be Command	<p>This option specifies whether commands can be allocated to PF keys: if the command processor receives at runtime a command line which contains all blanks, it checks if a PF key has been pressed by the user.</p> <p>Possible values are:</p> <p>A The identifier for this PF key (system variable *PF-NAME) is used as the command.</p> <p>K The content of the *PF-KEY system variable is used as the command.</p> <p>Y If *PF-NAME is empty, the content of the *PF-KEY system variable is used instead.</p> <p>N PF keys cannot be used as command, Natural error NAT6913 is issued with message "Command line not accepted".</p> <p>For more information on the system variables *PF-NAME and *PF-KEY see the <i>System Variables</i> documentation.</p>

#### Command Data Handling - Header 4

The attributes to be entered on the **Processor Header Maintenance 4** screen specify how command data are handled for a function; command data are optional.

Options are:

Field	Explanation
Data Delimiter	<p>Specifies the character to be used to precede data. Default data delimiter is "#".</p> <p>Example: ADD CUSTOMER #123</p>
Data Allowed	<p>Specifies if data input is allowed at runtime.</p> <p>N A runtime error occurs if data is found.</p> <p>D Data is dropped if present.</p> <p>S Data is placed at the top of the Natural stack. No verification is performed.</p> <p>Y Data is checked and keyword entries of type P (parameter indicator) are evaluated.</p> <p>Example of Y: DISPLAY CUSTOMER NAME=SMITH</p>
More than one Item Allowed	<p>Only applies if the option Data Allowed is set to Y. Specifies whether more than one data string is permitted.</p> <p>N A runtime error occurs if more than one data string is found.</p> <p>D All data after the first data string are dropped.</p> <p>Y More than one data string is permitted.</p> <p>Example: ADD ARTICLE #111 #222</p> <p>As long as uniqueness is guaranteed, the data delimiter can be omitted.</p> <p>Example: ADD ARTICLE 123</p>
Maximum Length of one Item	<p>Only applies if the option Data Allowed is set to Y.</p> <p>Specifies the maximum number of characters allowed for a data string. If the specified maximum is exceeded, a runtime error occurs. Valid range: 1 - 99.</p>
Item Must be Numeric	<p>Only applies if the option Data Allowed is set to Y. Specifies whether each data value must be an integer value.</p> <p>Y Data input must be a positive integer value. If not, a runtime error occurs.</p> <p>N Data can be of any type.</p>
Put to Top of Stack	<p>Only applies if the option Data Allowed is set to Y. Specifies where data is to be placed.</p> <p>Y Data is placed at the top of the Natural stack.</p> <p>1-9 Data is placed in the <i>n</i>th occurrence of the DDM field RESULT-FIELD. If the occurrence has already been filled as a result of a runtime action, it is overwritten.</p>
If Error, Drop all Data	<p>Only applies if the option Data Allowed is set to Y or N. Specifies the reaction to a data evaluation error:</p>

Field	Explanation
	<p>Y If an error occurs during evaluation of the data, data is discarded and processing continues.</p> <p>N If an error occurs during data evaluation, control is given to the error handler as described below.</p>

### Runtime Error Handling - Header 5

The attributes to be entered on the **Processor Header Maintenance 5** screen specify how to handle runtime errors:

Field	Explanation
General Error Program	<p>The name of the program which is to receive control when an error is detected during runtime processing by the command processor. The Natural stack contains the following information when this program is invoked:</p> <p>Error Number (N4)  Line Number (N4)  Status (A1)  Program Name (A8)  Level (N2)</p> <p>If no error program and no specific error handling is specified (see below), the program with the name as contained in the Natural system variable *ERROR-TA is invoked; otherwise, a Natural system error message is issued.</p>
Keyword not found	Indicates whether an action has been specified that is to be performed if a keyword could not be found.
Keyword missing	Indicates whether an action has been specified that is to be performed if the keyword type is missing.
Keyword Sequence Error	Indicates whether an action has been specified that is to be performed in the case of a keyword sequence error.
Command not defined	Indicates whether an action has been specified that is to be performed in the case of an undefined command.
Data disallowed	Indicates whether an action has been specified that is to be performed in the case of disallowed data.
Data Format/Length Error	Indicates whether an action has been specified that is to be performed in the case of a format/length error.
General Security Error	Indicates whether an action has been specified that is to be performed if an error is detected during a general security check.
Keyword Security Error	Indicates whether an action has been specified that is to be performed if an error is detected during a keyword security check.

Field	Explanation
Command Security Error	Indicates whether an action has been specified that is to be performed if an error is detected during a command security check.

### Statistics - Header 6

The **Processor Header Maintenance 6** screen contains only output fields which report statistical data about the keywords specified for a command processor.

The following statistical information is provided:

Field	Explanation
Entry $n$ Keywords	The number of keywords of type $n$ defined in the command processor (not including synonyms).
Entry $n$ Keywords + Synonyms	The sum of keywords of type $n$ and their assigned synonyms.
Highest IKN for Entry $n$	The largest Internal Keyword Number for the keyword of type $n$ .
Possible Combinations	The number of possible combinations for keywords defined.
Cataloged Functions	The number of keyword combinations currently cataloged.

### Status - Header 7

The **Processor Header Maintenance 7** screen contains only output fields which report the time and the date when parts of the command processor were executed or modified.

## Keyword Maintenance

---

Keywords are the basic components for defining functions. Before it is possible to define keywords, the header maintenance records must be created (see the section [Header Records](#)).

- [Define Keywords](#)
- [Editor Commands](#)
- [Positioning Commands](#)

- Line Commands

## Define Keywords

Keywords used in commands are created with the Define Keywords function and the Keyword Editor. The Keyword Editor is similar to existing Natural editors except that lines of the editor are broken up into separate fields. Most of the **editor commands** (see the relevant section) and the **line commands** (see the relevant section) which are used in the Natural program editor can also be used in the Keyword Editor.

### ▶ To invoke the Keyword Editor

- 1 In the Processor Source Maintenance menu, enter Function Code **K** (Define Keywords).
- 2 Press ENTER.

The Keyword Editor screen is displayed.

The Keyword Editor screen is shown below. Several keywords have already been defined to serve as examples for this section.

```

09:42:39                - SYSNCP Keyword Editor -                2000-05-04
Modify Keywords          Name SAGTEST  Library SYSNCP  DBID 10  FNR 32

I Line E Use  Keyword          IKN   ML Comment
-----
  1 1 Acti MENU              1004  1
  2 1 Acti DISPLAY          1002  2
  3 S Syno SHOW             1002  1
  4 1 Acti DELETE          1001  2
  5 S Syno PURGE           1001  1
  6 S Syno ERASE           1001  1
  7 1 Acti FILE            1003  4
  8 P Parm NAME            4002  2
  9 2 Obje FILE            2001  4
 10 P Parm NUMBER          4001  2
 11 2 Obje DOCUMENT        2003  2
 12 1 Acti INFORMATION     1005  1
 13
 14
----- All -----
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  Cmd  Exit  Last  List  Flip  -1   +1   Top  Bot  Info  Canc

```

Enter in the Keyword Editor all the keywords which you want to have in your command language. These can be entered in any order desired, except synonyms, which must immediately follow the keywords they are related to. To each keyword you assign a type which specifies to which part of command syntax the keyword belongs. Rules of command syntax for a command processor are specified in the processor header; see *Keyword Runtime Options - Header 1* in the section *Header Records*. For example, you can specify whether a keyword is to be of type 1 (entered in first position in a command), type 2, type 3, a synonym for another keyword or a parameter indicator.



**Note:** A command language requires a strict syntax because, to date, no computer is capable of understanding semantics. Word type is, therefore, the only practical way to communicate meaning in a command language.

In the example above, the keywords DELETE and DISPLAY are defined as keywords of type 1. As specified in the processor header, these keywords denote actions. The keyword DOCUMENT is defined as a keyword of type 2 and it denotes an object. The keyword FILE, however, is defined as both type 1 and 2, and it can, therefore, denote an action or an object, depending on where it is positioned in the command. It is possible to compose the two keyword types to make commands, such as DELETE FILE and FILE DOCUMENT.

You can save the keywords you have entered by issuing the SAVE or STOW command from the Command line. In addition to saving the keyword definitions in source form, the STOW command performs a consistency check on them. Once a keyword is stowed successfully, it is given an internal keyword number (IKN) which is used at runtime to evaluate a command. Synonyms are always linked to a master keyword and always take the IKN of their master.

Each line in the Keyword Editor contains the following fields:

Field	Explanation
I	Output field. An information field which can contain the following values:  E Indicates that a definition error has been detected. X Line is marked with X. Y Line is marked with Y. Z Line is marked with both X and Y. S Scan value found in this line.
Line	Output field. The line number of the editor.
E	Specifies the entry type for a keyword; that is, the position the keyword is to be entered in a command: first, second or third position, synonym or parameter indicator.  For instance, in the <a href="#">Keyword Editor screen</a> example above the keyword DELETE is of entry type 1 and DOCUMENT of type 2. Using these keywords, the command DELETE DOCUMENT can be defined.  The field takes any of the following characters as input:

Field	Explanation
	<p>1 The keyword defined in this line is to be used as the first titem in a command sequence.</p> <p>2 The keyword defined in this line is to be used as the second titem in a command sequence.</p> <p>3 The keyword defined in this line is to be used as the third titem in a command sequence.</p> <p>S The keyword defined in this line is to be used as a synonym for the preceding keyword with titem type 1, 2, 3 or P.</p> <p>P The keyword defined in this line is to be used as a parameter indicator in a command sequence.</p> <p>* No keyword is to be defined in this line. Instead, the line is to be used solely as a comment line.</p> <p>? This symbol is an output value which indicates an invalid keyword specification.</p>
Use	<p>Output field. The value displayed is determined by the value entered in the preceding field E:</p> <p>1-3 The first four characters of the user text specified in the processor header for the first, second and third keyword entries respectively are displayed. See also <a href="#">Keyword Editor Options - Header 2</a> in the section <i>Header Records</i>.</p> <p>S SYNO, the abbreviation for synonym, is displayed.</p> <p>P PARM, the abbreviation for parameter indicator, is displayed.</p>
Keyword	<p>Enter the keyword to be defined. Embedded blanks are not permitted. If you have specified in the processor header that keywords can only be upper case, then keywords are always translated to upper case, regardless of how they are entered. Otherwise, the case remains as entered.</p> <p>The maximum and minimum length of keywords depends on the settings specified in the header (default: 1 - 16 characters). Keywords must be unique unless specified otherwise in the header. Keyword prefixes can be used as described in <a href="#">Keyword Editor Options - Header 2</a> in the section <i>Header Records</i>.</p>
IKN	<p>Output field. The Internal Keyword Number (IKN) is an identifier assigned to each valid keyword. IKNs are useful for testing and debugging. They are allocated only when a keyword is successfully stowed (see also the <b>STOW</b> command under <i>Editor Commands</i>). Each keyword is assigned a unique IKN, except synonyms, which take the IKN of their master term (see the <a href="#">Keyword Editor screen</a> example above: DISPLAY and SHOW).</p>
ML	<p>Input and output field indicating the minimum length of a keyword. The field is an input field if <b>S</b> is specified in the <a href="#">Dynamic Length Adjustment</a> field of the processor header as described in <a href="#">Keyword Runtime Options - Header 1, Header Records</a>. In this case, you must specify the number of characters which must be entered for the keyword. For all other input, this field contains the minimum number of characters of a keyword a user must specify to avoid ambiguity with other keywords.</p> <p>For instance, in the <a href="#">Keyword Editor screen</a> example above, keyword MENU requires only input of <b>M</b> while keyword DISPLAY requires input of <b>DI</b> to avoid ambiguity with keyword DELETE.</p>
Comment	<p>Enter free text for a keyword. There are no input restrictions. The user text is included in the cataloged command processor if the field <a href="#">Catalog User Texts</a> is set to Y in the header definition as described in "Miscellaneous Options - Header 3", <i>Header Records</i>. It can be read at runtime</p>

Field	Explanation
	using the TEXT option of the PROCESS COMMAND statement. The header text appearing at the top of this column is controlled by the header definition fields "Header for User Text 1" and "Header for User Text 2".

## Editor Commands

In the Command line of the Keyword Editor, you can enter the following commands:

Command	Function
ADD	Adds ten empty lines to the end of the editor.
CANCEL	Returns to Processor Maintenance Menu.
CHECK	Tests the keyword source for consistency.
EXIT	Returns to Processor Maintenance Menu.
HELP	Displays valid escape characters and other useful processor settings.
INFO	Displays information on the keyword on which your cursor is positioned.
LET	Undoes all modifications made to the current screen since the last time ENTER was pressed.
POINT	Positions the line in which a line command .N is entered to the top of the current screen.
RECOVER	Returns keyword source that existed before last SAVE/STOW.
RESET	Deletes the current X and Y line markers.
SAVE	Keyword source is saved.
SCAN	Scans for the next occurrence of the scan value.
STOW	Keyword source is stowed and Internal Keyword Numbers (IKNs) are generated for valid keywords.

## Positioning Commands

Editor positioning commands are the same as the ones provided for the Natural program editor. For more information, see the description of the program editor in the *Editors* documentation.

The last line of the editor contains an output field which informs you of where your display is located in the editor. The following output values are displayed:

Top	Editor is currently positioned at the top of the keyword source.
Mid	Editor is currently positioned at the center of the keyword source.
Bot	Editor is currently positioned at the bottom of the keyword source.
Emp	Editor is currently empty.
All	The entire source is contained on the current screen.

## Line Commands

Line commands in the Keyword Editor are the same as in the Natural program editor with the exception of the commands `.J` and `.S`, which cannot be used.

Each command is entered beginning in the **E** field; the remaining part of the command is entered in the **Keyword** field, as illustrated in the screen below:

```

09:42:39                - SYSNCP Keyword Editor -                2000-05-04
Modify Keywords          Name SAGTEST   Library SYSNCP   DBID 10   FNR 32

I Line E Use  Keyword          IKN   ML Comment
-----
  1 1 Acti MENU                1004   1
  2 1 Acti DISPLAY            1002   2
  3 S Syno SHOW               1002   1
  4 . Acti i(3)TE             1001   2
  5 S Syno PURGE              1001   1

```

 **Caution:** When you move (`.M`) or copy (`.C`) lines, ensure that individual keywords are always moved or copied together with their synonyms.

When you delete (`.D`) lines, the corresponding keywords and any functions containing these keywords will not be deleted from the database until you issue the `STOW` editor command. As long as you do not issue the `STOW` command, these functions will still be displayed within the Function Editor.

## Function Maintenance

Functions are composed of the keywords entered in the Keyword Editor. Before it is possible to define functions, the keywords must be successfully stowed (see the section [Keyword Maintenance](#)).

- [Define Functions](#)
- [Editor Commands](#)
- [Direct Command QUICK-EDIT](#)
- [Local and Global Functions](#)

■ Procedure for Validating Functions

**Define Functions**

Use the Define Functions function and the Function Editor to specify functions and compose valid commands which can be accessed from a specific location.

▶ **To invoke the Function Editor**

- 1 In the Processor Source Maintenance menu, enter Function Code **F** (Define Functions).
- 2 Press ENTER.

The Function Editor screen is displayed.

The Function Editor displays all possible combinations of the keywords stowed in the Keyword Editor.

The screen below, shows the Function Editor with keywords used as examples in the **Keyword Editor screen** in the section *Keyword Maintenance*:

```

09:45:53          ***** NATURAL SYSNCP UTILITY *****          2000-05-04
User SAG          - Function Editor -
Edit Global Combinations   Name SAGTEST   Library SYSNCP   DBID 10   FNR 32

Global

I Ac  Action          Object          Addition          Global Local Any Loc
-----
      DELETE
      DELETE          DOCUMENT          Yes
      DELETE          FILE              Yes
      DISPLAY
      DISPLAY          DOCUMENT          Yes
      DISPLAY          FILE              Yes
      FILE
      FILE            DOCUMENT          Yes
      FILE            FILE              Yes
      INFORMATION          Yes
      INFORMATION          DOCUMENT
      INFORMATION          FILE

Repos: _____

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Cmd  Exit Last List Flip      +      Top   Loc  Loc+  Canc
    
```

You have to validate each keyword combination that you want to designate as a valid function in your application. A keyword combination can be validated as a global function, local function or both. A global function can be invoked from anywhere in an application, whereas a local function can only be invoked from a specific location within an application.

Two fields in the upper left corner of this screen indicate the current validation mode (local or global) and the location for which keyword combinations can currently be validated. In the screen above, the text "Edit Global Combinations" indicates that global mode is active. If the local mode were active, the text "Edit Local Combinations" would appear here. In the screen above, the text "Global" appears below this text. This indicates that global validation can be performed for all of the combinations listed. In local mode, in this field the name of the location appears for which local validation can be performed (for example, "Local DISPLAY FILE").

The Function Editor contains the following columns:

Column	Explanation
I	Output field. The following values are output as a result of function editing.  E Runtime action edited. D Referenced locations displayed. V Validation issued. R Validation removed.
Ac	Action to be taken. The following values can be entered:  VG Validate as global function. VL Validate as local function. RG Remove validation as global function. RL Remove validation as local function. DL Display all functions which reference the specified function as a local function. EG Invoke the Runtime Action Editor for a global function (see <a href="#">Runtime Action Editor</a> in the section <i>Runtime Actions</i> ). EL Invoke the Runtime Action Editor for a local function (see <a href="#">Runtime Action Editor</a> in the section <i>Runtime Actions</i> ). +G Invoke global mode, so that you can maintain any global functions. +L Invoke local mode for the current line, so that you can maintain local functions for this line. IN Information about keywords in this line.
Action	These three columns are used to display all possible combinations of currently defined keywords.
Object	The text which appears at the top of each keyword column is controlled by the fields <b>First Entry used as</b> , <b>Second Entry used as</b> and <b>Third Entry used as</b> as specified in the processor header (see <a href="#">Keyword Runtime Options - Header 1</a> in the section <i>Header Records</i> ).
Addition	

Column	Explanation
Global	If the function has been defined as a global command, Yes appears in this field.
Local	If the function has been defined as a local command, Yes appears in this field for the current location (only displayed in local mode).
Any Loc	Any Location. If the function has been defined as a local command anywhere else within the processor, Yes appears in this field for any other location.

## Editor Commands

In the Command line of the Function Editor, you can enter the following commands:

Command	Function
ANY ON	Enable the column Any Loc.
ANY OFF	Disable the column Any Loc (the column will be filled with question marks). This allows for faster scrolling in the Function Editor. Moreover, the third repositioning field is available. Also, processing-in-progress information windows will not be displayed.
FIELD	Display keyword-specific combinations.
GLOBAL	Activate global mode.
LOC	Position to next location group.
LOC+	Position forward by one location.
SINGLE ON	Display only single-word functions.
SINGLE OFF	Display all possible combinations.
TOP	Position to top of list.

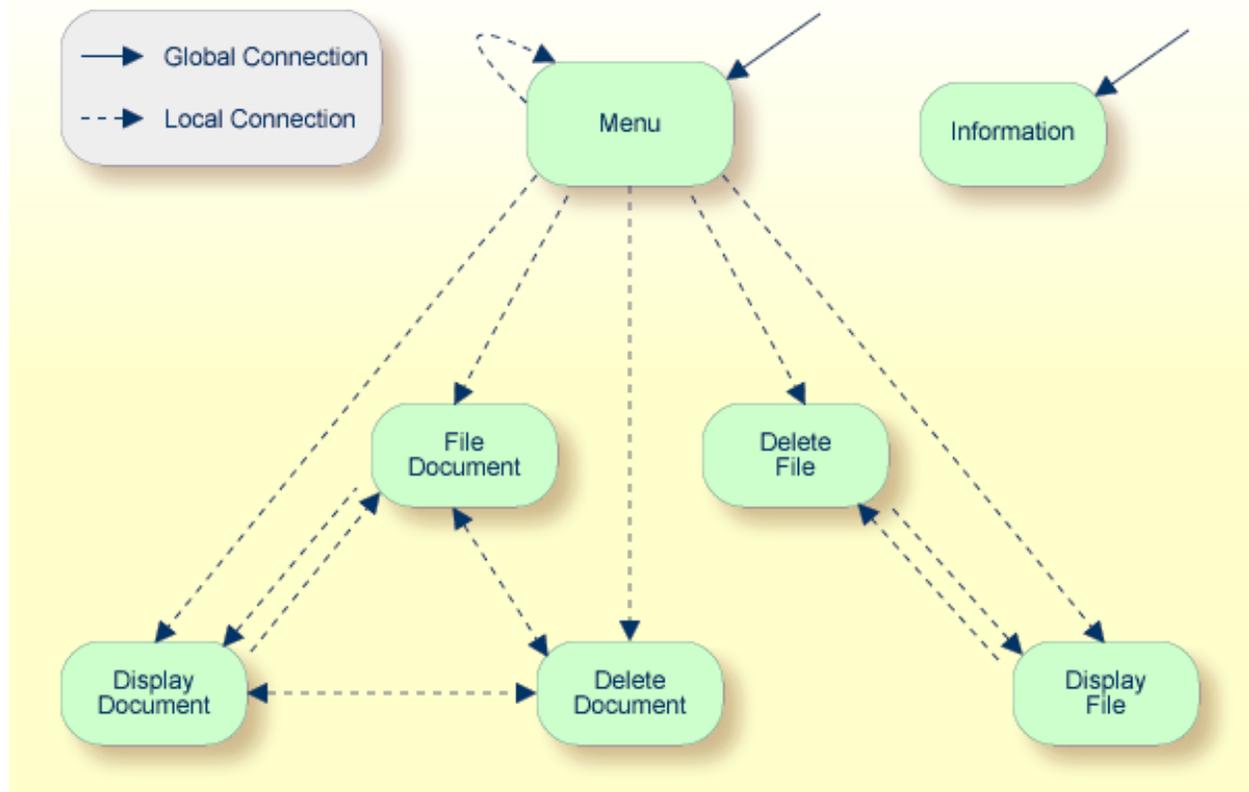
## Direct Command QUICK-EDIT

The direct command QUICK-EDIT enables you to quickly define local/global functions, as well as the corresponding runtime actions, by entering keywords or IKNs directly. This may be helpful for extremely large command processors. Note, however, that the location from which the command can be issued is not verified and navigation may not function correctly at runtime.

## Local and Global Functions

To understand the concept of local and global functions, you have to picture each valid keyword combination as a location in your application (for example, a location called Display File). In the Function Editor, you specify the commands which can be issued from this location, as well as from which locations this location can be reached using the command DISPLAY FILE.

### Local and Global Connections within a Sample Application:



In the sample application above, the Menu and Information locations are the only locations which have been designated as global. Thus, they can be accessed directly from all of the remaining locations in the application. All locations have been designated as local to the location Menu, except Information. The only way to get from the location Display File to Display Document is via Menu.

### Procedure for Validating Functions

The Function Editor operates in two modes: global and local. From global mode you can validate global functions and from local mode you can validate global and local functions. Global mode is the default mode. You can determine whether the editor is in global or local mode by the output field above the I field in the editor. If the editor is in global mode, then Global is displayed. If the editor is in local mode, then the location for which local functions are to be validated is displayed. Below is a general procedure for validating global and local functions for an application.

#### ▶ To validate global and local functions

- 1 With the Function Editor in global mode, enter **VG** (validate global) in the Ac field next to the corresponding action to validate all global functions.

Press ENTER.

The **Runtime Action Definition** screen appears.

- 2 Press PF3 to return to the Function Editor.

Yes appears under the column heading Global beside the validated functions.

- 3 Enter **+L** in the **Ac** field for each global function validated in the previous step, to switch to local mode.

Press ENTER.

- 4 Enter **VL** (validate local) in the **Ac** field for each function that is to serve as a location for this global function.

Press ENTER.

The Runtime Action Definition screen appears.

- 5 Press PF3 to return to the Function Editor.

Yes appears under the column heading Local beside the validated functions.

- 6 To validate local functions for a *local* location: Enter **+L** (invoke local mode) in the **Ac** field for each location validated in the previous step, to validate all local functions which are to be used from this location.

Press ENTER.

- 7 Enter **VL** (validate local) in the **Ac** field for each function that is to serve as a local function for the current location.

- 8 Press PF3 to return to the Function Editor.

Yes appears under the column heading Local beside the validated functions.



**Note:** If in the command processor header (Processor Header Maintenance 3) the field Invoke Action Editor is set to Y, in addition, the window Runtime Action Definition (see [Runtime Action Editor](#) in the section *Runtime Actions*) is displayed for each action.

## Runtime Actions

---

Once valid keyword combinations have been identified as either local or global functions in the Function Editor, it is possible to link each function with one or more runtime actions. Runtime actions consist of one or more steps which are to be carried out whenever a function is issued.

Below is information on:

- [Define Runtime Actions](#)

- Runtime Action Editor

## Define Runtime Actions

There are two different locations in SYSNCP from which you can define runtime actions: the Function Editor (see the section *Function Maintenance*) and the Result Editor. The Result Editor is explained in this section, including how to specify runtime actions for a function.

### ▶ To invoke the Result Editor

- 1 In the **Processor Source Maintenance** menu, enter Function Code **R** (Define Runtime Actions).
- 2 Press ENTER.

The Result Editor screen is displayed:

```

09:47:03          ***** NATURAL SYSNCP UTILITY *****
2000-05-04
User SAG          - Result Editor -

List defined combinations   Name SAGTEST   Library SYSNCP   DBID 10   FNR
32

I Ac Location          Command          Result
-----
< Global >            MENU            KR
< Global >            INFORMATION     SF
DELETE FILE           DISPLAY FILE     SF
DELETE DOCUMENT      DISPLAY DOCUMENT SF
DISPLAY FILE         DELETE FILE      SF
DISPLAY DOCUMENT     DELETE DOCUMENT SF
DISPLAY DOCUMENT     FILE DOCUMENT   SF
FILE DOCUMENT        DELETE DOCUMENT SF
FILE DOCUMENT        DISPLAY DOCUMENT SF
MENU                 DELETE FILE      KCS
MENU                 DELETE DOCUMENT KCCS

```

```

MENU                                DISPLAY FILE                        KRCS
Repo _____
-----
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Cmd  Exit Last List Flip      +      Top  Loc-- Loc+  Canc

```

The Result Editor contains all of the local and global functions specified in the Function Editor. Each line in the editor represents the location from which a command can be issued (Location field), the command itself (Command field) and an abbreviated summary of the action to be carried out when the command is issued (Result field).

The fields of the screen are explained in detail in the table below:

Field	Explanation
I	Output field. Information on the last action carried out on this line.
Ac	Action to be taken. The following values can be entered:  DI Display the runtime action definitions for this function. ED Edit the runtime action definitions for this function. PU Purge this function.
Location	Output field.  The location within the application from which the command (see Command field below) can be issued. If the function is global, then < Global > appears in this field (the command can be issued from any location).
Command	Output field. The command.  The contents of the Location and Command fields may be truncated if very long keywords are used.
Result	Output field.  Contains an abbreviated summary of the action to be performed when the command is issued. The first character represents the <b>Keep Location</b> information (see the following section); for all other characters, see the <b>Runtime Action Definition</b> table below.

## Runtime Action Editor

The Runtime Action Editor is used to define the actions to be taken when a command is issued from a specific location. The editor can only be invoked for functions which have been defined as global or local functions. The editor can be invoked either from the Function Editor or the Result Editor.

### ▶ To invoke the Runtime Action Editor from the Function Editor

- 1 In the **Ac** field, enter **EG** (edit global) for global functions.

Or:

In the **Ac** field, enter **EL** (edit local) for local functions.

- 2 Press ENTER.

### ▶ To invoke the Runtime Action Editor from the Result Editor

- 1 In the **Ac** field, enter **ED**.

- 2 Press ENTER.

The **Runtime Action Definition** window is displayed:

```

                                Runtime Action Definition

Location .... DISPLAY DOCUMENT
Command ..... DELETE DOCUMENT

Keep Location .... S
Data allowed ..... Y   More than one .... N   Max. Length ..... 99
Numeric ..... N   TOP of STACK ..... Y   Error: Drop ..... Y

A Runtime Action Definition
- - - - -
F DE-PGM_____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____

```

Actions are always associated with an origin and a destination. The origin is the location from which the command is issued, and the destination is the command itself. Thus, it is possible to link different actions to a command based on the context in which it is used.

In the Runtime Action Editor, you also specify whether the location is to remain the same after the actions have been carried out, or whether the command itself is to become the new current location.

Actions are specified by entering a single-letter code in the left column of the editor. Enter any parameters accompanying an action in the field next to the code. If the characters "/"\* are entered in this field, all subsequent input is considered a comment. If you omit a required parameter, you will be prompted for input.

The sequence in which actions are performed at runtime is determined by the order of entry in the editor (from top to bottom). Thus, if a FETCH is specified, all of the actions specified below it are not to be performed.

The Runtime Action Editor contains the following fields:

Field	Explanation
Location	Output field. The location from which the command is issued. If the function is defined as global, the field shows < Global >.
Command	Output field. Command for which actions are to be specified.
Keep Location	Specifies whether the current or a new location is to be active once the actions have been performed. A value in this field only affects commands with a specified EXEC option. Possible values are:  K Keep current location. The actions to be performed affect the current location only. S Set new location (global/local). Once the actions are performed, the command processor makes the command the new current location. Every command entered subsequently has to be either a local command of this new location or a global command.  <b>Note:</b> The defined actions themselves have no influence on the location; that is, any action performed does <i>not</i> cause the current location to be changed.
Other Options	All other options are related to the handling of parameters provided with this command sequence. For further information, see <a href="#">Command Data Handling - Header 4</a> in the section <i>Header Records</i> .  To activate the header defaults of these options, enter an asterisk (*).

#### ► To define runtime actions

- 1 Invoke the **Runtime Action Definition** window as described earlier.

- 2 In the field **A**, enter an action code and the corresponding action in the field opposite to it:

Code	Runtime Action Definition
V	Default value. No runtime action is specified.
T	Text which can be read at runtime using the TEXT or GET option of the PROCESS COMMAND statement.
M	Modify command line. The data are placed in the command line.
C	Command. This command is placed at the top of the Natural stack. If an asterisk (*) is specified here, the name of the program which issued this PROCESS COMMAND statement is put on top of the stack (STACK TOP COMMAND '*PROGRAM'). (*)
D	Data. These data are placed on top of the Natural stack. (*)
F	Natural program name. The program is invoked with a FETCH statement. (*)
S	Natural STOP statement. The statement is executed at runtime. (*)
E	The value specified in this line is to be moved immediately into the system variable *ERROR-NR.
R	A return code is entered in the DDM field RETURN-CODE as described in PROCESS COMMAND in the <i>Statements</i> documentation.
1 to 9	A text string. This value is entered into the multiple DDM field RESULT-FIELD as described in PROCESS COMMAND in the <i>Statements</i> documentation.
*	Comment line.

\* These actions are only performed with the EXEC option of the PROCESS COMMAND statement.

- 3 Press PF3 to leave the **Runtime Action Definition** window.



**Note:** The user exit NCP-REAM allows you to use some or all of the above codes. The user exit NCP-REEM allows you to modify the line that follows the heading of the Runtime Action Definition table. The user exit NCP-REDM allows you to define default values for runtime action definitions (if you use this user exit, see also *Invoke Action Editor* in the section *Header Records*). All user exits mentioned above are delivered in the Natural system library SYSNCP.

## Processor Cataloging

Once you have specified runtime actions for all of the functions you want to use in your command processor, you should catalog the command processor. Cataloging a command processor generates a Natural object of type Processor.

### ▶ To catalog a command processor

- 1 In the Processor Maintenance menu, enter Function Code **C** (Catalog Processor),

the name of the command processor to be cataloged,  
and the name of the Natural library in which the command processor is to be cataloged.

2 Press ENTER.



**Note:** If you have Natural Security installed, you have to allow the use of your command processor as described in the *Natural Security* documentation in the section *Functional Security*.

## Administrator Services

SYSNCP provides facilities for the administration of command processors. Only system administrators, as defined in *Natural Security*, are authorized to access these services.

### ▶ To access the administrative services

1 In the **Processor Source Maintenance** menu, enter Function Code **A** (Administrator Services).

2 Press ENTER.

The Administrator Services screen is displayed:

```

09:49:11          ***** NATURAL SYSNCP UTILITY *****          2000-05-04
User SAG          - Administrator Services -

                Code  Function
                S     Select Processor
                C     Copy Processor Source
                D     Delete Processor Source
                P     Print Source/Object/NCP-Buffer
                U     Unload Processor to Work File 3
                L     Load Processor from Work File 3
                F     Freeze Processor Source
                R     References from Natural Security
                ?     Help
                .     Exit

                Code .. _   Name .. SAGTEST_   Library .. SYSNCP__

Command ==>

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
```

Help Cmd Exit Last List Flip

Canc



**Note:** If you do not have Natural Security installed, be aware that all other users have administrator status.

Below is information on:

- [Select Processor](#)
- [Copy Processor Source](#)
- [Delete Processor Source](#)
- [Print Source/Object/NCP Buffer](#)
- [Unload Processor](#)
- [Load Processor](#)
- [Freeze Processor Source](#)
- [References from Natural Security](#)

## Select Processor

See the section [Processor Selection](#).

## Copy Processor Source

In copying processor sources, you have the choice of copying the entire processor or only selected sources (header, keywords, functions, runtime action definitions).

### ▶ To copy a command processor

- 1 In the Administrator Services menu, enter Function Code C.
- 2 Press ENTER.

The Copy Processor Source window is displayed to provide source and target information:

Copy Processor Source		
	Source	Target
Name .....	SAGTEST_	_____
Library .....	SYSNCP__	SYSNCP__
DBID .....	10__	10__
FNR .....	32__	32__
Password ....		
Cipher Key ..		
Replace .....	NO_	

- 3 In the **Source** fields, enter the name of the processor to be copied, and the library, database ID (DBID) and file number (FNR) in which the processor is stored. The default values correspond to the processor specified on the **Administrator Services** menu.

In the **Target** fields, enter the name of the processor to be copied to, and the library, database ID (DBID) and file number (FNR) into which the processor is to be copied.

In the **Cipher Key** field, enter the appropriate password and/or cipher key if the source and/or target file is protected by a password and/or cipher key.

In the **Replace** field, enter YES if you want to overwrite a processor in the target environment. The default for this field is NO.

- 4 Press ENTER.

The following window is displayed to select sources:

Copy Processor Source					
Mark	Copy	Source	Target		
—	Header .....	yes	no		
—	Keywords .....	yes	no		
—	Functions .....	yes	no		
	Runtime Action Definitions ..	no	no		
Source Name	SAGTEST	Library	SYSNCP	DBID	10 FNR 32
Target Name	TEST2	Library	SYSNCP	DBID	10 FNR 32
Replace ...	NO				

- 5 In the appropriate **Mark** fields, enter any character to select the sources you want to copy.
- 6 Press ENTER.

## Delete Processor Source

This function is used to delete processor sources.

### ▶ To delete a command processor

- 1 In the **Administrator Services** menu, enter Function Code **D**.
- 2 Press **ENTER**.

The **Delete Processor Source** window is displayed.

- 3 Specify the name of the processor to be deleted, and the library, database ID and file number in which the processor is stored. If the file is protected by a password and/or cipher key, you also have to enter the appropriate password and/or cipher key.
- 4 Press **ENTER**.

The following window is displayed to select the sources to be deleted:

Mark	Delete	Available
_	Header .....	yes
_	Keywords .....	yes
_	Functions .....	yes
_	Runtime Action Definitions ..	yes
Name SAGTEST    Library SYSNCP    DBID 10    FNR 32		

To the right of each processor source (header, keywords, functions, runtime action definitions) is a field which indicates whether the source exists. As command processor creation is a cumulative activity, you cannot delete a source without deleting all sources which are based on it. Thus, for example, in the screen above, you cannot delete the source of the functions without also deleting the source of the runtime action definitions.

- 5 In the appropriate **Mark** fields, enter any character to select each source indicated as **Available**.
- 6 Press **ENTER**.

### Print Source/Object/NCP Buffer

In addition to processor sources, you can also print the processor object and the NCP.

▶ **To print a command processor item**

- 1 In the **Administrator Services** menu, enter Function Code **P**.
- 2 Press **ENTER**.

The **Print Source/Object/NCP-Buffer** window is displayed.

- 3 Specify the name of the processor to be printed, and the library, database ID and file number in which the processor is stored. If the file is protected by a password and/or cipher key, you also have to enter the appropriate password and/or cipher key.
- 4 Press **ENTER**.
- 5 The following window is displayed to select items for printing:

```

                                Print Source/Object/NCP-Buffer

Mark  Print                                     Available
-----
_     Header ..... yes
_     Keywords ..... yes

_     Functions ..... yes
_     Runtime Action Definitions .. yes

_     Processor Object ..... yes
      NCP-Buffer ..... no

      Printer ..... _____

Name SAGTEST   Library SYSNCP   DBID 10   FNR 32
    
```

To the right of each processor source (header, keywords, functions, runtime action definitions) is a field which indicates whether the item exists.

Possible input values for the **Printer** field are the logical printer ID, VIDEO or SOURCE; see also DEFINE PRINTER in the *Statements* documentation.

- 6 In the appropriate **Mark** fields, enter any character to select the items you want to have printed and enter the logical printer name or the value VIDEO or SOURCE in the Printer field.
- 7 Press **ENTER**.

## Unload Processor

### ▶ To unload a command processor

- 1 In the **Administrator Services** menu, enter Function Code **U**.
- 2 Press ENTER. The **Unload Processor to Work File 3** window is displayed:

```

                                Unload Processor to Work File 3

                                Source           Target
Name ..... SAGTEST_
Library ..... SYSNCP__           SYSNCP__
DBID ..... 10__
FNR ..... 32__
Password ....
Cipher Key ..
Report ..... NO_

```

- 3 In the **Source** fields, enter the name of the processor to be unloaded, the library, database ID, and file number in which the processor can be found; the default value is the processor specified in the **Administrator Services** menu. Enter the appropriate password and/or cipher key if the file is protected by a password and/or cipher key.
- 4 In the **Report** field, enter YES if you want a report to be produced. Default is NO. You do not have to use a file extension. If you wish to use an extension, you must use the file extension ".sag".
- 5 Press ENTER.

When the processor is unloaded, all processor sources (header, keywords, functions, runtime action definitions) are written to Work File 3.



**Note:** Use the **Object Handler** to transfer command processors from one hardware platform to another.

## Load Processor

### ▶ To load a command processor

- 1 In the **Administrator Services** menu, enter Function Code **L**.
- 2 Press **ENTER**.

The **Load Processor from Work File 3** window is displayed for loading processors from Work File 3 to a Natural library:

```
Load Processor from Work File 3

Replace existing processors .. N
Produce load report ..... NO_
```

- 3 In the **Replace existing processors** field, enter **Y** or **N** (default is **N**) to specify whether existing processors with the same name are to be replaced by the processor to be loaded.
- 4 In the **Produce load report** field, enter **YES** (default is **NO**) if you want a report to be produced.
- 5 Press **ENTER**.



**Note:** Input for the processor name and the library into which the processor is to be loaded is taken from the work file.

## Freeze Processor Source

You can freeze a processor in its current state to prevent users from modifying it further.

### ▶ To freeze a command processor

- 1 In the **Administrator Services** menu, enter Function Code **F**.
- 2 Press **ENTER**. The **Freeze Processor Source** window is displayed.
- 3 Specify the name of the processor to be frozen, and the library, database ID and file number in which the processor is stored. If the file is protected by a password and/or cipher key, you also have to enter the appropriate password and/or cipher key.
- 4 Press **ENTER**.
- 5 In the following window, specify with **Y** or **N** whether modification of the processor sources is to be allowed or not. Default is **Y**.
- 6 Press **ENTER**.

## References from Natural Security

This function is only available if Natural Security is active in your environment. It is used to delete functional security references from Natural Security.

If functional security is defined for a processor in Natural Security, references are created automatically. These references are stored in the FNAT/FUSER system files along with the processor sources, not in FSEC.

### ▶ To invoke References from Natural Security function

- 1 In the **Administrator Services** menu, enter Function Code **R**.
- 2 Press ENTER.

The **Delete References** window appears.

- 3 Specify the name of the processor, and the library, database ID and file number in which the processor is stored. If the file is protected by a password and/or cipher key, you also have to enter the appropriate password and/or cipher key.
- 4 Press ENTER.
- 5 In the following window, you can delete main references, function references and auxiliary references.

For further information on functional security for command processors, refer to the section *Functional Security* in the *Natural Security* documentation.

## Session Profile

A session profile is a collection of user-definable defaults which determine how the SYSNCP screens appear or how SYSNCP reacts to input. In a session profile, for example, you can determine which command processor you want as default for a session or which colors you want assigned to screen attributes. In SYSNCP, there is a standard session profile called STANDARD which is issued to all new users. You can create several different session profiles and activate them as required.

Administrators for SYSNCP can access and modify any session profile in SYSNCP. Other users can access all session profiles, but can modify only those session profiles which are created under their user ID or which have the same name as their user ID.

### ▶ To define or modify a session profile

- Issue the PROFILE command from the Command line of the **Processor Source Maintenance** menu.

The first of three session profile maintenance screens is displayed.

Below is information on:

- [Session Profile Name](#)
- [Session Parameters - Profile 1](#)
- [Color Attributes - Profile 2](#)
- [Miscellaneous Attributes - Profile 3](#)

## Session Profile Name

The standard profile STANDARD or the value of the system variable \*USER is taken as default for the profile name.

If you are defining a new session profile, the parameters/attributes are defaults. You can modify these defaults as required and save them by entering the new name and pressing PF5.

The field Session Profile Name on each profile screen is both an input and output field. Thus, it is possible to define, read or save another profile from any of these screens by entering its name in the Profile Name field and pressing PF5 or PF4, respectively.

## Session Parameters - Profile 1

On the first profile maintenance screen, you can modify the following fields:

Field	Explanation
Apply Terminal Control 1	These fields can be used to enter the parameters of a SET CONTROL statement to be issued by SYSNCP at startup.  For example, when you enter <b>Z</b> in any of the fields, SYSNCP issues the statement SET CONTROL 'Z'.
Apply Terminal Control 2	
Default Processor Name	The default command processor name to be used for this session.
Default Processor Library	The Natural library to be used to store a command processor.
Cancel Reaction	Specifies whether a warning is to be issued whenever the requested modification is not completed and the CANCEL command is issued.  W Issue warning. B Back out and cancel without issuing warning.
Clear Key Allowed	Specifies whether clear key is allowed.  N Clear key disallowed. Y Clear key active and has same effect as CANCEL.
Default Cursor Position	Specifies placement of the cursor.

Field	Explanation
	1 Cursor to be positioned in first field of the screen. C Cursor to be positioned in command line.
Exec/Display Last Command	Specifies action to be taken as a result of the LAST command:  E Execute last command issued in command line. D Display last command issued in command line.

## Color Attributes - Profile 2

On the second profile maintenance screen, you can assign colors to various screen attributes, or overwrite existing color assignments.

By specifying the following color codes, you can assign the following colors:

Code	Color
BL	Blue
GR	Green
NE	Neutral
PI	Pink
RE	Red
TU	Turquoise
YE	Yellow

For color assignments to screen attributes, see also the terminal command %= in the *Terminal Commands* documentation.

## Miscellaneous Attributes - Profile 3

The following attributes can be specified on the third profile maintenance screen:

Field	Explanation
Message Line Position	The line on which messages are to be displayed. The value 21 is recommended. See also the terminal command %M in the <i>Terminal Commands</i> documentation for more information.
Text for PF5 Key	The PF5 function key is reserved for global (session-wide) use. The text to be displayed on the PF-key line for PF5 can be entered in this field.
Command for PF5 Key	The PF5 function key is reserved for global (session-wide) use. The command to be executed when PF5 is pressed can be entered in this field.

In addition, the screen displays when and by which user this profile was last modified.

# 101 SYSPARM Utility

---

▪ Invoking SYSPARM .....	670
▪ List Profiles .....	671
▪ Display Profile .....	672
▪ Add New Profile .....	674
▪ Modify Profile .....	674
▪ Editing Profiles .....	675
▪ Copy Profile .....	678
▪ Delete Profile .....	679
▪ Direct Commands and Batch Processing .....	679
▪ Maintaining Profiles in Different Environments .....	687

The SYSPARM utility is used to create and maintain a set of Natural profile parameters as an individual parameter profile that can be used for each Natural session.

When invoking Natural with dynamic profile parameters, you can specify individual parameters each time you invoke Natural. More comfortably, however, you can specify a set of parameters once in SYSPARM, store this set under a parameter profile name, and then invoke Natural with only one dynamic parameter: `PROFILE=profile-name`. The parameters defined in this parameter profile are then passed to Natural as dynamic parameters and must therefore comply with the syntax of `PROFILE` described in the *Parameter Reference* documentation.

For descriptions of the individual profile parameters that can be defined in a parameter profile, refer to the *Parameter Reference* documentation.

The parameter profiles are stored under user-defined names in the specified FNAT or FUSER Natural system file.

You can restrict the use of a parameter profile to specific users by setting the profile parameter `USER` as described in the *Parameter Reference* documentation.

## Invoking SYSPARM

### ▶ To invoke the SYSPARM utility

- Issue the following Natural system command:

```
SYSPARM
```

The **Menu** of the SYSPARM utility is displayed which provides the following functions and fields:

Field/Function	Explanation
<b>List Profiles</b>	Displays a list of all parameter profiles. From the list, you can select one or more profiles for display, modification or deletion.
<b>Display Profile</b>	Displays a specific parameter profile.
<b>Add New Profile</b>	Creates a new parameter profile.
<b>Modify Profile</b>	Changes an existing parameter profile.
<b>Copy Profile</b>	Creates a new parameter profile by copying an existing one.
<b>Delete Profile</b>	Deletes an existing parameter profile.
<b>Profile</b>	Selects the specified parameter profile. Enter a valid profile name or use asterisk (*) notation to select a range of profiles. Use asterisk (*) or leave this field blank to select all profiles available.
<b>Copy to</b>	Copies the specified parameter profile.

Field/Function	Explanation
<b>DBID</b>	Selects the database ID (DBID) of the Natural system file where the parameter profile is stored. Default is the current FNAT.
<b>FNR</b>	Selects the file number (FNR) of the Natural system file where the parameter profile is stored. Default is the current FNAT.
<b>Password</b>	Specifies the password (8 characters) of the Adabas file where the parameter profile is stored.
<b>Cipher</b>	Specifies the cipher code (8 digits) of the Adabas file where the parameter profile is stored.

The SYSPARM functions listed above are described in the remainder of this documentation.

## List Profiles

This function is used to list all parameter profiles contained in the specified Natural system file. From the parameter profile list, you can view, modify or delete one or more profiles.

### ▶ To list parameter profiles

- In the SYSPARM **Menu**, enter function code **L** and, in the **Profile** field, enter the name of a parameter profile or specify a range of names:
  - Enter an asterisk (\*) or leave the field blank to list all parameter profiles.
  - Use asterisk (\*) notation to list all parameter profiles with names that start with a specified value, where value is any combination of one or more characters, for example: ABC\*
  - Use the greater than (>) sign to list all parameter profiles with names greater than or equal to a specified value, for example: ABC>
  - Use the less than (<) sign to list all parameter profiles with names less than a specified value, for example: ABC<.

The **List Profiles** screen is displayed with a list of all parameter profiles of the specified name range.

You can press PF7 and PF8 to scroll up or down one page in the list.

▶ **To view, modify or delete a single parameter profile**

- Place the cursor anywhere in the **Sel.** or **Profile** column of the parameter profile you want to process and press one of the following PF keys:

PF Key	Line Command	Function
PF4	D	Invokes the <b>Display Profile</b> function.
PF5	M	Invokes the <b>Modify Profile</b> function.
PF6	X	Invokes the <b>Delete Profile</b> function.

Or:

In the **Sel.** column, next to the parameter profile you want to process, enter one of the line commands listed above and press **ENTER**.

Depending on the key pressed or line command entered, the selected parameter profile is either displayed on the screen or a delete confirmation window opens for this profile.

▶ **To view, modify or delete multiple parameter profiles**

- 1 In the **Sel.** column, next to the parameter profiles you want to process, enter one of the line commands listed in the **table** above.

You can use **PF7** or **PF8** to scroll up or down in the list of parameter profiles to go to the required items.

- 2 Press **ENTER** when you have finished entering all line commands for all parameter profiles.

Depending on the line command entered, the first parameter profile selected is either displayed on the screen or a delete confirmation window opens for this profile.

- 3 When finished with the first parameter profile, press **PF3** or **PF12** to process the next parameter profile. **PF3** also saves the current parameter profile, if modified.

## Display Profile

---

This function is used to display a parameter profile.

▶ **To display a parameter profile**

- In the **SYSPARM Menu**, enter function code **D** and the name of an existing parameter profile.

Or:



## Add New Profile

---

This function is used to create a new parameter profile.

▶ **To create a parameter profile**

- In the SYSPARM **Menu**, enter function code A and the name of the parameter profile you want to create, and press ENTER.

A blank **edit screen** appears. For instructions on entering the parameters you want to specify in the new parameter profile, see *Editing Profiles*.

## Modify Profile

---

This function is used to change the parameter specifications of a parameter profile.

▶ **To modify a parameter profile**

- In the SYSPARM **Menu**, enter function code M and the name of an existing parameter profile, and press ENTER.

Or:

Invoke the **Display Profile** screen for an existing parameter profile and press PF5.

Or:

Invoke the **List Profiles** screen and select one or more parameter profiles as described in the relevant section.

An **edit screen** with the parameter definitions of the specified parameter profile appears. For instructions on using this screen to add or modify definitions, see *Editing Profiles*.



You can enter a commentary text in each input line, or spread a comment over as many lines as you like. Comments must be preceded by `/*` and followed by `*/` as indicated in the example above.

If you need additional space for editing or want to combine different parameter profiles or group parameter profiles by categories, concatenate multiple parameter profiles by entering the `PROFILE` profile parameter as the last entry in the profile(s) concerned. The first parameter profile then evaluates the parameter strings specified in the second parameter profile as part of the string of the first parameter profile.

### Example:

`PROFILE=P2` entered at the end in parameter profile P1, and `PROFILE=P3` entered at the end in parameter profile P2, firstly invokes P1 and then P2 followed by P3.

For detailed information on the `PROFILE` profile parameter, see the relevant section in the *Parameter Reference* documentation.

This section covers the following topics:

- [PF Keys](#)
- [Help with Parameters](#)

## PF Keys

The following PF keys are available on the edit screen:

PF Key	Function
PF4	Checks whether the parameter specifications within the parameter profile are syntactically correct.
PF5	Stores the parameter profile.
PF9	Inserts one blank line below the line containing the cursor.
PF10	Deletes the line containing the cursor.
PF11	Copies the line containing the cursor.

## Help with Parameters

The **Help with parameters** field on the edit screen can be used to obtain help information on valid parameter specifications and incorporate new specifications into the current parameter profile.

### ▶ To view online help and include new parameters

- 1 In the **Help with parameters** field, enter one of the following:
  - The name of the required parameter.

- One or more characters with asterisk (\*) notation (for example, PR\*) for all parameters whose names start with the specified character(s).
  - An asterisk (\*) for all parameters available.
- 2 Do *not* press ENTER yet if you want to *insert* a new parameter specification into an existing parameter profile and proceed with the following step.

Mark the input line below which you want to insert the new parameter specification(s) by placing the cursor on this line, and then press ENTER.

Or:

Press ENTER if you want to *append* a new parameter specification to the end of the parameter profile.

- 3 After performing the previous step, depending on the parameter value entered earlier, one of the following appears:
- If you entered a full name, a help screen (see below) appears.
  - If you specified a name range using the asterisk (\*), a window opens with a list of all parameters of the specified range.

From this list, select the required parameter by entering any character in the input field next to the required parameter and pressing ENTER.

The help screen looks similar to the example below:

```

13:08:21          ***** NATURAL SYSPARM UTILITY *****          2009-02-11

The parameter AUTO is used to cause an automatic logon at the start
of the Natural session.

AUTO=ON          An automatic logon is executed at the start of the Natural
                  session. The value contained in the Natural system variable
                  *INIT-USER is used as the user ID for the logon.
AUTO=OFF          No automatic logon is performed.

Enter your parameter specification:          More Help:  (+,-)
>
>
>
>
>

```

```
Command==>

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

The upper screen section contains help text on the specified parameter (in the example above, AUTO). If the text contains more than one page, you can enter a plus (+) sign before the **More Help** field to display the next page. Enter a minus (-) sign to return to the previous screen or enter a period (.) to terminate the help function and return to the edit screen.

The lower section contains five input lines (marked with >).

- 4 Press ENTER.
- 5 In the input lines, enter the required parameter specifications.

You can press PF4 to check the specifications.

- 6 Press PF3 to terminate the help function.

The edit screen appears with the new parameter specifications either appended to the end of the parameter profile or inserted below the input line marked with the cursor in Step 3.

## Copy Profile

---

This function is used to copy parameter profiles.

### ▶ To copy a parameter profile

- 1 In the SYSPARM **Menu**, enter the following:
  - Function code C.
  - In the **Profile** field, the name of the parameter profile from which you want to copy the new parameter profile.
  - In the **Copy to** field, the name of the new parameter profile.
- 2 Press ENTER.

A message appears indicating successful completion of the copy operation.

---

## Delete Profile

---

This function is used to delete an existing parameter profile.

▶ **To delete a parameter profile**

- 1 In the SYSPARM **Menu**, enter function code `X` and the name of the parameter profile you want to delete, and press `ENTER`.

Or:

Invoke the **List Profiles** screen and select one or more parameter profiles as described in the relevant section.

A **Delete a Profile** window opens with the name of the parameter profile you want to delete.

- 2 Confirm the deletion by entering the name of the parameter profile in the input field and pressing `ENTER`.

You can cancel a delete operation by pressing `PF3` or leaving the input field blank and pressing `ENTER`.

A message appears indicating either successful completion or cancellation of the delete operation.

---

## Direct Commands and Batch Processing

---

The SYSPARM utility functions described earlier can also be executed by using corresponding SYSPARM commands in batch or online mode.

In addition to the functions provided on the SYSPARM utility screens, in batch mode, options are provided for specifying commentary text, the input delimiter and the input assign character.

The symbols used in the syntax diagrams shown in this section are explained in *System Command Syntax* in the *System Commands* documentation.

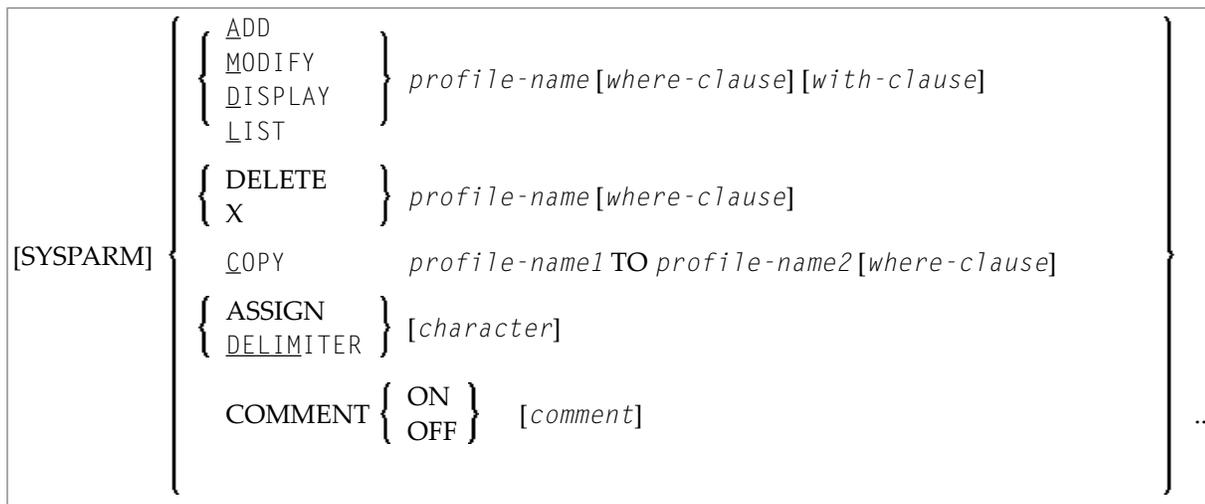
This section covers the following topics:

- **Basic Command Syntax**
- **where-clause**
- **with-clause**
- **Batch Processing**

- Example of SYSPARM in Batch

### Basic Command Syntax

Displayed below is the basic syntax that applies when processing SYSPARM commands.



The keywords, the variable values to be supplied with the keywords, and the optional clauses indicated in the basic command syntax of SYSPARM are described in the following table:

Keyword	Explanation
SYSPARM	Invokes the SYSPARM utility.  This keyword is only required when executing SYSPARM commands from the NEXT or MORE prompt.
ADD	Executes the <b>Add New Profile</b> function.  To terminate an ADD command, enter a period (.) in a separate line.
MODIFY	Executes the <b>Modify Profile</b> function.  To terminate a MODIFY command, enter a period (.) in a separate line.  To insert a commentary text into a parameter profile, place the comment between the characters /* and */.  See also <i>Example Input</i> .
DISPLAY	Executes the <b>Display Profile</b> function.
LIST	Executes the <b>List Profiles</b> function.
DELETE	Executes the <b>Delete Profile</b> function.
or X	

Keyword	Explanation						
COPY	Executes the <b>Copy Profile</b> function.						
ASSIGN	Specifies or displays the input assign character. You need to change the input assign character before you start processing data that contains the input assign character defined for your current session. Otherwise, you will receive an appropriate Natural system error.  If no value is specified, the output contains the character defined for your current session.						
DELIMITER	Specifies or displays the input delimiter character. You need to change the input delimiter character before you start processing data that contains the input delimiter character defined for your current session. Otherwise, you will receive an appropriate Natural system error.  If no value is specified, the output contains the character defined for your current session.						
COMMENT	Not applicable to the command REPLACE.  Activates (ON) the comment option and writes a comment after each parameter that is modified, or deactivates (OFF) the comment option.  If activated, and if no individual <i>comment</i> (see below) has been specified, the comment inserted for each parameter contains the following data: <ul style="list-style-type: none"> <li>■ the ID of the user who last modified the parameter profile as generated by the Natural system variable *USER and</li> <li>■ the date and time when the parameter profile was last modified as generated by the Natural system variables *DATV and *TIMX. (See also the relevant sections in the <i>System Variables</i> documentation.)</li> </ul> To modify a comment, use the REPLACE command (see the <i>with-clause</i> ).						
<i>character</i>	Any special character: see the Natural session parameters ID (Input Delimiter Character) and IA (Input Assign Character) described in the <i>Parameter Reference</i> documentation.						
<i>comment</i>	A commentary text that is to be placed between the characters /* and */.						
<i>profile-name</i>	The name of a parameter profile or a range of profiles. For a range, you can specify one of the following values where <i>value</i> is any combination of one or more characters: <table border="1" style="width: 100%; border-collapse: collapse;"> <tbody> <tr> <td style="width: 30%;"><i>value</i>*</td> <td>Selects all parameter profiles with names that start with <i>value</i>, for example: AB* selects AB and AB1 but not AA1.</td> </tr> <tr> <td><i>value</i>&gt;</td> <td>Selects all parameter profiles with names greater than or equal to <i>value</i>, for example: AB&gt; selects AB and AB1 but not AA1.</td> </tr> <tr> <td><i>value</i>&lt;</td> <td>All items with names less than or equal to <i>value</i>, for example: AX&lt; selects AB and AWW but not AXA.</td> </tr> </tbody> </table>	<i>value</i> *	Selects all parameter profiles with names that start with <i>value</i> , for example: AB* selects AB and AB1 but not AA1.	<i>value</i> >	Selects all parameter profiles with names greater than or equal to <i>value</i> , for example: AB> selects AB and AB1 but not AA1.	<i>value</i> <	All items with names less than or equal to <i>value</i> , for example: AX< selects AB and AWW but not AXA.
<i>value</i> *	Selects all parameter profiles with names that start with <i>value</i> , for example: AB* selects AB and AB1 but not AA1.						
<i>value</i> >	Selects all parameter profiles with names greater than or equal to <i>value</i> , for example: AB> selects AB and AB1 but not AA1.						
<i>value</i> <	All items with names less than or equal to <i>value</i> , for example: AX< selects AB and AWW but not AXA.						

Keyword	Explanation
	Name ranges are only allowed with the command LIST or, in batch mode, the commands LIST and DISPLAY.
<i>profile-name1</i> <i>profile-name2</i>	Only applies to the COPY command. The source parameter profile ( <i>profile-name1</i> ) from which to create a new parameter profile and the new target parameter profile ( <i>profile-name2</i> ) into which to copy the data.
<i>where-clause</i>	Indicates a <i>where-clause</i> described in the relevant section.
<i>with-clause</i>	Indicates a <i>with-clause</i> described in the relevant section.
...	Only applies in batch mode. Illustrates that you can specify more than one SYSPARM function by placing each function in a separate line.

### where-clause

The *where-clause* is optional and applies to the commands ADD, MODIFY, DISPLAY, LIST, DELETE and COPY. Its syntax is as follows:

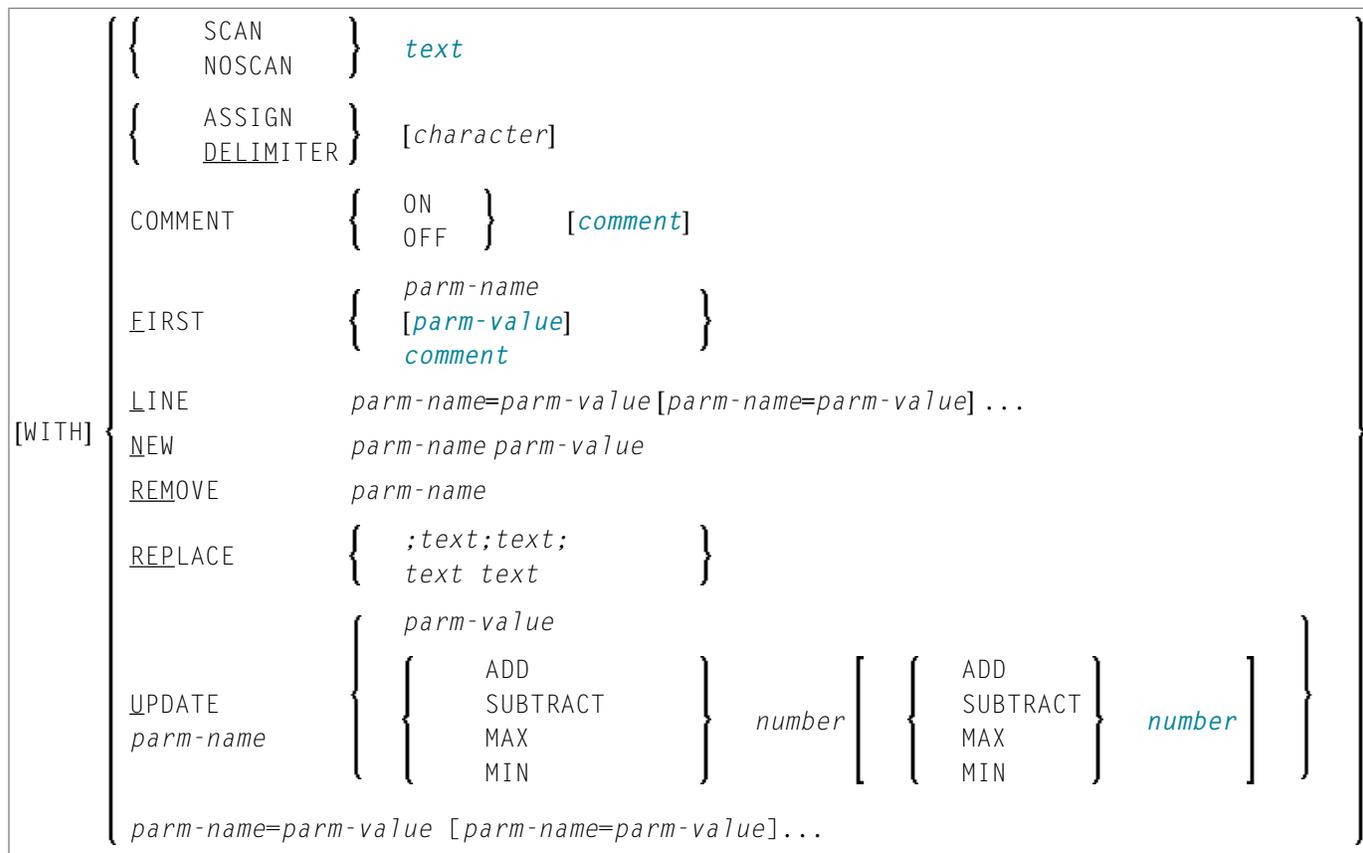
```
[WHERE] [DBID dbid] [FNR fnr] [PASSWORD password] [CIPHER cipher]
```

The keywords of the *where-clause* and the variable value that must be supplied with each keyword are explained in the following table:

Keyword	Valid Values
WHERE	Optional keyword that indicates the start of a <i>where-clause</i> .
DBID	The database ID ( <i>dbid</i> ) of the Adabas file where the parameter profile is stored.
FNR	The file number ( <i>fnr</i> ) of the Adabas file where the parameter profile is stored.
PASSWORD	The 8-character Adabas password ( <i>password</i> ) of the Adabas file where the parameter profile is stored.
CIPHER	The 8-digit cipher code ( <i>cipher</i> ) of the Adabas file where the parameter profile is stored.

## with-clause

The *with-clause* is optional and applies to the commands ADD, MODIFY, DISPLAY and LIST. Its syntax is as follows:



**Note:** In the syntax diagram above, the following only applies to the ADD command: [WITH] *parm-name=parm-value* [*parm-name=parm-value*] ...

The keywords of the *with-clause* and the variable value that must be supplied with each keyword are explained in the following table:

Keyword	Valid Values
WITH	Optional keyword that indicates the start of a <i>with-clause</i> . Exception: When using the ADD command, you must use the keyword WITH to clearly separate the command from the contents of the parameter profile.
SCAN	Only applies to the commands LIST and DISPLAY.  Scans one or more parameter profiles available in the specified Natural system file for a <b>text</b> string (for example, a parameter or a parameter value) and provides a list of the parameter profiles that contain the specified <b>text</b> string.

Keyword	Valid Values
NOSCAN	<p>Only applies to the commands LIST and DISPLAY.</p> <p>Exempts a <b>text</b> string from the scan over one or more parameter profiles available in the current Natural system file and provides a list of the parameter profiles that do <i>not</i> contain the specified text string.</p>
ASSIGN	See <b>ASSIGN</b> in the basic command syntax.
DELIMITER	See <b>DELIMITER</b> in the basic command syntax.
COMMENT	See <b>COMMENT</b> in the basic command syntax.
FIRST	<p>Moves a parameter and its value or a comment to the first position in a parameter profile. This can be required, for example, if you want to specify the profile parameter PARM.</p> <p>Comments contained in the first position of a parameter profile will remain in this position. A parameter and its value will only be moved to the first position if no comment occupies this position. Otherwise, the parameter will be appended after the comment(s). A comment inserted with FIRST will always be placed in the first position, before any comment(s) that originally occupied this position.</p>
LINE	Places the specified parameter(s) after LINE in a new line at the end of a parameter profile.
NEW	<p>Appends a parameter to the end of a parameter profile but not necessarily in a new line. To place a parameter in a new line, use the parameter LINE.</p>
REMOVE	Removes a parameter and the value assigned (including parentheses) from a parameter profile. The parameter to be removed can be a subparameter, such as the profile parameter DFS, which is a subparameter of the profile parameter RPC.
REPLACE	Replaces an old <b>text</b> string with a new text string contained in a parameter profile. See also the option <b>COMMENT</b> in the basic command syntax.
UPDATE	<p>Updates the value assigned to a parameter.</p> <p>If the specified parameter does not yet exist, it will be appended to the parameter profile.</p>
ADD	<p>For numbers: Increases the size of a parameter value by adding the specified number. The addition can be limited to a maximum or minimum value by specifying MAX or MIN (see below).</p> <p>For text: Appends the specified value to an existing list of values. If the primary parameter value is a single word, a list with two members consistig of the primary value and the specified value to be added is created.</p>
SUBTRACT	<p>For numbers: Decreases the size of a parameter value by subtracting the specified number. The subtraction can be limited to a maximum or minimum value by specifying MAX or MIN (see below).</p> <p>For text: Removes the specified value from an existing list of values. If this results in a one member or empty list, list notation (parentheses) remains.</p>
MAX	Specifies the maximum of a parameter value.
MIN	Specifies the minimum of a parameter value.

Keyword	Valid Values
<i>text</i>	<p>Any text string contained in a parameter profile.</p> <p>The following applies to the commands <code>SCAN</code> and <code>NOSCAN</code>: The text string must <i>not</i> contain any blank characters.</p> <p>The following applies to the <code>REPLACE</code> command:</p> <p>If neither the old text nor the new text contains blank characters, place a blank character between old and new text. If the old text or the new text contains blank characters, place the input delimiter character specified between old text and new text and around the entire old/new text string.</p> <p>For example:</p> <pre>;This comment is old;This comment is new;</pre> <p>As an alternative to the specified delimiter character, you can choose any of the following characters:</p> <ul style="list-style-type: none"> <li>' an apostrophe</li> <li>,</li> <li>.</li> <li>;</li> <li>/</li> <li>\</li> <li> </li> </ul>
<i>character</i>	See <a href="#">character</a> in the basic command syntax.
<i>parm-name</i>	The full name of a parameter.
<i>parm-value</i>	The value assigned to a parameter.
<i>comment</i>	See <a href="#">comment</a> in the basic command syntax.
<i>number</i>	A numeric value.

## Batch Processing

When processing SYSPARM in batch mode, consider the following:

- To terminate SYSPARM, in a separate line, enter a period (.) or `FIN`, where `FIN` ends the Natural session.
- To execute more than one SYSPARM function, specify each function in a separate line.
- To extend a function over two lines, enter the character defined with the session parameter `CF` (default is %) anywhere in the first line. This indicates continuation on the next line.

See also [Example of SYSPARM in Batch](#).

**Related Topics:**

*Natural in Batch Mode (Operations documentation)*

*Using the INPUT Statement in Non-Screen Modes (Statements documentation)*

*Using the INPUT Statement in Batch Mode (Statements documentation)*

**Example of SYSPARM in Batch**

The examples shown in this section demonstrate the use of SYSPARM commands and the result of parameter profile modifications executed in batch:

- [Example Profile TESTPROF - Before Job Submission](#)
- [Example Input](#)
- [Example Parameter Profile TESTPROF - After Job Execution](#)
- [Example Parameter Profile TESTPRO1 - After Job Execution](#)

**Example Profile TESTPROF - Before Job Submission**

```
/* This is a test profile. */
AUTO=ON FNAT = (102,110,PASSWORD) FUSER=(1099,1100,PASSWORD,12345678)
RPC=(RPCSIZE=80,SRVNAME=MYSERV,SERVER=ON,DFS=(SRV2,NODE1,,ACI))
PRINT=((2,12,18),AM=STD,DEST='PRINT**',OPEN=INITOBJ,CLOSE=CMD)
PRINT=((1,3,6-11,15),AM=NAF)
ESIZE=90
```

**Example Input**

```
/*JCL
.
.
.
*/
SYSPARM
COPY TEST1 TO TESTPROF WHERE DBID 10 FNR 32
DISP TESTPROF
COMMENT ON /* TESTUSER 29.Jan.2009 */
MODIFY TESTPROF with REM DFS
    REPLACE ;test profile;test profile for SYSPARM in batch;
    NEW BPSIZE 4096
    UPDATE ESIZE ADD 20 MAX 100
    FIRST PARM INHOUSE
    DELIMITER $
    UPDATE FUSER (,6)
    ASSIGN :
    LINE IM=D, INTENS=1, AUTO=T, MT=0, MADIO=0
.
DISP TESTPROF
ADD TESTPRO1 WITH /* 106,210 */
```

```

    FNAT=(106,210,PASSWORD),FUSER=(,211)
.
DISPLAY TESTPRO1
DELETE TESTPRO2 WHERE DBID 10 FNR 32
X TESTPRO3   WHERE DBID 10 FNR 32
.
FIN

```

### Example Parameter Profile TESTPROF - After Job Execution

```

/* This is a test profile for SYSPARM in batch. */
PARM=INHOUSE /* TESTUSER 29.Jan.2009 */
AUTO=ON FNAT = (102,110,PASSWORD) FUSER=(,6) /* TESTUSER 29.Jan.2009 */
RPC=(RPCSIZE=80,SRVNAME=MYSERV,SERVER=ON, )
PRINT=((2,12,18),AM=STD,DEST='PRINT**',OPEN=INITOBJ,CLOSE=CMD)
PRINT=((1,3,6-11,15),AM=NAF)
ESIZE=100 /* TESTUSER 29.Jan.2009 */ BPSIZE=4096 /* TESTUSER 29.Jan.2009
*/
IM=D, INTENS=1, AUTO=T, MT=0, MADIO=0 /* TESTUSER 29.Jan.2009 */

```

### Example Parameter Profile TESTPRO1 - After Job Execution

```

/* 106,210 */
FNAT=(106,210,PASSWORD),FUSER=(,211)

```

## Maintaining Profiles in Different Environments

The SYSPARM utility is used to maintain parameter profiles within the same FNAT or FUSER system file.

To transfer parameter profiles (for example, copy or move) between different FNAT and/or FUSER system files, and to perform a parameter profile operation (for example, delete or find) in a different environment, you can use the SYSMAIN utility. For details, see [Processing Profiles](#) in the *SYSMAIN Utility* documentation.



# 102

## SYSRDC Utility

---

- Functional Components of SYSRDC ..... 690
- Data-Collecting Events ..... 691
- Data Collected ..... 693
- Activating the Natural Data Collector ..... 695
- Trace Recording ..... 696
- User Exits for External Monitoring/Accounting ..... 697
- Calling the CMRDC Interface ..... 698

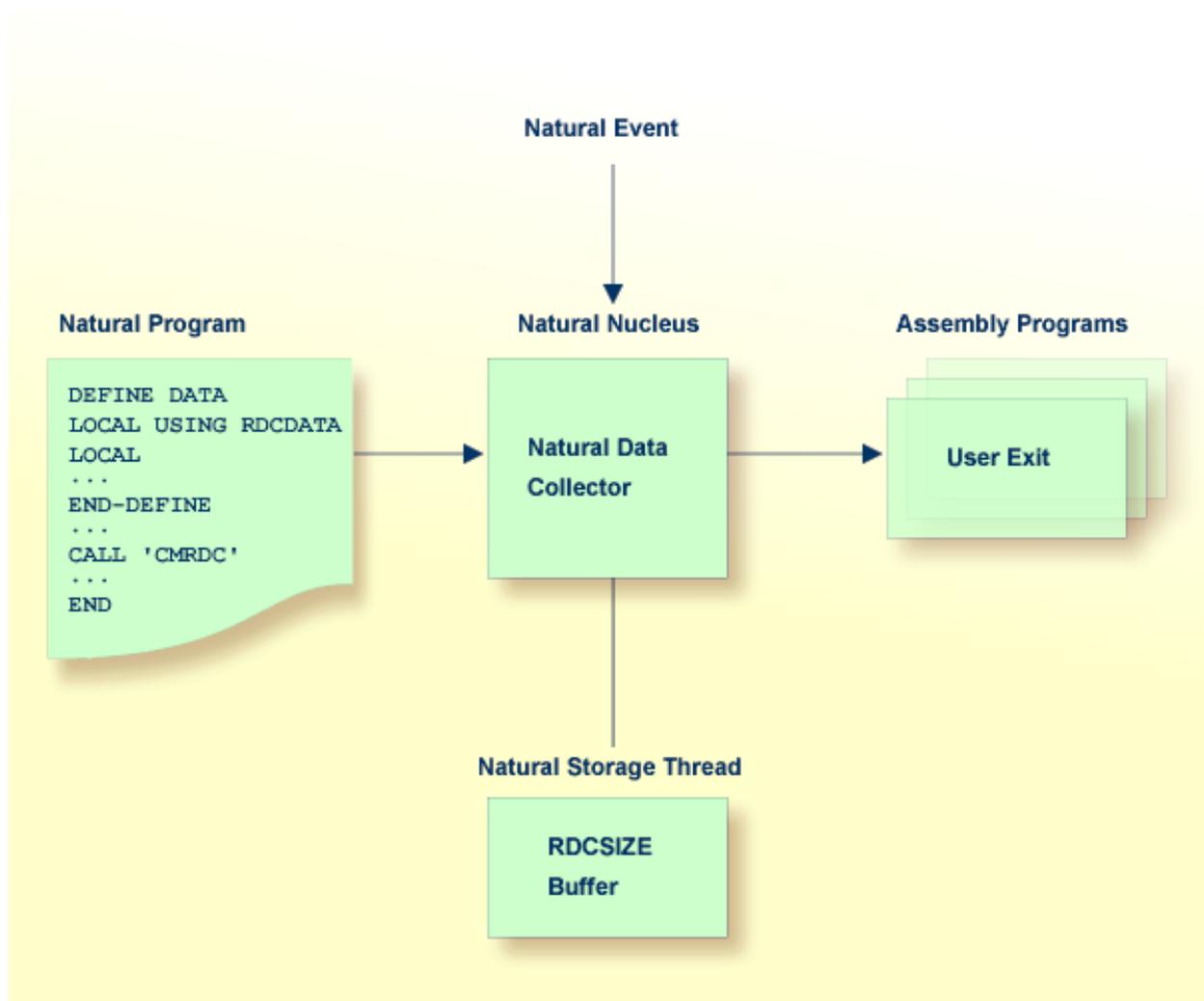
The SYSRDC utility is used to record monitoring and accounting data on the internal process flow within a Natural application. This data can be used for evaluating Natural session activities in external or Natural programs. The data is collected at events performed within Natural.

## Functional Components of SYSRDC

---

The components SYSRDC utility provides to collect data and supply it for further processing are listed and illustrated in the following section:

- **Natural Data Collector as part of the Natural nucleus:**  
Collects data and controls data recording in the RDCSIZE buffer.
- **User exits:**  
Passes data to external monitoring and accounting programs.
- **CMRDC interface:**  
Evaluates data in a Natural program for the current Natural session.



## Data-Collecting Events

Events specify activities within Natural. The table below lists the events at which data is collected within Natural and the types of event available.

Each type of event is assigned a one- or two-letter event code where the first letter represents the type of event and any second letter the subtype.

For example, in the event code **PL**, **P** represents the event type "program" and **L** the subtype "load".

Event	Event Code
At session initialization.	SI
At session termination.	ST
At program load.	PL
At program start.	PS
At program termination.	PT
Before a database call.	DB
After a database call.	DA
Before a terminal I/O.	IB
After a terminal I/O.	IA
Before a call of a non-Natural program.	CB
After a call of a non-Natural program.	CA
At a runtime error.	E
At an internal trace call.  The information internal traces provide are only intended for debugging purposes by Software AG personnel. See also "To activate an internal trace call event" below.	N
At a Natural statement.	NS
At outbound RPC message	RO
At inbound RPC message	RI
At start of RPC request execution	RS
At a user-defined event	U

#### ► To activate an internal trace call event

- 1 Specify the Natural profile parameter ITRACE=ON.
- 2 Define the Natural components that are to issue internal trace calls:
  - By using the NTTRACE macro of the Natural parameter module.

Or, by setting the dynamic Natural profile parameter TRACE.

The NTTRACE macro and the parameters ITRACE and TRACE are described in the *Parameter Reference* documentation.

---

## Data Collected

---

The data the Natural Data Collector collects at events are described in the following section.

For the layout of the data, see the Natural source dataset NAMRDC, or the local data area RD-CDATA that is provided in the Natural system library SYSRDC.

The data collected can be split into two categories: general and event-specific data:

- [General Data](#)
- [Event-Specific Data](#)

### General Data

The following general data is collected at every event:

- Software AG product name
- Product version
- Operating system
- TP monitor
- Run modes, such as addressing or residence mode of the Natural nucleus and the buffer pool.
- TP user or batch job name
- TP terminal ID
- Current Natural user ID
- Current Natural Security user group ID
- Current Natural library
- Current Natural program,
- Current program level,
- Line number of program statement currently executed.

## Event-Specific Data

The following data is only collected at the following events:

Event	Data Elements
Session initialization	none
Session termination	Termination return code. Natural termination message code NAT99 $nn$ . Name of back-end program.
Program load	Name of program to be loaded. Name of load library.
Program start/termination	Program type. Program name. Program library name. Database ID of program library.
Database call	Database type. Command code. Command ID. Database ID. Response code. Pointer that lists parameter addresses. (Only useful for user exits.)
Terminal I/O	Number of bytes sent. Number of bytes read. Total session storage allocated. Compressed session storage length.
Call of non-Natural program	Name of program called. Calling mode, such as dynamic or static mode. Program link location. Parameter type. Parameter address. Program entry address. Response code.
Runtime error	Natural system error message code. External abend code Name of error handling program as specified with the Natural profile parameter ETA (see the relevant section in the <i>Parameter Reference</i> documentation).
Internal trace	Up to 250 bytes of information on Natural nucleus components. The information provided is only intended for debugging purposes by Software AG personnel.
RPC request	RPC-specific information:  Environment: C = client, S = server.  RDC subtype S: subprogram name,

Event	Data Elements
	Adabas user ID (ETID), conversation status, logon indicator (Y = logon performed), impersonation indicator of RPC request (Y = impersonation performed).  RDC subtypes O and I: transport protocol, RPC function, type of client user ID, length of message, return code of transport layer, external conversation ID, client user ID, server node (outbound message of client only), server name (outbound message of client only).
User-defined event	Up to 250 bytes of user-defined information.

## Activating the Natural Data Collector

The Natural Data Collector is activated and controlled by the Natural profile parameter RDCSIZE (see also the relevant section in the *Parameter Reference* documentation).

This parameter determines the size (in KB) of the RDCSIZE buffer for the Natural Data Collector and controls **trace recording** as described in the relevant section.

By default, the parameter is set to RDCSIZE=0, which means that the Natural Data Collector is deactivated.

### ▶ To activate the Natural Data Collector

- Set RDCSIZE=2.

The Natural Data Collector is activated and the data collected is supplied to the user exits. No data is recorded in the RDCSIZE buffer.

To record data in the RDCSIZE buffer, see Trace Recording below.

## Trace Recording

---

The event data collected is always supplied to **user exists** to be used by external monitoring/accounting programs as described in the relevant section. Independently of the user exists, you can record the event data of the current Natural session in the RDCSIZE buffer. This can be useful for testing purposes. In this section, the recording of data in the RDCSIZE buffer is referred to as trace recording.

Trace recording starts when initializing a session and ends when terminating a session. The data recorded in the RDCSIZE buffer can be accessed by any Natural program within the same session.

The RDCSIZE buffer is filled in wrap-around mode; that is, the oldest record is overwritten when the buffer becomes full. At the end of the session, the contents of the buffer are deleted.

Trace recording is activated with the profile parameter **RDCSIZE** (see also the relevant section in the *Parameter Reference* documentation). This parameter also determines the size of the RDCSIZE buffer.

### ► To activate and use trace recording

- 1 Set the Natural profile parameter **RDCSIZE** to a value greater than 2.

Use the following formula to calculate the number of records that fit into the RDCSIZE buffer:

$$11 * (RDCSIZE - 2) = \text{number of records}$$

All event data collected for the current Natural session is recorded in the RDCSIZE buffer.

- 2 Retrieve the contents of the RDCSIZE buffer by using the CMRDC interface as described in *Calling the CMRDC Interface*.
- 3 If desired, use the CMRDC interface to stop or restart and trace recording in the RDCSIZE buffer, specify events for which data is to be recorded or execute additional functions described in *Calling the CMRDC Interface*.

## User Exits for External Monitoring/Accounting

The event data can be passed to external monitoring and accounting programs for evaluation of activities in Natural sessions. This is accomplished with the user exits and the examples of user-exit programs provided. User-exit programs are written in assembly language.

The Natural Data Collector supports any number of user exits. A user exit can be defined by the Natural profile parameter RDCEXIT (see the relevant section in the *Parameter Reference* documentation). An external monitoring/accounting program can be attached to each user exit.

Three exit names are predefined: RDCEX1, RDCEX2 and RDCEX3. If you use one of these names as entry point for your exit (which is linked to the Natural nucleus), the Natural profile parameter RDCEXIT is not required.

At every event listed under *Data-Collecting Events*, the user exits take over control by using the standard linkage call conventions described in the table below:

Register	Contents
1	Points to a parameter address list that consists of two addresses: one address points to the general data and the other points to event-specific data. The layout of these areas is mapped by the DSECT RDCGDATA and RDCLDATA respectively. Both DSECTs are supplied in source form in the Natural macro NAMRDC.
13	Points to a 72-byte standard save area.
14	Contains the return address.
15	Contains either the entry point address or the return code of the user exit.



**Note:** The user exits are called independently of the CMRDC interface.

A user-exit program must have the same attributes as Natural; that is, it must have the same addressing mode, and it has to be reentrant. It must be linked with the Natural nucleus according to the conventions of statically linked non-Natural programs. See also the profile parameters CSTATIC and RCA described in the *Parameter Reference* documentation.

By default, a 400-byte exclusive work area (per session) is supplied for each user exit (the field RDCGWRKA). If a larger work area is required for a user exit, it can be specified after the user-exit name in the Natural profile parameter RDCEXIT (see the relevant section in the *Parameter Reference* documentation). The work-area length is passed on to the user exits in the field RDCGWRKL and can be used for verification. The location of this work area can change during a session due to Natural relocation, but its contents are preserved.

In TP-monitor environments, the TP anchor address is supplied (the field RDCGANCH); for example, the CSA address under CICS. It can be used to access system information.

If a Program Check occurs during the execution of a user-exit program, further data collecting is disabled for the rest of the session to avoid recursive abend situations.

The section below contains information on:

- [Return Codes](#)
- [Examples of User-Exit Programs](#)

## Return Codes

Non-zero return codes are only supported for two events:

- Before a database call, where Register 15 can contain an Adabas response code, which is stored in the control block; the Adabas call will not be executed.
- At program start, where Register 15 can contain a Natural error message code; the program will not be executed, but an error condition will occur with the specified number.

## Examples of User-Exit Programs

The following examples of user-exit programs are provided in the following Natural source datasets:

Program	Dataset	Used for
NAMRDC	NAT $nnn$ .SRCE	DSECT macro for general and event-specific data.
XNATRDC1	NAT $nnn$ .SRCE	Natural TSO Interface and Natural batch interface under operating systems z/OS and z/VSE.
XNCFRDC1	NAT $nnn$ .SRCE	Com-plete.
XNCIRDC1	NCI $nnn$ .SRCE	Natural CICS Interface.

## Calling the CMRDC Interface

The CMRDC application programming interface is used for retrieving and controlling the trace data recorded in the RDCSIZE buffer. This can be useful for testing purposes.

### ▶ To invoke the CMRDC interface

- In the Natural program, issue a CALL statement.

The functions provided by the CMRDC interface and the syntax that applies to the corresponding Natural CALL statements are described in the following section.

In addition, the Natural system library SYSRDC provides example programs and the appropriate local data area RDCDATA.

The section below contains information on:

- Retrieving Trace Records
- Stopping and Restarting Trace Recording
- Selecting Event Types for Trace Recording
- User-Defined Events
- Retrieving the Trace Status
- CMRDC Return Codes
- Example Programs in Library SYSRDC

## Retrieving Trace Records

To read the data from the RDCSIZE buffer, invoke CMRDC with the following Natural statement:

```
CALL 'CMRDC' function event-time gen-data event-data seq.-number
```

The following parameters are passed:

Parameter	Format/Length	Explanation
<i>function</i>	A1	Possible functions:  F Get first trace record. G Get next trace record. N Get record of sequence number specified.
<i>event-time</i>	N10 or N12	Time of event.  The time is passed back in either of the following formats:  <i>HHMMSSXXXX</i> HH= hours, MM= minutes, SS= seconds, XXXX= fraction of a second  <i>HHMMSSXXXXXX</i> HH= hours, MM= minutes, SS= seconds, XXXXXX = fraction of a second
<i>gen-data</i>	A252	General data.
<i>event-data</i>	A252	Event-specific data.
<i>seq.-number</i>	I4	Sequence number of record.  Only applies to Function <b>N</b> (see above).

The retrieval functions stop recording data in the RDCSIZE buffer implicitly. To restart recording, use Function **S** described below.

## Stopping and Restarting Trace Recording

To stop or restart trace recording data in the RDCSIZE buffer, invoke CMRDC with the following statement:

```
CALL 'CMRDC' function
```

The following parameter is passed:

Parameter	Format/Length	Explanation
<i>function</i>	A1	Possible functions:  S Clear RDCSIZE buffer and restart trace recording. P Stop trace recording.

## Selecting Event Types for Trace Recording

By default, all events are selected for trace recording. Use this function if you wish only specific events to be recorded.



**Note:** This function only selects the events at which data is to be recorded in the RDCSIZE buffer. It does not affect the data passed to the user exits. Neither does it affect the status (started/stopped) of trace recording.

To select the types of event to be recorded, invoke CMRDC with the following statement:

```
CALL 'CMRDC' function type ...
```

The following parameters are passed:

Parameter	Format/Length	Explanation
<i>function</i>	A1	Possible function:  T Select events for trace recording.
<i>type</i>	A1, A2 or A3	The one- or two-letter <b>event code</b> for the type of event to be recorded as listed in <i>Data-Collecting Events</i> .  Specify any number of parameters for the event codes desired.  Alternatively, you can select a range of events or no event:  ALL All events.

Parameter	Format/Length	Explanation
		<i>value</i> * All events with event codes that start with <i>value</i> . For example, <b>P*</b> selects all events of the type program: PL, PS and PT. blank A blank character selects no event.

## User-Defined Events

To specify a user-defined event, invoke CMRDC with the following statement:

```
CALL 'CMRDC' function record
```

The following parameters are passed:

Parameter	Format/Length	Explanation
<i>function</i>	A1	Possible function:  U User-defined trace event.
<i>record</i>	A <i>nnn</i>	Trace record with a length ( <i>nnn</i> ) of up to 250 bytes.

## Retrieving the Trace Status

To retrieve the trace status, invoke CMRDC with the following statement:

```
CALL 'CMRDC' function status types
```

The following parameters are passed:

Parameter	Format/Length	Explanation
<i>function</i>	A1	Possible function:  C Retrieve trace status.
<i>status</i>	A1	Current trace status:  S Trace active. P Trace inactive.
<i>types</i>	A <i>nn</i>	The two-letter <b>event codes</b> of the event types currently set; see <a href="#">Selecting Event Types for Trace Recording</a> .  The variable <i>types</i> requires a minimum length ( <i>nn</i> ) of 64 bytes to provide sufficient space for all event types.

Parameter	Format/Length	Explanation
		This parameter is optional.

### CMRDC Return Codes

Code	Meaning
0	Function executed successfully.
4	Last trace record Only applies to Functions <b>F</b> and <b>G</b> (see above).
8	Too few parameters for this function.
12	Invalid function code.
16	Natural Data Collector not active, for example, RDCSIZE=0.
20	Natural Data Collector disabled after an error.
24	No buffer space available for trace recording (RDCSIZE=2 or smaller).
28	Invalid parameter value. Only applies to Functions <b>T</b> and <b>N</b> (see above).

### Example Programs in Library SYSRDC

The Natural system library SYSRDC contains the following example programs:

Program	Function
RDCDISP	Display all records in the RDCSIZE buffer and show the fields as specified in the program. See also <i>Example Output of Program RDCDISP</i> below.
RDCSTART	Restart trace recording.
RDCSTOP	Stop trace recording.
RDCSET	Select events for trace recording.
RDCUSER	User-defined event.
RDCSTAT	Retrieve trace status.

### Example Output of Program RDCDISP

The example screen below shows the extract of an output report produced by the example program RDCDISP.

ETIME1	TY	GCUID	LV	GPGM	T	GCAPL	PRLIB	PRNAM	CO	DB	FN	RC
0.0000	SI	SAG				SYSTEM						
0.0002	DB	SAG				SYSTEM			OP	10		
0.0002	DA	SAG				SYSTEM			OP	10		
0.0000	DB	SAG				SYSTEM			S1	10	1640	
0.0005	DA	SAG				SYSTEM			S1	10	1640	
0.0001	PL	SAG				SYSTEM	SYSLIB	MAINMENU				
0.0001	PS	SAG	1	MAINMENU	F	SYSTEM	SYSLIB	MAINMENU		10	1640	
0.0001	PL	SAG	1	MAINMENU		SYSTEM	SYSLIB	NAT00029				
0.0000	DB	SAG	1	MAINMENU		SYSTEM			S1	10	1640	
0.0002	DA	SAG	1	MAINMENU		SYSTEM			S1	10	1640	
0.0000	PS	SAG	2	NAT00029	N	SYSTEM	SYSLIBS	NAT00029		10	1640	
0.0000	CB	SAG	2	NAT00029		SYSTEM		CMMPP	S			
0.0000	CA	SAG	2	NAT00029		SYSTEM		CMMPP	S			
0.0000	PT	SAG	2	NAT00029	N	SYSTEM	SYSLIBS	NAT00029				
0.0000	PL	SAG	1	MAINMENU		SYSTEM	SYSLIB	USR2003P				
0.0000	PS	SAG	2	USR2003P	N	SYSTEM	SYSLIB	USR2003P		10	1640	
0.0001	PT	SAG	2	USR2003P	N	SYSTEM	SYSLIB	USR2003P				
0.0000	CB	SAG	1	MAINMENU		SYSTEM		CMUB	S			

The following table describes the columns that are displayed on the example screen, and the variables to which they refer. For further information, see the comments in the program source and the local data area RDCDATA. For an explanation of the Natural system variables mentioned, refer to the *System Variables* documentation.

Column	Explanation
ETIME1	Time interval in seconds between the execution of the current and the previous event.
TY	Type of event as listed in <a href="#">Data-Collecting Events</a> .
GCUID	Current Natural user ID as assigned by the Natural system variable *USER.
LV	Program level.
GPGM	Name of the current program as assigned by the Natural system variable *PROGRAM.
T	Type of program.
GCAPL	ID of the current application library as assigned by the Natural system variable *APPLIC-ID.
PRLIB	ID of the library where the program is stored. Applies to events of the type program, for example, Event Code PL.
PRNAM	Name of the program to be loaded for the type of event.
CO	Database command.

Column	Explanation
DB	Database ID.
FN	File number of Database.
RC	Database response code.

# 103

## SYSRPC Utility

---

The utility SYSRPC is used to maintain remote procedure calls.

- [Invoking and Terminating SYSRPC](#)
- [Service Directory Maintenance](#)
- [Stub Generation](#)
- [Calculating Size Requirements](#)
- [Parameter Maintenance](#)
- [Server Command Execution](#)
- [Remote Directory Maintenance](#)

### Related Topics:

- 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 Remote Procedure Call (RPC)* documentation.
- For explanations of expressions relevant to the SYSRPC utility, see also the section *Natural RPC Terminology* in the *Natural Remote Procedure Call (RPC)* documentation.
- The use of SYSRPC can be controlled by Natural Security: see *Protecting Utilities* in the *Natural Security* documentation.
- For information on Application Programming Interfaces provided to maintain remote procedure calls, see *Application Programming Interfaces for Use with Natural RPC* in the *Natural Remote Procedure Call (RPC)* documentation.



# 104 Invoking and Terminating SYSRPC

---

- Invoking SYSRPC ..... 708
- Terminating SYSRPC ..... 709
- Invoking Online Help ..... 709

This section provides instructions for starting and terminating the SYSRPC utility and invoking the help function.

## Invoking SYSRPC

---

You can invoke the SYSRPC utility by using either a system command or menu functions.

### ▶ To invoke SYSRPC using a system command

- In the Command line, enter the following command:

```
SYSRPC
```

and choose ENTER.

The **Client Maintenance** menu of the SYSRPC utility appears.

### ▶ To invoke SYSRPC using menu functions

- 1 From the Natural **Main Menu**, select **Maintenance and Transfer Utilities** and choose ENTER.

The **Maintenance and Transfer Utilities** menu is displayed.

- 2 Select **Maintain Remote Procedure Calls** and choose ENTER.

The **Client Maintenance** menu of the SYSRPC utility appears.

From the **Client Maintenance** menu, you can invoke all functions available for RPC (remote procedure call) maintenance:

- **Service Directory Maintenance**
- **Stub Generation**
- **Parameter Maintenance**
- **Server Command Execution**
- **Remote Directory Maintenance**

See the relevant sections for descriptions of these functions.

## Terminating SYSRPC

---

▶ To terminate the SYSRPC utility

- In the **Code** field of the **Client Maintenance** menu, enter a period (.).

Or:

Choose PF3 (Exit).

## Invoking Online Help

---

▶ To invoke the online help function

- Choose PF1 (Help).



# 105

## Service Directory Maintenance

---

- Service Directory Concept ..... 712
- Invoking Service Directory Maintenance ..... 714
- Fields on the Service Directory Screen ..... 716
- Commands for Service Directory Maintenance ..... 718
- Defining Logical Node Names and Logical Services ..... 721

The **Service Directory Maintenance** function is used to maintain a service directory in order to connect the client's calling program to a subprogram on a server.

The service directory information is stored in the NATCLTGS subprogram in the library that is defined with the NTRPC/RPC keyword subparameter RPCSDIR (see the *Parameter Reference* documentation). If RPCSDIR is set, the **Service Directory Maintenance** function references the library specified with RPCSDIR. If RPCSDIR is not set (this is the default), the library where you are logged on is referenced. In this case, log on to the library (or one of its steplibs) used by the client at runtime before you perform the **Service Directory Maintenance** function.

The name of the library referenced for service directory maintenance is indicated in the upper right corner of the **Service Directory** screen (see *Invoking Service Directory Maintenance*). If RPCSDIR is set, the screen title contains **Central**, which indicates that the library displayed on the screen is *not* the library where you are currently logged on, but the central library specified with RPCSDIR.

**Attention:**

If NATCLTGS is stored in the Natural system library SYSRPC, we strongly recommend that you move NATCLTGS to the application library (or one of its steplibs) used by the client.

For further information on how to apply the **Service Directory Maintenance** function, refer to *Specifying RPC Server Addresses* described in *Operating a Natural RPC Environment* in the *Natural Remote Procedure Call (RPC)* documentation.

## Service Directory Concept

---

A service directory has a hierarchical structure with a cascading list to assign subordinate to superior fields. The highest hierarchical level is node and the lowest is program. You cannot enter node, server, library and program in the same line. If you do so, an appropriate error message appears. You need to enter the value of a subordinate field in the lines below the superior field. You can assign several servers to a node, several libraries to a server and several programs to a library.

The node and server names specified in the service directory are either physical names or logical names and logical services:

- [Physical Nodes and Servers](#)

- [Location Transparency](#)

## Physical Nodes and Servers

Physical node and server names denote the names of real nodes (valid TCP/IP or Entire Net-Work addresses) and servers.

In *Example 1 - Standard View of Service Directory*, two servers are defined for one node. Both servers are connected to the same node: ETB045. The remote CALLNAT to subprogram SUB1 is executed on server NRPC001, whereas subprograms SUB2 and SUB3 are executed on server NRPC002.

The server names specified here must be identical to the server names specified for the server with the NTRPC/RPC keyword subparameter SRVNAME described in the *Parameter Reference* documentation. Analogously, the node name in the service directory must be identical to the node name specified for the server with the NTRPC/RPC keyword subparameter SRVNODE described in the *Parameter Reference* documentation.

## Location Transparency

Location transparency is a concept where physical node names can be replaced by logical node names, and a combination of physical node and server names can be replaced by logical services.

Logical node names and logical services are defined with EntireX and are assigned to physical node and server names at Natural runtime.

In *Example 1 - Standard View of Service Directory*, \*LOCTRAN in the field **Node** indicates that the field **Server** contains the logical service NRPC001 - LOGICAL. LOGBROKER=NODE in the field **Node** indicates the logical node name.

### Related Topics:

- [Defining Logical Node Names and Logical Services.](#)
- *Using Location Transparency* in the section *Operating a Natural RPC Environment* in the *Natural Remote Procedure Call (RPC)* documentation.
- The relevant sections in the EntireX documentation.

## Invoking Service Directory Maintenance

### Attention:

The **Service Directory Maintenance** function invokes the Natural editor. As a result, data stored in the source work area may be lost when invoking **Service Directory Maintenance**. An appropriate message will warn you not to delete any existing entries unintentionally: choose PF12 to cancel the function or choose ENTER to confirm the action and clear the source work area.

### ▶ To invoke the Service Directory Maintenance function

- 1 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
SM
```

- 2 Choose ENTER.

- If the service directory already contains service definitions, a window appears with the following message:

```
Existing service definitions found
```

In the **Code** field of the window, enter an A (default) to keep old definitions and append new ones and choose ENTER.

Or:

In the **Code** field of the window, enter an I to ignore all existing definitions and delete them from the service directory and choose ENTER.

The standard view of the **Service Directory** screen is displayed as shown in the following example:

### Example 1 - Standard View of Service Directory

	Node	Tr.	Server	Logon	Library	Program
1	ETB045_____	B	_____	_____	_____	_____
2	_____	—	NRPC001_____	N	_____	_____
3	_____	—	_____	—	SYSTEM_	_____
4	_____	—	_____	—	_____	SUB1_
5	_____	—	NRPC002_____	Y	_____	_____
6	_____	—	_____	—	SYSTEM_	_____
7	_____	—	_____	—	_____	SUB2_

```

8      _____  -  _____  -  _____  SUB3_____
9      *LOCTRAN_____  -  _____  -  _____  _____
10     _____  B  NRPC001-LOGICAL_  N  _____  _____
11     _____  -  _____  -  _____  SYSTEM_____
12     _____  -  _____  -  _____  SUB1_____
13     LOGBROKER=NODE  B  _____  N  _____  _____
14     _____  -  NRPC002_____  N  _____  _____
15     _____  -  _____  -  _____  SYSTEM_____
16     _____  -  _____  -  _____  S?B*_____

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  LocTr Exit  Find  -H   +H   -P   +P   Top  Bot  <   Canc
  
```

The **Service Directory** screen provides a maximum of 500 lines for input.

- If you choose PF11 or enter the less than (<) sign in the Command line, the extended node/server view of the **Service Directory** screen is displayed similar to the following example:

**Example 2 - Extended Node/Server View of Service Directory**

```

14:48:33          *** NATURAL Remote Procedure Call ***          2004-04-14
                  Service Directory                               SYSRPC

      Node                Tr.                Server                Logon
1  ETB045_____  B  _____  _____  _____
2  _____  -  NRPC001_____  N  _____  _____
3  _____  -  _____  -  _____  _____
4  _____  -  _____  -  _____  _____
5  _____  -  NRPC002_____  Y  _____  _____
6  _____  -  _____  -  _____  _____
7  _____  -  _____  -  _____  _____
8  _____  -  _____  -  _____  _____
9  *LOCTRAN_____  -  _____  -  _____  _____
10 _____  B  NRPC001-LOGICAL_____  N  _____  _____
11 _____  -  _____  -  _____  _____
12 _____  -  _____  -  _____  _____
13 LOGBROKER=NODE_____  B  _____  _____  N  _____
14 _____  -  NRPC002_____  N  _____  _____
15 _____  -  _____  -  _____  _____
16 _____  -  _____  -  _____  _____

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  LocTr Exit  Find  -H   +H   -P   +P   Top  Bot  >   Canc
  
```

If you choose PF11 or enter the greater than (>) sign in the Command line, the standard view of the **Service Directory** screen is displayed as shown in *Example 1 - Standard View of Service Directory*.

## Fields on the Service Directory Screen

The **Service Directory** screen contains the following input fields (one entry per line):

Field	Description						
<b>Node</b>	<p>The name of the node to which the remote CALLNAT is sent. See also <i>Natural RPC Terminology</i> in the <i>Natural Remote Procedure Call (RPC)</i> documentation.</p> <p>The maximum length of input is as follows:</p> <table border="0"> <tr> <td>Standard view of the <b>Service Directory</b> screen:</td> <td>16 characters</td> </tr> <tr> <td>Extended node/server view of the <b>Service Directory</b> screen:</td> <td>32 characters</td> </tr> <tr> <td>Using the <b>Location Transparency</b> window (see <b>PF2</b> in <i>Direct Commands and PF Keys</i>):</td> <td>192 characters</td> </tr> </table>	Standard view of the <b>Service Directory</b> screen:	16 characters	Extended node/server view of the <b>Service Directory</b> screen:	32 characters	Using the <b>Location Transparency</b> window (see <b>PF2</b> in <i>Direct Commands and PF Keys</i> ):	192 characters
Standard view of the <b>Service Directory</b> screen:	16 characters						
Extended node/server view of the <b>Service Directory</b> screen:	32 characters						
Using the <b>Location Transparency</b> window (see <b>PF2</b> in <i>Direct Commands and PF Keys</i> ):	192 characters						
<b>Tr.</b>	<p>The transport protocol:</p> <p>B indicates EntireX Broker ACI protocol.</p>						
<b>Server</b>	<p>The name of the server to which the remote CALLNAT is sent. See also <i>Natural RPC Terminology</i> in the <i>Natural Remote Procedure Call (RPC)</i> documentation.</p> <p>The maximum length of input is as follows:</p> <table border="0"> <tr> <td>Standard view of the <b>Service Directory</b> screen:</td> <td>16 characters</td> </tr> <tr> <td>Extended node/server view of the <b>Service Directory</b> screen:</td> <td>32 characters</td> </tr> <tr> <td>Using the <b>Location Transparency</b> window (see <b>PF2</b> in <i>Direct Commands and PF Keys</i>):</td> <td>192 characters</td> </tr> </table>	Standard view of the <b>Service Directory</b> screen:	16 characters	Extended node/server view of the <b>Service Directory</b> screen:	32 characters	Using the <b>Location Transparency</b> window (see <b>PF2</b> in <i>Direct Commands and PF Keys</i> ):	192 characters
Standard view of the <b>Service Directory</b> screen:	16 characters						
Extended node/server view of the <b>Service Directory</b> screen:	32 characters						
Using the <b>Location Transparency</b> window (see <b>PF2</b> in <i>Direct Commands and PF Keys</i> ):	192 characters						
<b>Logon</b>	<p>Initiates a Natural logon to the server.</p> <hr/> <p>This is possible at server or node level and applies to all definitions made at a hierarchically lower level. If the <b>Logon</b> option has been set for a specific server, it applies to all associated library and subprogram definitions.</p> <hr/> <p>Possible values are as follows:</p> <table border="1"> <tr> <td>Y</td> <td>If set to Y (Yes), for each non-conversational CALLNAT request or for each start of a conversation, the client initiates a Natural logon to the server using the current library name</td> </tr> </table>	Y	If set to Y (Yes), for each non-conversational CALLNAT request or for each start of a conversation, the client initiates a Natural logon to the server using the current library name				
Y	If set to Y (Yes), for each non-conversational CALLNAT request or for each start of a conversation, the client initiates a Natural logon to the server using the current library name						

Field	Description				
	<p>on the client, regardless of the libraries in the subordinate <b>Library</b> column that belongs to the <b>Server</b> field. You can use the Application Programming Interface USR4008N to specify a different library (see also <i>Logging on to a Different Library in Using the Logon Option</i>).</p>				
	<p>N or <i>blank</i></p> <p>If set to N (No) or if no value is entered, no logon is initiated.</p>				
	<p>After the remote CALLNAT has been executed (successfully or not) or at the end of a conversation, the server library is reset to its previous state. For more information, see <i>Using the Logon Option in the Natural Remote Procedure Call (RPC) documentation</i>.</p>				
	<p>See also <i>Server Command Execution</i>.</p>				
<b>Library</b>	<p>SYSTEM or the name of the library to which your client application is logged on during the execution of the remote CALLNAT.</p>				
<b>Program</b>	<p>The name of the remote subprogram to be accessed from the client.</p> <p>You can enter a name or a range of names. Valid names are any combinations of one or more alphanumeric characters with one or more asterisks (*) and/or one or more question marks (?) where:</p> <p>asterisk (*) denotes any string of characters, question mark (?) denotes a single character.</p> <p>Invalid combinations are:</p> <table border="1"> <tbody> <tr> <td>*?</td> <td>An asterisk followed by a question mark is converted to ?*.</td> </tr> <tr> <td>**</td> <td>Two or more consecutive asterisks are converted to a single asterisk.</td> </tr> </tbody> </table>	*?	An asterisk followed by a question mark is converted to ?*.	**	Two or more consecutive asterisks are converted to a single asterisk.
*?	An asterisk followed by a question mark is converted to ?*.				
**	Two or more consecutive asterisks are converted to a single asterisk.				

### Selection Criteria for Node and Server

At Natural runtime, the selection of a node and server depends on the value of the fields **Program** and **Library**. Comply with the following conditions:

#### Non-conversational CALLNAT

1. The **Library** field must contain the name of the current application library or SYSTEM.
2. The name of the subprogram specified in the CALLNAT statement must be contained in the **Program** field, which belongs to the **Library** field in point (1).

**Conversational CALLNAT**

1. The **Library** field must contain the name of the current application library or SYSTEM.
2. All subprograms specified in the OPEN CONVERSATION statement must be contained in a **Program** field, which belongs to **Library** field in point (1).

The node and server used for a non-conversational or conversational CALLNAT are taken from the superior **Node** and **Server** fields of the **Library** field in point (1).

---

## Commands for Service Directory Maintenance

---

This section contains information on the commands provided on the **Service Directory** screen:

- [Line Commands](#)
- [Direct Commands and PF Keys](#)

### Line Commands

The line commands provided on the **Service Directory** screen can be used to copy, move or delete single or multiple lines that contain field values.

Enter a line command at the beginning of a line, that is, overwrite the sequential number and choose ENTER.

See also [To copy or move a block of lines](#) and the direct command [RESET](#).

Line Command	Function
A	Copies or moves the block of lines marked with CC or MM below the line in which the command was entered.
CC	Marks the block of lines to be copied.
D	Deletes the marked line.
DD	Marks and deletes a block of lines.  Mark a block of lines by entering this command in the first and the last line of the block and choose ENTER to execute the command.
I	Inserts five empty lines below the line in which the command was entered.
MM	Marks the block of lines to be moved.
P	Copies or moves the block of lines marked with CC or MM above the line in which the command was entered.

► **To copy or move a block of lines**

- 1 At the beginning of the line where the block starts, overwrite the sequence number with either of the following line commands:

```
CC
```

to copy the block or

```
MM
```

to move the block.

- 2 At the beginning of the line where the block ends, overwrite the sequence number with either of the following line commands:

```
CC
```

to copy the block or

```
MM
```

to move the block.

- 3 Choose ENTER.

The line commands disappear, the sequence numbers are displayed again and the block of lines has been marked.

- 4 At the beginning of the line below or above which you want to place the marked block of lines, enter either of the following line commands:

```
A
```

to copy or move the block *below* the specified line or

```
P
```

to copy or move the block *above* the specified line.

- 5 Note that you can only execute A or P on lines where at least one field is filled.

- 6 Choose ENTER.

The block of lines is copied or moved below or above the specified line.

## Direct Commands and PF Keys

The following direct commands and PF keys are provided on the **Service Directory** screen:

Direct Command	PF Key	Function
EXPIRATION		<p>The remote directory data is loaded at runtime. The expiration time in seconds determines the period of validity of this data. If directory data is requested after the expiration time set, it will automatically be reloaded. If the expiration time is set to 0, the remote directory data will not be reloaded.</p> <p>With the direct command EXPIRATION, you can enter an expiration time in seconds, for example, EXPIRATION 86400. Maximum is an 8-digit number.</p> <p>If you do not provide a parameter with the command, the <b>Expiration Time</b> window appears where you can display or modify the current time.</p>
RESET		<p>Removes the line marks set with a line command as described in <i>Line Commands</i>.</p> <p>Note that if lines have been marked incorrectly, an appropriate message occurs and you must remove the erroneous line command before you enter RESET.</p>
	PF1	Invokes the editor online help.
	PF2	Invokes the <b>Location Transparency</b> window where you can define a logical node name or a logical service as described in <i>Defining Logical Node Names and Logical Services</i> .
	PF3	Exit. Prompts you to save modifications and exit the <b>Service Directory</b> screen.
FIND	PF4	<p>Invokes the <b>Find Item</b> window where you can search for a name:</p> <p><b>Find what</b> Enter an alphanumeric search string of up to 32 characters.</p> <p><b>Case sensitive</b> Replace the default setting N (No) by Y (Yes) to distinguish between uppercase and lowercase characters.</p> <p><b>Whole words only</b> Replace the default setting N (No) by Y (Yes) to search for complete search strings only.</p> <p>Choose ENTER to start searching and move from one hit to the next if one exists. Press PF4 to restart searching from the beginning.</p> <p>The hits are marked with the cursor.</p>
-H	PF5	Scrolls half a page backward/forward.
+H	PF6	
-P	PF7	Scrolls one page backward/forward.
+P	PF8	
TOP	PF9	Scrolls to the beginning of the list.
BOT	PF10	Scrolls to the end of the list.

Direct Command	PF Key	Function
	PF11	Toggles between the standard view of the <b>Service Directory</b> screen (see <a href="#">Example 1 - Standard View of Service Directory</a> ) and the extended view of the fields <b>Node</b> and <b>Server</b> (see <a href="#">Example 2 - Extended Node/Server View of Service Directory</a> ).
>	PF11	Displays the extended view of the fields <b>Node</b> and <b>Server</b> . The extended node/server view does not display the fields <b>Library</b> and <b>Program</b> as shown in <a href="#">Example 2 - Extended Node/Server View of Service Directory</a> .
<	PF11	Displays the standard view of the <b>Service Directory</b> screen as shown in <a href="#">Example 1 - Standard View of Service Directory</a> .
CANCEL	PF12	Exits the <b>Service Directory</b> screen without saving any modifications.

## Defining Logical Node Names and Logical Services

Logical node names or logical services can only be defined for node or server fields that already contain any values.

Note that defining a logical service, the original (physical) node name will be replaced by \*LOCTRAN and it is *not* possible to automatically convert back logical node names or logical services. For instructions on removing logical names and servers, see [To remove a logical node name or logical service](#).

### ► To define a logical service

- 1 Place the cursor on a **Server** field and choose PF2 (LocTr).

The **Location Transparency - Logical Service** window appears.

- 2 If desired, modify the existing values and choose ENTER.

The **Server Type Conversion** window appears as an additional window. Choose either of the following options:

- Enter a Y (Yes) and choose ENTER to confirm and execute the conversion.

The value in the field **Node** that relates to the specified server is replaced by the following character string: \*LOCTRAN. This string indicates that a node/server combination was converted to a logical service.

- Enter any character except Y or do not enter any value to cancel the conversion.

The physical node and server names are retained.

▶ **To define a logical node name**

- 1 Place the cursor on a **Node** field and choose PF2 (LocTr).

The **Location Transparency - Logical Node Name** window appears with the preset value of LOGBROKER=*name* where *name* denotes the logical EntireX Broker name.

If desired, modify *name* but do not modify the character string LOGBROKER=.

- 2 Choose ENTER to confirm and execute the conversion.

The physical node name is converted to a logical name.

Or:

Choose PF12 (Canc) to cancel the conversion.

The physical node name is retained.

▶ **To remove a logical node name or logical service**

- For a logical node name:  
In the **Node** field, remove the character string LOGBROKER=.

For a logical service:

Delete the logical service and insert physical server(s) by using the **line commands** D and I as described in *Line Commands*.

# 106

## Stub Generation

---

A stub subprogram (interface object) is a Natural subprogram that is used to connect the client's calling program to a subprogram on a server.

Client stub subprograms are actually not required if automatic Natural RPC (Remote Procedure Call) execution is used with the one important exception described below. However, it can be advantageous to generate stub subprograms as explained in *Stubs and Automatic RPC Execution* in the section *Operating a Natural RPC Environment* in the *Natural Remote Procedure Call (RPC)* documentation.

**Note for EntireX RPC Servers:**

It is recommended to generate a stub subprogram if you want to call an EntireX RPC server. In this case, you must use the appropriate SYSRPC stub generation function described in this section to define the same group structure and attributes (parameter direction) as in the IDL (Interface Definition Language) definition of the subprogram. If the IDL does not contain group structures, it is recommended to use the direct command `COMPAT IDL` before generating the stub subprogram. For details refer to [Special Considerations for Calling EntireX RPC Servers](#).

**Note for Reliable RPC:**

It is recommended to generate a stub subprogram if you want to use reliable RPC. If the parameter definitions do not contain group structures, you must set `COMPAT IDL` before generating the stub subprogram. (For details refer to [Special Considerations for Reliable RPC](#)).

You can generate a stub subprogram from new parameter definitions or from existing definitions in a subprogram or a parameter data area (PDA).

 **Caution:** The subprogram or PDA used for generating a stub subprogram can no longer be referenced in the local environment on the client side. The stub generation function completely changes the source of the subprogram so that it becomes unusable for local program calls.

The following sections describe the functions and commands provided to generate single or multiple stub subprograms:

- **Generating Single Stubs with Parameter Specification**
- **Generating Multiple Stubs**

# 107

## Generating Single Stubs with Parameter Specification

---

- Using the Stub Generation Function ..... 726
- Specifying Parameters ..... 729
- Examples of Stub Generation ..... 731

The **Stub Generation** function provides the option to generate single stub subprograms (interface objects) online by using a stub generation screen where you either type in the parameter definitions required or read them in from an existing subprogram or a parameter data area (PDA).

Generating a stub subprogram from a PDA saves you the effort of creating a subprogram and defining an inline parameter data area before generating the stub subprogram.

## Using the Stub Generation Function

---

The stub subprograms (interface objects) are generated into the current Natural library in the current system file. Therefore, we strongly recommend that you log on to the application library (or one of its steplibs) used by the client at execution time of the remote `CALLNAT`.

 **Important:** The stub generation function overwrites any data contained in the source work area. When you invoke the stub generation function, a corresponding message will warn you not to delete any existing data unintentionally: choose `PF12` to cancel the function or choose `ENTER` to confirm the action and overwrite the contents of the source work area.

### To generate a single stub subprogram

- 1 Before you invoke the `SYSRPC` utility, consider the following:
  - Log on to the library into which you want to generate the stub subprogram.
  - If you generate a stub subprogram from a PDA: Rename or copy the PDA from which you want to generate the stub subprogram if there are objects on the client side that still reference this PDA. The new name of the PDA must be identical to the name of the remote `CALLNAT` program.

- 2 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
SG
```

- 3 Choose `ENTER`.

The **Generate Client Stub Routine** window appears.

- 4 In the **Program Name** field, enter the name of the stub subprogram to be generated.

The name of the stub subprogram must be identical to the name of the remote `CALLNAT` program.

The **Library** field is preset to the name of the current library and cannot be changed.

In the **Compression** field, enter compression type 0, 1 or 2 (default is 1); see *Using Compression* described in *Operating a Natural RPC Environment in the Natural Remote Procedure Call (RPC)* documentation.

5 Choose ENTER.

- If the name entered in the **Program Name** field corresponds to the name of an object that already exists in the assigned library, a window appears with an appropriate message:

Enter an N (No) and choose ENTER if you want to cancel the operation. You will return to the **Client Maintenance** menu.

Or:

Enter a Y (Yes) and choose ENTER if you want to continue with the stub generation.

If the specified name is identical to a cataloged object of the type subprogram or PDA, the parameter definitions of the respective subprogram or PDA are displayed on the **Stub Generation** screen.

If the specified name is identical to a stub subprogram for which also a source object exists, all field attributes (see also *Specifying Parameters*) from a previous stub generation are retained. Otherwise, all field attributes are set to M (modifiable).

- If the name entered in the **Program Name** field, does *not* correspond to the name of an object that already exists in the assigned library, an empty **Stub Generation** screen is displayed.

6 On the **Stub Generation** screen, add or modify the parameters to be used in the stub subprogram as described in *Specifying Parameters*.

The commands provided on the **Stub Generation** screen correspond to the commands described in *Commands and PF Keys* in the section *Service Directory Maintenance*.

Exceptions:

Attribute	Values
EXPIRATION	Not applicable to stub generation.
COMPAT	IDL   NONE   void  IDL    Stub generation according to IDL requirements. NONE   Stub generation according to Natural requirements. void   Show COMPAT setting.  <b>Note:</b> See also: <i>Special Considerations for Reliable RPC</i> and <i>Special Considerations for Calling EntireX RPC Servers</i> .
LIMIT	32000   1GB   void

Attribute	Values
	32000 Sets the upper size limit to 32000 bytes. 1GB Sets the upper size limit to 1 GB. void removes a size limit set with <code>LIMIT 32000</code> or <code>LIMIT 1GB</code>

- 7 Choose `ENTER` to generate the stub subprogram and to exit. The stub subprogram is generated into the assigned library.

The **SYSRPC - Information** window appears which indicates the size the stub subprogram requires for sending data from the client to the server or vice versa. The size includes internal RPC information used for the stub subprogram. The indication of the size helps you configure the middleware layer used; for example, the Broker attribute file when EntireX Broker is used.

The following message appears in the **SYSRPC - Information** window when you generate a stub subprogram from the example subprogram TESTS5 (see [Example 1](#) below):

```
Stub TESTS5 is generated in library SAGTEST (99,49).
It requires:
    Send length: 2249 bytes
    Receive length: 2221 bytes
```

If dynamic parameters, X-arrays or X-group arrays are used, this message only indicates the minimum length requirements. The actual length requirements can only be determined during program execution and may be different from call to call. If the `Send length` or the `Receive length` exceeds the Entire Net-Work limit of 32000 bytes, a window appears with a corresponding warning:

Enter a `Y` (Yes) to continue, or an `N` (No) to cancel the generation. If you enter a `Y`, this setting is kept for the entire SYSRPC session, that is, you can continue generating stub subprograms without receiving further warnings.

If the total data (without internal RPC information) sent or received exceeds the limit of 1073739357 bytes (which is 1 GB minus 2467 bytes of internal RPC information), SYSRPC stops processing and issues a corresponding error message. This error message displays the subtotal of the data in bytes that could be transferred at the field up to which the subtotal was calculated. The corresponding field is then marked. In this case, reduce the amount of data and then continue generating the stub subprogram.

If the stub subprogram was generated in the Natural system library SYSRPC, you must move the stub subprogram to the application library or steplib using the Natural transfer utility `SYSMAIN` or the Object Handler. Note that you may have to recatalog the source of the stub subprogram in the target environment.

## Specifying Parameters

In the input fields provided on the **Stub Generation** screen, you can enter the parameter definitions that are used in the stub subprogram (interface object). You can specify a maximum of 5000 parameters. Unless indicated in the table below, input in the fields is mandatory.

Field	Description
<b>Level</b>	<p>The level of the field.</p> <p>A level can be a number in the range from 01 (highest level) to 99 (lowest level). The leading 0 is optional.</p> <p>See also <i>Defining Groups</i> and <i>Example 2</i> for an example of a group definition.</p>
<b>Attr</b>	<p>The attribute of the parameter:</p> <p>M (modifiable - INOUT), 0 (output - OUT) or I (input - IN).</p> <p>Parameters assigned a level number of 2 or greater are considered to be a part of a group. Parameters within a group must have the same attribute as the immediately preceding group that is assigned one level higher. For nested groups, this is the attribute of the group with the highest level. For an example of a group definition, see <i>Example 2</i>.</p> <p>If a stub subprogram has been generated from a subprogram or from a PDA, the attribute is M by default and may need modification.</p> <p>If a stub subprogram has been generated from another stub subprogram, the attribute values specified for the original stub subprogram are retained.</p> <p>The generated stub subprogram contains a comment that indicates the attribute specified for the parameter: IN, OUT or INOUT.</p>
<b>Type</b>	<p>A Natural data format such as N (numeric) and G (group), or K (Kanji). Natural data formats C (attribute control) and Handle are not allowed.</p> <p>For a description of Natural data formats, see <i>Format and Length of User-Defined Variables</i> and <i>Special Formats</i> in the section <i>User-Defined Variables</i> in the <i>Programming Guide</i>.</p>
<b>Length</b>	<p>The length of the parameter or DYNAMIC.</p> <p>This field does not apply to the following Natural data formats: D (date), G (group), L (logical) and T (time).</p> <p>The Natural data format A is restricted to 1073739357 bytes, Natural data format B is restricted to 536869678 bytes.</p> <p>DYNAMIC indicates a dynamic parameter and applies to the Natural data formats A and B.</p>
<b>Prec</b>	<p>Only applies to Natural data formats N (numeric) and P (packed). Optional.</p> <p>The precision of the parameter, that is, the number of digits after the decimal point.</p>

Field	Description
<b>Dimension</b> 1/2/3	<p>Only applies to arrays. Optional.</p> <p>The first, second and third dimension of the parameter.</p> <p>An X-array or an X-group array is specified by entering an asterisk (*) for a dimension.</p> <p>See also <i>Defining X-Arrays and X-Group Arrays</i>.</p>

The section below contains information on:

- [Defining Groups](#)
- [Defining X-Arrays and X-Group Arrays](#)
- [Special Considerations for Reliable RPC](#)
- [Special Considerations for Calling EntireX RPC Servers](#)

### Defining Groups

You only need to define a group structure for a client Natural object that calls a non-Natural object located on an EntireX RPC server. The group structure must correspond to the IDL definition in EntireX (see *Special Considerations for Calling EntireX RPC Servers*). A group structure is not required for a client Natural object that calls a subprogram located on a Natural RPC server.

Group arrays and X-group arrays passed from a client Natural object to a stub subprogram must be contiguous. Therefore, we strongly recommend that you always pass a complete array to the stub subprogram by using asterisk (\*) notation for all dimensions. We also strongly recommend that you use identical data definitions in the client Natural program, the stub subprogram and the server program.

See also *Example 2* for an example of a group definition.

### Defining X-Arrays and X-Group Arrays

If any dimension of a parameter is extensible, all other dimensions of the parameter are also extensible. If you define extensible and fixed dimensions for a parameter in a subprogram, the stub generation function issues a warning and automatically changes the fixed dimension to an extensible dimension as demonstrated in *Example 3*. In a group structure, you can define either an extensible or a fixed dimension for each level. There is no automatic change of a fixed dimension to an extensible dimension between levels.

Natural RPC only supports extensible upper bounds. All X-arrays and X-group arrays in the generated DEFINE DATA PARAMETER area of the stub subprogram are therefore defined as (1: \*).

 **Caution:** If you generate a stub subprogram from a subprogram or a PDA that contains an X-array or X-group array with an extensible lower bound, the extensible lower bound will be converted to an extensible upper bound.

For an example of a group with an extensible dimension, see [Example 3](#).

### Special Considerations for Reliable RPC

If you want to use reliable RPC and your parameter definitions do not contain group structures, you must set `COMPAT IDL` before generating the stub subprogram.

### Special Considerations for Calling EntireX RPC Servers

The attribute definitions on the **Stub Generation** screen depend on the perspective of the client. Conversely, the parameter direction in the IDL definition depends on the perspective of the server. This means:

- `OUT` on the **Stub Generation** screen corresponds to `IN` in the IDL definition.
- `IN` on the **Stub Generation** screen corresponds to `OUT` in the IDL definition.

If you want to call an EntireX RPC server and the parameter definitions on the **Stub Generation** screen contain group structures, group structure and attribute definitions on the **Stub Generation** screen must correspond to the group structure and parameter direction in the IDL definition.

If you want to call an EntireX RPC server and the corresponding IDL file does not contain group structures, it is recommended to set `COMPAT IDL` before generating the stub subprogram. In this case, the attribute definitions on the **Stub Generation** screen must correspond to the parameter direction in the IDL definition.

## Examples of Stub Generation

---

This section provides examples of Natural subprograms and the stub subprograms (interface objects) generated from them.

The parameter definitions indicated below are extracted from example subprograms, which are supplied in the Natural system library `YSRPC`.

### Example 1

The following DEFINE DATA PARAMETER area (example subprogram TESTS5) shows four modifiable parameters and the corresponding parameter definitions on the **Stub Generation** screen:

```

DEFINE DATA
PARAMETER
  01 #IDENTIFIER (A10)
  01 #N-OF-ID (I4)
  01 #FREQ (P5.2)
  01 #A100 (A100/5,4)
    
```

Stub Generation								
	Level	Attr	Type	Length	Prec	Dimension 1	Dimension 2	Dimension 3
1	01	M	A	10				
2	01	M	I	4				
3	01	M	P	5	2			
4	01	M	A	100		5	4	

### Example 2

The following DEFINE DATA PARAMETER area (example subprogram TESTS6) shows a nested group structure and the corresponding parameter definitions on the **Stub Generation** screen:

```

DEFINE DATA
PARAMETER
  01 GROUP-1(10)
    02 A (A20)
    02 B (A20)
    02 GROUP-2(20)
      03 C (A10/5)
      03 D (A10)
  01 LINE (A) DYNAMIC
    
```

Stub Generation								
	Level	Attr	Type	Length	Prec	Dimension 1	Dimension 2	Dimension 3
1	01	M	G			10		
2	02	M	A	20				
3	02	M	A	20				
4	02	M	G			20		
5	03	M	A	10		5		
6	03	M	A	10				

Stub Generation							
7	01	M	A	DYNAMIC			

### Example 3

The following DEFINE DATA PARAMETER area (example subprogram TESTS7) shows a nested group structure with extensible dimensions and the corresponding parameter definitions on the **Stub Generation** screen.

```

DEFINE DATA
PARAMETER
  01 GROUP-1(10)
    02 A (A20)
    02 B (A20)
    02 GROUP-2(0:*)
      03 C (A10/5)
      03 D (A10)
  01 LINE (A) DYNAMIC
    
```

Stub Generation								
	Level	Attr	Type	Length	Prec	Dimension 1	Dimension 2	Dimension 3
1	01	M	G			10		
2	02	M	A	20				
3	02	M	A	20				
4	02	M	G			*		
5	03	M	A	10		5		
6	03	M	A	10				
7	01	M	A	DYNAMIC				



# 108

## Generating Multiple Stubs

---

- Using the SYSRPC SGMASS Command ..... 736
- Name Specification and Compression ..... 737

You can generate single or multiple stub subprograms (interface objects) in either online or batch mode by using the command `SYSRPC SGMASS`.

You generate stub subprograms from subprograms or parameter data areas (PDAs).

## Using the SYSRPC SGMASS Command

---

You can enter the command `SYSRPC SGMASS` at any `NEXT` or `MORE` command prompt for generating stub subprograms online.

The section below contains information on:

- [Syntax of SYSRPC SGMASS](#)
- [SYSRPC SGMASS Report](#)

### Syntax of SYSRPC SGMASS

The syntax that applies to `SYSRPC SGMASS` is illustrated in the diagram below:

```
SYSRPC SGMASS [name] [compression]
```

The syntactical items *name* and *compression* are explained in the section *Name Specification and Compression*.

### SYSRPC SGMASS Report

The command `SYSRPC SGMASS` produces a report that lists the stub subprograms (objects) generated with the command as shown in the following example:

```
Page          1                               2006-05-24 16:09:17
SYSRPC - Stub Generation in Library SAGTEST

Generation Criteria:

Object name or range: RPC*
Compression: 1

Generation Results:

Number of objects found:      8
Maximum send length: 200228
```

```
Maximum receive length: 1024192
```

Object	Type	Send Length	Receive Length	Message
RPCCALL1	N	209	202	
RPCCALL2	N	219	240	Compression=2
RPCCALL3	N	204	193	
MORE				

The report is organized in three sections, which contain the following information:

- **Generation Criteria:**

The criteria based on which the stub subprogram(s) were generated: a single object name or a range of names (here: `RPC*`) and the compression (here: 1).

- **Generation Results:**

The number of objects selected for the stub generation.

The maximum buffer sizes all generated stub subprograms require for sending and receiving data from the client.

- **Object List:**

The name and type (here: N for type subprogram) of each generated stub subprogram. The buffer sizes each object requires for sending (**Send Length**) and receiving (**Receive Length**) data from the client. A possible comment on each stub generation in the **Message** column. In the example above, `Compression=2` indicates that object `RPCCALL2` was not generated with Compression 1 as requested in the command. The object list is sorted in alphabetical order of object names.

If the MORE prompt appears, choose `ENTER` to scroll to the end of the report.

If the stub generation fails for single or multiple objects, the report shows the number of objects affected and appropriate error messages.

## Name Specification and Compression

You can specify the objects (subprograms or PDAs) to be selected for stub generation and the type of compression to be used:

- Name

- [Compression](#)

## Name

You can specify an object name or a range of names. The specification of an object name or a range of names is optional.

 **Caution:** If you do not specify an object name or a range of names, with few exceptions (see below), all subprograms or PDAs in the current library will be converted to stub subprograms.

Valid name specifications are described below where *value* is any combination of one or more alphanumeric characters:

Input	Objects Selected
*	All subprograms or PDAs.  This is the default setting.
<i>value</i>	A subprogram or a PDA with a name equal to <i>value</i> .
<i>value</i> *	All subprograms or PDAs with names that start with <i>value</i> .
<i>value</i>	All subprograms or PDAs with names less than or equal to <i>value</i> .
<i>value</i>	All subprograms or PDAs with names greater than or equal to <i>value</i> .

## Exceptions to Names

In the Natural system library SYSRPC, SYSRPC SGMASS exempts from stub generation all subprograms with names that start with any of the following prefixes: RDS, RPC, NAT, NAD or NSC.

In user libraries, SYSRPC SGMASS exempts from stub generation the subprogram NATCLTGS.

## Compression

You can specify any of the following compression types: 0, 1, 2. The specification of compression is optional. The default type used for stub generation is 1.

See also *Using Compression* described in *Operating a Natural RPC Environment in the Natural Remote Procedure Call (RPC)* documentation.

# 109

## Calculating Size Requirements

---

- Using the SYSRPC CSMASS Command ..... 740
- Name Specification and Compression ..... 742

The command `SYSRPC CSMASS` is used to calculate the buffer size stubless RPC calls require for sending data from the client to the server or vice versa. The indication of the size helps you configure the middleware layer used; for example, the Broker attribute file when EntireX Broker is used.

If desired, you can also perform size calculations for stub subprograms (interface objects), even though sizes are already calculated when generating the stub subprograms.

`SYSRPC CSMASS` can be used in online or batch mode.

## Using the SYSRPC CSMASS Command

---

You can enter the command `SYSRPC CSMASS` at any `NEXT` or `MORE` command prompt for calculating size requirements online.

The section below contains information on:

- [Syntax of SYSRPC CSMASS](#)
- [SYSRPC CSMASS Report](#)

### Syntax of SYSRPC CSMASS

The syntax that applies to the command `SYSRPC CSMASS` is illustrated in the diagram below:

```
SYSRPC CSMASS [name] [compression]
```

The syntactical items *name* and *compression* are explained in the section *Name Specification and Compression*.

### SYSRPC CSMASS Report

The command `SYSRPC CSMASS` produces a report that indicates the send and receive length requirements of the subprograms (objects) specified with the command as shown in the following example:

```
Page          1                               2006-05-24 15:54:12
SYSRPC - Calculation of Buffer Sizes for Stubless RPC in Library SAGTEST

Calculation Criteria:
Object name or range: RPC*
Compression: 1
```

## Calculation Results:

```

Number of objects found:      8
  Maximum send length: 200228
  Maximum receive length: 1024192

```

Object	Type	Send Length	Receive Length	Message
RPCCALL1	N	209	202	
RPCCALL2	N	219	240	Compression=2
RPCCALL3	N	204	193	
MORE				

The report is organized in three sections, which contain the following information:

- **Calculation Criteria:**

The criteria based on which the calculation was made: a single object name or a range of names (here: RPC\*) and the compression (here: 1).

- **Calculation Results:**

The number of objects selected for the size calculation.

The maximum buffer sizes all selected objects require for sending and receiving data from the client.

- **Object List:**

The name and type (here: N for type subprogram) of each object selected for the calculation. The buffer sizes each object requires for sending (**Send Length**) and receiving (**Receive Length**) data from the client. A possible comment on each object calculation in the **Message** column. In the example above, `Compression=2` indicates that object RPCCALL2 was not calculated with Compression 1 as requested in the command. The object list is sorted in alphabetical order of object names.

If the MORE prompt appears, choose ENTER to scroll to the end of the report.

If the size calculation fails for single or multiple objects, the report shows the number of objects affected and appropriate error messages.

## Name Specification and Compression

---

You can specify the objects (subprograms or PDAs) to be selected for size calculation and the type of compression to be used:

- Name
- Compression

### Name

You can specify an object name or a range of names. If you do not specify a name or a range of names, the size of all subprograms or PDAs contained in the current library will be calculated.

Valid name specifications are described below where *value* is any combination of one or more alphanumeric characters:

Input	Objects Selected
*	All subprograms or PDAs. This is the default setting.
<i>value</i>	A subprogram or a PDA with a name equal to <i>value</i> .
<i>value</i> *	All subprograms or PDAs with names that start with <i>value</i> .
<i>value</i> <	All subprograms or PDAs with names less than or equal to <i>value</i>
<i>value</i> >	All subprograms or PDAs with names greater than or equal to <i>value</i> .

### Compression

You can specify any of the following compression types: 0, 1, 2. The specification of compression is optional. The default type used for stub generation is 1.

See also *Using Compression* described in *Operating a Natural RPC Environment* in the *Natural Remote Procedure Call (RPC)* documentation.

# 110

## Parameter Maintenance

---

- Invoking Parameter Maintenance ..... 744
- Specifying NTRPC/RPC Keyword Subparameters ..... 744

**Applies to client sessions only.**

The **Parameter Maintenance** function is used to dynamically (within a session) modify keyword subparameters of the `RPC` profile parameter or the `NTRPC` macro.

 **Caution:** The parameter modifications are only retained as long as the user session is active; they are lost when the session is terminated.

---

## Invoking Parameter Maintenance

---

▶ **To invoke the Parameter Maintenance function**

- 1 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
PM
```

The **Client Parameter Maintenance** screen appears.

- 2 Modify the values of the input fields: see [Specifying NTRPC/RPC Keyword Subparameters](#).
- 3 Choose PF3 (Exit) to save any modifications and exit the **Client Parameter Maintenance** screen.

Or:

Choose PF12 (Canc) to exit without saving any parameter modifications.

The **Client Maintenance** menu appears.

---

## Specifying NTRPC/RPC Keyword Subparameters

---

In the input fields provided on the **Client Parameter Maintenance** screen, you can modify the settings of the keyword subparameters of the `NTRPC` macro or the `RPC` profile parameter described in the table below:

Field	Explanation
<b>Timeout</b>	Specifies the number of seconds the client is to wait for an RPC server response.  See also the <code>NTRPC/RPC</code> keyword subparameter <code>TIMEOUT</code> described in the <i>Parameter Reference</i> documentation.
<b>Try alternative servers</b>	Specifies whether an RPC client is to try to execute a service on an alternative server (ON) or not (OFF). See also <i>Using an Alternative Server</i> in the <i>Natural Remote Procedure Call (RPC)</i> documentation.

Field	Explanation
	See also the NTRPC/RPC keyword subparameter TRYALT described in the <i>Parameter Reference</i> documentation.
<b>Compression for auto remote RPC</b>	Specifies the compression type for an automatically generated RPC call; see <i>Using Compression</i> described in the <i>Natural Remote Procedure Call (RPC)</i> documentation.  See also the NTRPC/RPC keyword subparameter COMPR described in the <i>Parameter Reference</i> documentation.  For more information on automatic RPC execution, see <i>Working with Automatic Natural RPC Execution</i> in the <i>Natural Remote Procedure Call (RPC)</i> documentation.

For further information on parameter settings, see *Keyword Subparameters* in the section *RPC - Remote-Procedure-Call Settings*, in the *Parameter Reference* documentation.



# 111 Server Command Execution

---

- Using Server Command Execution ..... 748
- Pinging a Server ..... 749
- Terminating a Server ..... 751

The SYSRPC utility provides the server execution commands ping and terminate. They are used to control active servers that have been defined in the service directory. The ping command sends an internal message to the server to verify a server connection. Terminate either sends an internal message to the server requesting termination of a single server task, or issues a command to EntireX Broker requesting termination of all server tasks associated with an EntireX Broker service.

The server execution commands reference the service directory in the library that is defined with the NTRPC/RPC keyword subparameter RPCSDIR (see the *Parameter Reference* documentation). If RPCSDIR is not set (this is the default), the library where you are currently logged on is used. The name of the library is indicated in the upper right corner of the **Server Command Execution** screen shown in the following section.

## Using Server Command Execution

### ▶ To use Server Command Execution

- 1 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
XC
```

- 2 Choose ENTER.

The standard view of the **Server Command Execution** screen appears similar to the following example:

```

14:09:38          *** NATURAL Remote Procedure Call ***          2006-01-27
                               Server Command Execution                               SYSRPC

Node                Server                Message
1  ETB045
2
3                   NRPC001                Natural
4                   NRPC002
5  *LOCTRAN
6                   NRPC001-LOGICAL
7  LOGBROKER=NODE
                   NRPC002

-----
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help ERR  Exit <   -H   +H   -P   +P   TOP  BOT  >   Canc
    
```

The standard view displays the columns **Node**, **Server** and **Message**. The fields under the column **Message** are truncated and display a maximum of 8 characters.

- If you choose PF4 or enter the less than (<) sign in the command line at the bottom of the screen, the extended message view of the **Server Command Execution** screen is displayed similar to the following example:

```

14:14:08          *** NATURAL Remote Procedure Call ***          2006-01-27
                   Server Command Execution                      SYSRPC

   Server          Message
1
2   NRPC001        Natural RPC Server 6.3.1.0 on WNT-x86
3   NRPC002
4
5   NRPC001-LOGICAL
6
7   NRPC002

Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help  ERR   Exit  <   -H   +H   -P   +P   TOP  BOT  >   Canc

```

The extended view allows you to display a maximum of 50 characters of message text in the **Message** column. This view does not display the **Node** column and the fields under the **Server** column are truncated and display a maximum of 16 characters (the standard view shows 30 characters).

If you choose PF11 or enter the greater than (>) sign in the command line, the standard view of the **Server Command Execution** screen is displayed as shown in [Example of a Standard View](#).

## Pinging a Server

You can ping a server from the standard or extended message view of the **Server Command Execution** screen.

The following instructions are an example of how to ping a server from the standard view.

For an alternative method of pingging an RPC server, see Application Programming Interface USR2073N described in the *Natural Remote Procedure Call (RPC)* documentation.

► **To ping a server**

- 1 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
XC
```

- 2 Choose ENTER.

The standard view of the **Server Command Execution** screen is displayed.

- 3 In the empty column between the sequence number and the **Node** column, in the line(s) which belong to the server(s) to be pinged, enter the following command:

```
PI
```

as shown in the example below:

```
09:49:41          *** NATURAL Remote Procedure Call ***          2006-01-27
                               Server Command Execution                               SYSRPC

Node                Server                Message
1      ETB045
2      PI                NRPC001
3                NRPC002
4      *LOCTRAN
5      PI                NRPC001-LOGICAL
6      LOGBROKER=NODE
7                NRPC002
```

- 4 Choose ENTER. The server(s) return the message:

```
Server V.R.S.PL on operating system
```

where

*Server* denotes the type of server; *V.R.S.PL* denotes the 1-digit version *V*, the 1-digit release *R*, the 1-digit system maintenance level *S* and the 1- or 2-digit patch level *PL* of the server; *operating system* denotes on which operating system the server runs.

Example message:

Natural RPC Server 6.3.1.0 on WNT-x86

If pinging the server fails and an error occurs instead, you can choose PF2 (ERR) to display RPC-related Natural and EntireX Broker messages as described in *Using the RPCERR Program (Monitoring the Status of an RPC Session, Natural Remote Procedure Call (RPC) documentation)*.

- 5 To display more of the message text which appears truncated in the standard view of the **Server Command Execution** screen (see also [Example of a Standard View](#)) proceed as follows:

Choose PF4.

Or:

In the command line, enter the less than (<) sign.

## Terminating a Server

The SYSRPC utility provides two commands to terminate a server: TE (Terminate Server) and TS (Terminate EntireX Broker Service).

TE terminates a single server task by sending an internal message to the server. If a server is associated with multiple server tasks (including replicas on mainframe platforms), you can either terminate each server task separately by using TE, or terminate all server tasks in one go by using the TS command.

TS terminates all server tasks associated with an EntireX Broker service by calling EntireX Broker's Command and Information Services (ETBCIS; for details, see the EntireX documentation). The term *service* here summarizes all server tasks that run with the same server name on the same or on different platforms.

You can terminate server tasks from the standard or extended message view of the **Server Command Execution** screen.

The following section provides instructions for terminating a single server task or an EntireX Broker service from the standard view.

For alternative methods of terminating servers, see *Terminating a Natural RPC Server* and the Application Programming Interface USR2073N, and *Terminating the EntireX Broker Service* and the Application Programming Interface USR2075N described in the *Natural Remote Procedure Call (RPC) documentation*.

### ▶ To terminate a single server task

- 1 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
XC
```

The standard view of the **Server Command Execution** screen is displayed.

- 2 In the empty column between sequence number and **Node** column, in the line which belong to the server to be terminated, enter the following command:

```
TE
```

(This is similar to entering the command `PI` as show in the [example](#) of pinging a server.)

- 3 Choose `ENTER`.

The server returns the message:

```
Terminating Server V.R.S.PL on operating system
```

where

*Server* denotes the type of server; *V.R.S.PL* denotes the 1-digit version *V*, the 1-digit release *R*, the 1-digit system maintenance level *S*, and the 1- or 2-digit patch level *PL* of the server; *operating system* denotes on which operating system the server runs.

Example message:

```
Terminating Natural RPC Server 6.3.1.0 on WNT-x86
```

If terminating the server fails and an error occurs instead, you can choose `PF2 (ERR)` to display RPC-related Natural and EntireX Broker messages as described in *Using the RPCERR Program (Monitoring the Status of an RPC Session, Natural Remote Procedure Call (RPC) documentation)*.

To display more of the message text which appears truncated in the standard view of the **Server Command Execution** screen:

Choose `PF4`.

Or:

In the command line, enter the less than (<) sign.

- 4 If the **Logon** option is set in the service directory, logon data (user ID, password and library name) is sent to the server with the `TE` command, as is usual for remote `CALLNAT` execution. The **Security Token Data** window pops up and requests input of user ID and password if no Natural Security is installed on the client side and no logon data is set with the Application Programming Interface `USR1071N` for the current Natural session. See also `USR1071N` described in *Using Security, Using Natural RPC with Natural Security*, in the *Natural Remote Procedure Call (RPC) documentation*.

If `LOGONRQ=ON` (see also *Using Security* in the *Natural Remote Procedure Call (RPC) documentation*) has been set on the server side, logon data must be sent from the client with the `TE` command.

If Natural Security is installed on the server, the logon data transferred must enable a logon to the Natural system library `SYSRPC`.

▶ **To terminate an EntireX Broker service**

- 1 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
XC
```

The standard view of the **Server Command Execution** screen is displayed.

- 2 In the empty column between sequence number and **Node** column, in the line which belong to the server to be terminated, enter the following command:

```
TS
```

(This is similar to entering the command `PI` as show in the [example](#) of pinging a server.)

- 3 Choose `ENTER`.

The **SYSRPC - Terminating EntireX Broker Service** window appears.

- 4 If required for the logon, enter the appropriate user ID and password for EntireX Broker.

If you want to terminate server tasks that are involved in a conversation, in the **Terminate immediately** field, enter a `Y` to request immediate termination. If you enter an `N` (this is the default setting), all server tasks involved in a conversation remain operational.

If you do not want this window to appear repeatedly during the current `SYSRPC` session, choose **Do not show this window again**.

- 5 Choose `ENTER` to terminate the EntireX Broker service.



# 112 Remote Directory Maintenance

---

- Using Remote Directory Maintenance ..... 756
- Fields on the Editor Screen of a Remote Directory ..... 757
- Commands for Remote Directory Maintenance ..... 758

The **Remote Directory Maintenance** function is used to maintain a remote directory in order to connect the client's calling program to a subprogram on a server.

For further information on how to apply the **Remote Directory Maintenance** function, refer to *Specifying RPC Server Addresses (Operating a Natural RPC Environment)*, *Using a Remote Directory Server (RDS)* and *Natural RPC Terminology* described in the *Natural Remote Procedure Call (RPC)* documentation.

## Using Remote Directory Maintenance

---



**Caution:** If you create a new remote directory by entering code C (see the instructions below), the entries of an existing directory will be overwritten.

### ▶ To use the Remote Directory Maintenance function

- 1 In the **Code** field of the **Client Maintenance** menu, enter the following command:

```
RD
```

- 2 Choose ENTER.

A window appears.

- 3 In the input field, enter either of the following commands:

```
C
```

to create a directory or

```
M
```

to modify a directory.

- 4 Choose ENTER.

An additional window appears.

- 5 Enter an expiration time in seconds (see also *Expiration Time* below) and choose ENTER.

An editor screen similar to the example below appears:

```

-----S 01-----Columns 001 072
=====>
***** NODE          L T    SERVER    L T    LIBRARY    L T    PROGRAM    L T
***** ***** top of data *****
000001 ETB01          Y B    NRPC2301          SYSTEM          SUB1
000002
000003
000004
000005
000006
000007 ETB01          NRPC2301 Y    SYSTEM          SUB7
000008
000009
000010
000011
000012
000013 ETB01          Y    NRPC2301          SYSTEM          SUB6
000014
000015
000016 ETB01          Y    NRPC2301          SYSTEM          SUB3
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help          Quit  Save  Find  Chng  Up    Down          Canc

```

## Expiration Time

The remote directory data is loaded at runtime. The expiration time in seconds determines the period of validity of this data. If directory data is requested after the expiration time set, it will automatically be reloaded. If expiration time is set to 0, the remote directory data will not be reloaded.



**Note:** If you create a new directory (code C), you invoke an input line by entering the line command I at the beginning of the line top of data. See also [Line Commands](#) below.

## Fields on the Editor Screen of a Remote Directory

The fields contained in the editor screen of **Remote Directory Service** maintenance are identical to the fields described in [Fields on the Service Directory Screen](#) in the section *Service Directory Maintenance*. The field L is the equivalent to the field **Logon**.

In addition to the fields provided on the **Service Directory Maintenance** screen, the field T (Transport) appears: enter a B for EntireX Broker.

## Commands for Remote Directory Maintenance

This section contains information on the commands provided on the editor screen of **Remote Directory Service** maintenance.

- [Line Commands](#)
- [Direct Commands and PF Keys](#)

### Line Commands

The line commands available on the editor screen can be used to copy, move or delete single or multiple lines that contain field values.

Enter a line command at the beginning of a line, that is, overwrite the sequential number and choose ENTER.

See also [To copy or move a block of lines](#) and the direct command [RESET](#).

Line Command	Function
A	Copies or moves the line(s) marked with C, CC, M or MM below the line in which the command was entered.
B	Copies or moves the line(s) marked with C, CC, M or MM above the line in which the command was entered.
C	Marks the single line to be copied.
CC	Marks the block of lines to be copied.
D( <i>n</i> )	Deletes one or <i>n</i> lines beginning with the line in which the command was entered. <i>n</i> can be in the range from 1 to 9.
DD	Marks and deletes a block of lines.  Mark the block of lines by entering the command in the first and last line of the block, and choose ENTER to execute the command.
I( <i>n</i> )	Inserts one or <i>n</i> empty lines below the line in which the command was entered. <i>n</i> can be in the range from 1 to 9.
M	Moves a single line below the line in which the command was entered.
MM	Marks the block of lines to be moved.

#### ▶ To copy or move a block of lines

1 At the beginning of the line where the block starts, enter either of the following line commands:

```
CC
```

to copy the block or

```
MM
```

to move the block.

- 2 At the beginning of the line where the block ends, enter either of the following line commands:

```
CC
```

to copy the block or

```
MM
```

to move the block.

- 3 Choose ENTER.

The block of lines is marked which is indicated by the message: Block is pending.

- 4 At the beginning of the line below or above which you want to place the block, enter either of the following line commands:

```
A
```

to copy or move the block *below* the specified line or

```
B
```

to copy or move the block *above* the specified line.

- 5 Choose ENTER.

The block of lines is copied or moved below or above the specified line.

## Direct Commands and PF Keys

The following direct commands and PF keys are provided on the editor screen:

Direct Command	PF Key	Function
RESET		Removes the line marks set with a line command (see <a href="#">Line Commands</a> ) or with the direct command CHANGE (see below).
TOP		Scrolls to the beginning of the list.
BOT		Scrolls to the end of the list.
FIND <i>string</i>		Scans the editor for a <i>string</i> of characters, for example: FIND ETB1. Choose PF5 (Find) to scan for the next occurrence.
CHANGE <i>string1</i> <i>string2</i>		Replaces character <i>string1</i> by <i>string2</i> , for example: CHANGE ETB1 ETB2. Choose PF6 (Chng) to replace the next occurrence.

Direct Command	PF Key	Function
	PF1	Help. Invokes the online help.
	PF3	Quit. Saves any modifications and exits the editor screen.
	PF4	Saves any modifications.
	PF5	Find. Scans for the next occurrence of the character string specified with the direct command <code>FIND</code> (see above).
	PF6	Change. Replaces the next occurrence of the character string specified with the direct command <code>CHANGE</code> (see above).
	PF7	Up. Scrolls one page backward.
	PF8	Down. Scrolls one page forward.
	PF12	Cancel. Exits the editor screen without saving modifications.

# 113 SYSTP Utility

---

The SYSTP utility is used to monitor and control characteristics of Natural that are specific to TP monitors.

The SYSTP utility provides functions that are available in most environments, under most TP monitors. They are described in *General SYSTP Functions*.

The SYSTP utility provides additional functions for the following TP monitors:

- CICS
- IMS TM
- TIAM and *openUTM*

These environment-dependent functions are described in the relevant sections of the *SYSTP Utility* documentation.



**Note:** In this documentation, *openUTM* is referred to as UTM.

- [Invoking SYSTP and Executing Functions](#)
- [Using SYSTP Utility Screens](#)
- [General SYSTP Functions](#)
- [SYSTP Functions under CICS](#)
- [SYSTP Functions under IMS TM](#)
- [SYSTP Functions under TIAM and UTM](#)
- [SYSTP in Batch for CICS Sessions](#)



# 114

## Invoking SYSTP and Executing Functions

---

This section provides instructions for invoking the SYSTP utility and executing a SYSTP utility function. You can execute a SYSTP utility function by using either a SYSTP utility menu or a SYSTP direct command.

### ▶ To invoke SYSTP and execute a menu function

- 1 At any command prompt, enter the following system command:

```
SYSTP
```

The **Main Menu** of the SYSTP utility appears.

- 2 Execute a SYSTP function by entering the character code that corresponds to the function required in the **Code** field and pressing `ENTER`.

If you enter `E`, the menu for environment-dependent TP-monitor functions appears. This function is not available under Com-plete and TSO.

### ▶ To issue a SYSTP direct command

- At any command prompt or in batch mode, use the `SYSTP` command followed by the function code that corresponds to the SYSTP menu function required. You can enter multiple function codes in the sequence in which they are executed from a menu. If you enter multiple function codes, separate them by periods (.).

For example:

```
■ SYSTP M.A
```

Invokes the **Natural Monitor Menu** (function code **M** in the SYSTP **Main Menu**) and activates the Natural monitor (function code **A** in **Natural Monitor Menu**).

```
■ SYSTP S.A.C
```

Invokes the **Slot Size Calculation** screen after:  
**S** was executed from the **SYSTP Main Menu**,  
**A** was executed from the **Natural Swap Main Menu**, and  
**C** was executed from the **Natural Swap Administration Menu**.

See also *[SYSTP in Batch for CICS Sessions](#)*.

# 115

## Using SYSTP Utility Screens

---

The functions provided in a SYSTP utility menu can be invoked by either entering a function code or pressing the PF key (if available) that corresponds to the function required.

The line commands and PF keys (or corresponding direct commands) which are available on many of the SYSTP utility screens are described in the following table. You enter a line command in the C column next to the required list item on the screen. You enter a direct command in the Command line.

PF Key or Direct Command	Line Command	Function
? or *	? or *	If entered in the C column, all line commands available on the current SYSTP screen are displayed.  If entered in the Command line, all direct commands available within the SYSTP utility are displayed.
PF1 or HELP	-	Provides help information about the SYSTP function currently used.
PF3	.	Exits the current screen and returns to the previous level/screen.
PF5	/ or P	Positions the line in which the line command is entered (or where the cursor is placed) to the top of the screen.
PF4	S or U	Displays detailed information about the item contained in the line, in which the line command is entered (or where the cursor is placed).
PF6	-	Scrolls to the beginning of a list.

PF Key or Direct Command	Line Command	Function
or --		
PF7 or -	-	Scrolls one page backward in a list.
PF8 or +	-	Scrolls one page forward in a list.
PF9 or ++	-	Scrolls to the end of a list.

The headers of all SYSTP statistics screens contain the following information:

- The **User** field with the ID of the current user as assigned by the Natural system variable \*USER (see the *System Variables* documentation).
- The **TID** field with the terminal ID assigned to the current user by the Natural system variable \*INIT-ID (see the *System Variables* documentation).

# 116

## General SYSTP Functions

---

▪ Natural Monitoring (SYSMON) .....	768
▪ Natural Print/Work Files (SYSFILE) .....	771
▪ Natural Swap Information .....	772
▪ Buffer Usage Statistics (BUS) .....	776
▪ Natural Sub-Systems and Roll Server Information .....	778
▪ Natural Thread Usage Statistics .....	779
▪ Natural License Information .....	781

## Natural Monitoring (SYSMON)

---

The Natural monitoring function can be used to view statistics about the Natural programs and terminals used during the current Natural session.

The scope of the Natural monitor is determined by the area in which the statistical data is collected:

- When a global or local monitor buffer pool is used (as assigned by the `BPI` profile parameter or the `NTBPI` macro of the `NATPARM` parameter module; see the *Parameter Reference* documentation), program and terminal statistics of all Natural sessions that share this buffer pool are collected.
- When a monitor buffer is used within the thread (as determined by the `MONSIZE` profile parameter; see the *Parameter Reference* documentation), only program statistics and information about the terminal of the current Natural session are collected. For a monitor buffer, we recommend a minimum value of 64 KB.

In addition to defining the scope for the Natural monitor, you have to set the `RDCSIZE` profile parameter, which activates the Natural Data Collector.



**Caution:** When active, the monitoring function can affect overall system performance.

When you invoke the **Natural Monitoring (SYSMON)** function, the **Natural Monitor Menu** appears, which provides the following functions:

- [Activate/Deactivate Monitor](#)
- [Monitor Status Information](#)
- [Display Program/Terminal Statistics](#)

### Activate/Deactivate Monitor

With these functions you can activate or deactivate the monitor function.

When the monitor function is activated, it begins collecting statistical information of current session(s). Once the monitor function is deactivated, a statistical summary is written to the system log file.

## Monitor Status Information

This function provides statistical information about whether monitoring data is collected in a monitor buffer pool and/or the space allocated with the `MONSIZE` parameter, and indicates the size and the address of the allocated space.

## Display Program/Terminal Statistics

You can view statistical information about all Natural programs that have been executed since the monitor was started and the terminals that have been activated since then.

### ▶ To execute the program or terminal statistics function

- 1 In the **Code** field of the **Natural Monitor Menu**, enter `P` (for program statistics) or `T` (for terminal statistics).

You can specify selection criteria for the programs/terminal and/or libraries to be viewed: In the **Name of LTERM or Program** field and/or the **Name of Library** field, enter the name of the required item or specify a range of names by using asterisk (\*) and/or a wildcard (?). If you leave the fields empty or only enter an asterisk (\*), all programs/terminals and libraries are selected.

Examples of Name Ranges:

- \*CD selects ABCD, ACD.
- AB\* selects AB, AB1, ABC, ABEZ.
- ABC? selects ABCA, ABCZ.
- A?C\*Z selects ABCZ, AXCBBBZ and AN CZ.

- 2 When you have finished specifying selection criteria and press `ENTER`, a statistics overview screen similar to the example below appears:

13:44:35		***** NATURAL SYSTP UTILITY *****							2008-08-29		
User SAG		- Natural Monitor Program Statistics -							TID TCK8		
C	Program	Library	NAT-	ADA-	Ext-	Mean-	Screen	I/O	User	Sys	Fetch
*	*	*	time	time	time	time	No	KB	Acc	Acc	
_	MONMNU1M	SYSTP	0	0	0	0.0	18	9	6	0	19
_	SYSMAP0M	SYSTP	0	0	0	0.0	0	0	0	0	52
_	S2SCOM01	SYSTP	0	0	0	0.0	0	0	0	0	36
*	MONMNU1P	SYSTP	0	0	0	0.0	1	0	28	0	6
_	MONLST1P	SYSTP	0	0	0	0.0	3	0	21	0	12
_	SYSMAP1M	SYSTP	0	0	0	0.0	19	13	2	0	20
_	NAT00059	SYSTP	0	0	0	0.0	0	0	0	0	346
_	STPMNU1P	SYSTP	0	0	0	0.0	4	0	30	4	8
_	NAT42004	SYSTP	0	0	0	0.0	0	0	0	0	9
_	STPMMM1M	SYSTP	0	0	0	0.0	7	5	4	0	9

```

_ SYSTPRET SYSTP      0  0  0  0.0  0  0  13  0  4
_ NAT00030 SYSTP      0  0  0  0.0  0  0   0  0  4
_ NSCC0002 SYSTP      0  0  0  0.0  0  0  14  0  7
_ LOGON     SYSTP      0  0  0  0.0  0  0   0  0  1
  Top of List                                     1 of 3
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Cont Help Menu Exit Sel      --  -  +  ++  Term  Canc

```

This screen lists all programs/terminals and libraries that have been active in your current Natural session.

PF4 invokes a **Selection** window that can be used to specify selection criteria (see Step 1) to reduce the list of items displayed on the screen.

You can use PF10 to toggle between the program statistics and terminal statistics screen.

The columns contained on a statistics overview screen and the corresponding field names on a detailed statistics screen (see column C below) are explained in the following table:

Column	Corresponding Field	Explanation
<b>C</b>	-	<p>Only applies to the statistics overview screen.</p> <p>This input field can be used to invoke a detailed statistics screen for a selected program/terminal:</p> <p>Next to the list item about which you require more detailed statistics information, enter any character and press ENTER.</p> <p><b>Note:</b> If the statistics overview of active programs/terminals is displayed repeatedly, an asterisk (*) appears in the C column next to the program/terminal most active since the last repetition.</p>
<b>Program</b>	<b>Name of program</b>	<p>Only applies to program statistics.</p> <p>Name of the active program.</p>
<b>Terminal</b>	<b>Name of terminal</b>	<p>Only applies to terminal statistics.</p> <p>Name (ID) of the active terminal.</p>
<b>Current Program</b>	<b>Current program / library</b>	<p>Only applies to terminal statistics.</p> <p>Name of the executed program and the name of the library that contains this program.</p>
<b>Library</b>	<b>Name of library</b>	<p>Only applies to program statistics.</p> <p>Name of the library that contains the program that is executed.</p>

Column	Corresponding Field	Explanation
NAT-time	Time in Natural	Time in the Natural nucleus and in the interface.
ADA-time	Time in Adabas	Time waiting for response from Adabas.
Ext-time	Time in external program	Time needed by a user-written module.
Mean-time	Mean evaluation time	Elapsed time of one Natural screen transaction.
Screen I/O No	Number of Screen I/Os	Number of screen I/Os.
Screen I/O KB	Amount of data transmitted	Amount of data transferred to or from the screen.
-	Evaluation time > 3 sec	Only applies to the detailed statistics screen for a terminal.  Percentage of evaluation time longer than 3 seconds.
-	Evaluation time > 6 sec	Only applies to the detailed statistics screen for a terminal.  Percentage of evaluation time longer than 6 seconds.
User Acc	Number of user file accesses	Counter for accesses to Adabas user files.
Sys Acc	Number of system file accesses	Counter for accesses to Natural system file, including fetches.
Fetch	Number of fetches	Counter for total number of fetches.

## Natural Print/Work Files (SYSFILE)

This function provides information about available work files and print files.

You can also invoke this function with the system command `SYSFILE` described in the *System Commands* documentation.

This function can also be used in batch mode for CICS sessions.

When you invoke this function, the **Work File Information** screen appears with a list of all work and print files defined. The following information is provided for each file:

Column	Explanation
No.	Number of the work/print file.
Type	Type of assignment; that is, the operating system, TP monitor or Natural product file to which the work/print file is assigned.
Name	Name of the work/print file.
Recfm	Record format of the work/print file.

Column	Explanation
<b>Lrecl</b>	Logical record length of the work/print file (if applicable).
<b>Blksz</b>	Block size of the work/print file.
<b>Status</b>	One of the following statuses:  Available for Input/Output  or  Open for Input/Output

Under z/VSE, the logical-unit assignments are also displayed.

### Commands for Natural Print/Work Files

In addition to the commands described in *Using SYSTP Utility Screens*, the **Work File Information** screen provides the following PF keys and line command:

PF Key	Line Command	Function
PF10	-	Scrolls to the list of print files.
PF11	-	Scrolls to the list of work files.
-	D	Displays the corresponding Natural control block (work file area) in dump format (for internal use by Software AG technical support).

## Natural Swap Information

---

This function is only available under CICS and UTM.

The swap pool manager enables online monitoring and control of the Natural swap pool. This section describes how the swap pool manager is used rather than how the swap pool operates. For further information about the operation of the Natural swap pool, see *Natural Swap Pool* in the *Operations* documentation.

When you invoke this function, the **Natural Swap Main Menu** appears, which provides the following functions:

- Administration
- Debugging Facilities
- Information
- Maintenance Services

- [Status Information](#)

## Administration

- [Slot Size Calculation](#)
- [Change Swap Pool Status](#)
- [Update Reorg Control Data](#)

### Slot Size Calculation

This function displays the optimum values for the layout of the swap pool based on the current usage.

You can store these values to be used for a later initialization/reorganization (once they have been stored, they can also be maintained with the [Maintenance Services](#) function).

You can also initiate a swap pool reorganization using these values.

For further information, see the online help about this function.

### Change Swap Pool Status

This function is used to activate or deactivate the Natural swap pool. In addition, you can modify the wait time and the number of waits for swap pool synchronization.

For further information, see the online help about this function.

### Update Reorg Control Data

With this reorganization function, you can modify the most important parameters in swap pool management. You must enter a valid password to modify the values.

For further information, see the online help about this function.

### Debugging Facilities

This function is only available under UTM.



**Caution:** Do not use this function without prior consultation of Software AG technical support.

With this function, it is possible to activate or deactivate an internal screen debugging buffer. Activation of the screen debugging buffer is used to locate terminal I/O inconsistencies if they occur. The function records information about the last three terminal I/O sequences. The buffer has a size of 3 KB and is used in a wrap-around procedure.

In addition, you can activate/deactivate a trace function for asynchronous write operations to the Natural roll file.

For further information, see the online help about this function.

## Information

- [Show Addresses](#)
- [Show Summary of Buffer Usage](#)
- [Show Swap Pool Information](#)
- [Show Logical Swap Pools](#)
- [Show Reorg Control Data](#)
- [Show Swap Pool Usage](#)
- [Create Statistics List](#)

### Show Addresses

This function displays the addresses of various pools.

### Show Summary of Buffer Usage

This function is used to optimize the sizes of the various Natural buffers and the Natural user threads (see `MAXSIZE` in the *Operations* documentation). It activates, deactivates and displays a summary of Natural buffer usage.

The activation and deactivation of buffer statistics can only be performed with a valid password. For the display of buffer statistics, no password is required.

The buffers displayed are the same as those displayed by the function [Buffer Usage Statistics \(BUS\)](#).

### Show Swap Pool Information

This function displays information about the swap pool currently in use, including control/statistics data and memory sizes.

The individual items of information shown are explained in the online help about this function.

### Show Logical Swap Pools

This function displays the current table of logical swap pools.

On the table, you can mark a specific logical swap pool with any character to get additional information about it.

The individual items of information shown are explained in the online help about this function.

### Show Reorg Control Data

This function displays all information related to the swap pool reorganization.

The swap pool reorganization table is displayed in the left half of the screen. The table contains cumulative statistics about the comparative sizes between compressed Natural user threads and standard slot size. The table is cleared with each reorganization of the swap pool. The left half of the table shows how often and to what extent the user threads are larger than the standard slot size. The right half of the table shows how often and to what extent the user threads are smaller than the standard slot size. Sizes in this half of the table are expressed in units that are dependent on the factor specified by the swap pool manager.

In the row labeled **n**, count is taken of user threads which exceed/fall short of the standard slot size by over 9 pages/units. The average length of these user threads is displayed in the row labeled **Av.+n**.

The individual items of information shown are explained in the online help about this function.

### Show Swap Pool Usage

This function displays information about the usage of the swap pool since its initialization or the last reorganization.

The individual items of information shown are explained in the online help about this function.

### Create Statistics List

This function is used to create a list of the current swap pool usage statistics:

- Under UTM, the swap pool statistics are written to SYSLIST.
- Under CICS, the swap pool statistics are written to the CICS destination specified with the MSGDEST parameter of the NCMPRM macro described in the *TP Monitor Interfaces* documentation.

## Maintenance Services

- [Parameter Maintenance](#)
- [Password Maintenance](#)

### Parameter Maintenance

This function is used to change online the parameters for the initialization or reorganization of the swap pool.

The subfunctions as well as the individual items that can be modified are explained in the online help about this function.

The use of this function is password-protected (see below).

### Password Maintenance

This function is used to change or recover the password used for the [Parameter Maintenance](#) function.

The initial password is SYSTP.

### Status Information

With this function, you can display the current status of the Natural swap pool, of the summary of buffer usage and of the UTM screen debugging.

## Buffer Usage Statistics (BUS)

---

This function provides statistical information about the usage of Natural buffers: which buffers are allocated for the current Natural session, and how much buffer space is being used.

The **Total** figures at the end of the statistics list allow you to draw conclusions about the efficiency of buffer compression.

You can invoke this function either from the **SYSTP** menu or with the system command `BUS`.

When you invoke the function, a list is displayed showing all buffers which are actually being used in the current Natural session.

For each of these buffers, the following information is displayed on the **Buffer Usage Statistics** screen:

Column	Explanation								
<b>No.</b>	Buffers are numbered sequentially in order of allocation.								
<b>Name</b>	Name of the buffer. Only those buffers which have actually been requested in the current session are listed.								
<b>Type</b>	Type of the buffer: <table border="1" data-bbox="509 583 1474 1003"> <tr> <td><i>blank</i></td> <td>Fixed/static Natural buffer.</td> </tr> <tr> <td>V</td> <td>Variable buffer allocated in the Natural thread.</td> </tr> <tr> <td>0</td> <td>Variable buffer currently allocated outside the Natural thread. The buffer is copied and compressed into the thread at the next terminal I/O.</td> </tr> <tr> <td>P</td> <td>Physical buffer allocated in a Natural work pool (see also the WPSIZE profile parameter described in the <i>Parameter Reference</i> documentation). The buffer is released at the next terminal I/O.</td> </tr> </table>	<i>blank</i>	Fixed/static Natural buffer.	V	Variable buffer allocated in the Natural thread.	0	Variable buffer currently allocated outside the Natural thread. The buffer is copied and compressed into the thread at the next terminal I/O.	P	Physical buffer allocated in a Natural work pool (see also the WPSIZE profile parameter described in the <i>Parameter Reference</i> documentation). The buffer is released at the next terminal I/O.
<i>blank</i>	Fixed/static Natural buffer.								
V	Variable buffer allocated in the Natural thread.								
0	Variable buffer currently allocated outside the Natural thread. The buffer is copied and compressed into the thread at the next terminal I/O.								
P	Physical buffer allocated in a Natural work pool (see also the WPSIZE profile parameter described in the <i>Parameter Reference</i> documentation). The buffer is released at the next terminal I/O.								
<b>Size</b>	Size of the buffer (in bytes).								
<b>Used</b>	Number of bytes currently being used. This value is used for buffer compression in environments using threads (for example, CICS or UTM).								
<b>Perc. (Used)</b>	Percentage currently being used; that is, the value of the <b>Used</b> column in relation to the value of the <b>Size</b> column.								
<b>MaxUsed</b>	Maximum number of bytes which have been used in the course of the current session so far ( <i>not</i> the size being used at present).								
<b>Perc. (MaxUsed)</b>	Percentage of current session usage; that is, the value of the <b>MaxUsed</b> column in relation to the value of the <b>Size</b> column.								
<b>MaxSize</b>	Maximum size (in bytes) that has been allocated to the buffer in the course of the current session so far (applies to variable buffers only).								
<b>Perc. (MaxSize)</b>	Maximum size allocated so far (value of the <b>MaxSize</b> column) in relation to the current size (value of the <b>Size</b> column; applies to variable buffers only). A percentage of 10000 or more is indicated by 9999.9 displayed intensified.								
<b>At the end of the list, the following information is displayed:</b>									
<b>ThrdSize</b>	Current size (in KB) of the Natural thread.								
<b>Total</b>	Sums of all buffer sizes (in both bytes and KB) and percentages used/allocated. These totals can also be displayed with PF10 (see below).  For <b>MaxSize</b> , the total shows the maximum additional amount of thread size that would have been needed in the course of the session so far.								

## Commands for Buffer Usage Statistics

In addition to the commands described in *Using SYSTP Utility Screens*, the **Buffer Usage Statistics** screen provides the following PF keys and line command:

PF Key	Line Command	Function
PF4	D	Displays the contents of the buffer marked with the cursor/command in dump format (for internal use by Software AG technical support).
-	S	Displays details of the selected buffer in a tabular overview, for example <b>Thread Offset</b> , <b>Address</b> or <b>Buffer Size</b> .
PF10	-	Displays the <b>Total</b> buffer usage figures.
PF11	-	Displays the relative addresses of the buffers, that is, relative to the input/output control buffer (IOCB).
PF12	-	Displays the buffer compression optimization degree for all buffers in a separate column <b>CmprOpt</b> .  <b>Note:</b> By pressing PF12 twice the table is reverted to the original state.

## Natural Sub-Systems and Roll Server Information

This function displays a list of the Natural subsystems available in your environment and the current status (active or inactive) of authorized service managers and roll servers associated with the subsystems.

In addition to the commands described in *Using SYSTP Utility Screens*, the **Natural Sub-Systems** screen provides the following line commands:

Line Command	Function	Operating System
B	Displays information about the buffer pool and buffer pool cache (if available) such as name, type and size.	z/OS and z/VSE
D	Displays server directory entries from the Session Information Pool (SIP) in dump format (for internal use by Software AG technical support).  For information about SIP, refer to <i>Authorized Services Manager under z/OS</i> in the <i>Operations</i> documentation.	z/OS
L	Displays and resets entries in the roll server file directory.	z/OS
R	Displays roll server statistics, such as the number of roll server slots and roll server files, roll-out and roll-in activities, as well as roll file I/O.  This information can help tune the roll server as described in <i>Roll Server Performance Tuning</i> in the <i>Operations</i> documentation. For example, you can use	z/OS

Line Command	Function	Operating System
	the statistics data to determine an optimum thread size or roll file size for a Natural application.	
S	Displays a list of Zaps applied to the Authorized Services Manager.	z/OS
U	Displays server directory entries from the Session Information Pool (SIP).  For information about SIP, refer to <i>Authorized Services Manager under z/OS</i> in the <i>Operations</i> documentation.	z/OS
Z	Displays a list of Zaps applied to the roll server.	z/OS

## Natural Thread Usage Statistics

This function is only available under CICS, Com-plete, IMS TM and UTM. It is not available in a z/OS Parallel Sysplex environment.

This function allows you to determine an optimum thread size or roll file size for a Natural application.

You should activate this function only when needed, and deactivate it after you have determined your optimum thread size, because this function occupies space in the Natural buffer pool. When you deactivate it, the space in the buffer pool becomes available again.

### ► To execute the Natural Thread Usage Statistics function

- 1 Define an oversized thread in the range of 512 to 1024 KB for your Natural application. Take into account the number of Natural add-on products used.
- 2 Start your Natural application, either in production or in test mode.
- 3 Invoke the SYSTP utility and choose the **Natural Thread Usage Statistics** function by entering T in the **Code** field of the SYSTP **Main Menu**.

The **Thread Usage Statistics** screen appears.

The columns contained on this screen are described in the following table.

- 4 Enter A in the **Code** field to activate the function.

A message appears indicating that the **Thread Usage Statistics** function has been activated.

- 5 Use your Natural application under typical production conditions.

The **Thread Usage Statistics** function runs in the background and logs the buffer sizes used.

- 6 When you want to view the statistics, again, invoke the **Thread Usage Statistics** screen and enter one of the following in the **Code** field:

S

to show the statistics, or

P

to print the statistics, or

G

to show GETMAIN statistics (see [Show Physical GETMAIN Statistics](#)), or

D

to print the statistics and deactivate the function.

We recommend that use function code D after function completion to free buffer pool space.

The following information is displayed on the **Thread Usage Statistics** screen:

Column	Explanation
<b>No.</b>	Buffers are numbered sequentially in order of allocation.
<b>Ext. Buffer</b>	Sizes of these buffers are defined externally (in the NATPARM parameter module).
<b>Defined Size</b>	Buffer size as defined in the NATPARM parameter module.
<b>Max. Allocated Size</b>	Maximum buffer size allocated.  Note that for the internal BB area, 14368 bytes are added to the value of the ESIZE profile parameter (see the <i>Parameter Reference</i> documentation).
<b>Max. Used Size</b>	Maximum buffer size used.
<b>Sum of external buffer sizes</b>	Total of all buffer sizes defined in the NATPARM parameter module.
<b>Sum of internal buffer sizes</b>	Total of all buffer sizes requested by Natural internally.
<b>Sum of physical GETMAINS</b>	Total of all physical GETMAINS for the Natural work pools and the variable Natural buffers outside the Natural user threads.
<b>Max. used thread length</b>	Maximum thread length used by Natural. Define this length as your minimum (optimum) Natural thread length.  Round it up to the next KB number that can be divided by 2.

Column	Explanation
Max. compressed thread length	Maximum length of a compressed Natural thread that was written to the Natural roll file.  Define this length as your minimum (optimum) Natural roll file length.

### Show Physical GETMAIN Statistics

The physical GETMAIN statistics provide information about all physical GETMAINS relevant for the Natural work pools and the variable Natural buffers outside the Natural user threads. They indicate the original buffer sizes (**Org. Size**) during the startup of a Natural session, the number of physical GETMAINS, (**No.**) the buffer length for the physical GETMAIN (**Bytes**) and the buffer position (**P**), above or below the 16-MB line.

The statistics data always refers to the buffers with the greatest lengths requested within a terminal I/O, for all users of the Natural application. The statistics provides a maximum of six entries for each buffer. These entries can be overwritten through the wrap-around procedure. The highest number equals the maximum number of the physical GETMAINS within a terminal I/O, for each buffer concerned.

The first two entries in the statistics refer to the Natural work pools (if available) above (**WRK-POOLA**), respectively, below (**WRKPOOLB**) the 16-MB line. Here, the highest physical GETMAIN number refers to the amount of work pools simultaneously available during the terminal I/O. The sum of all work pool lengths amounts to the total storage requirement of the work pools within a terminal I/O.

All subsequent statistics entries refer to the physical GETMAINS for the variable Natural buffers, which either could not be defined in the Natural user thread due to insufficient space, or were increased outside the Natural user threads. For these buffers, the highest physical GETMAIN number indicates the greatest space requirement for each buffer within a terminal I/O. The total storage space requested earlier was freed before each of the following physical GETMAINS. That is, the sum of all physical GETMAINS with the highest number shows the maximum storage requirement for the variable buffers outside the Natural user threads during a terminal I/O, for all users of the Natural application.

## Natural License Information

This function invokes a screen that displays the contents of the license file in XML format.



# 117

## SYSTP Functions under CICS

---

▪ Natural User Sessions .....	784
▪ Natural Roll Facilities .....	789
▪ Natural Thread Groups .....	790
▪ Natural Storage Threads .....	791
▪ NCI Global System Information .....	792
▪ NCI Generation Options .....	794
▪ Natural Thread Group Definitions .....	794
▪ Own Natural User Session .....	795
▪ CICS Task Information .....	795
▪ System Administration Facilities .....	796

The SYSTP utility provides functions that are specific to CICS.

► **To invoke specific SYSTP functions under CICS**

- In the **Code** field of the SYSTP **Main Menu**, enter E for **Environment-Dependent Functions**.

From the menu displayed then, you can select the functions explained in this section.



**Note:** In the remainder of this section, the Natural CICS Interface is also referred to as NCI.

## Natural User Sessions

This function is used to display a list of active user sessions in a Natural environment.

When you invoke this function, the **Natural User Sessions** screen appears, which displays the following information:

Column	Explanation
<b>Term ID</b>	Unique terminal ID within CICS associated with the Natural session.
<b>User ID</b>	Natural user ID of the Natural session.
<b>Tran</b>	CICS transaction ID under which Natural session is currently running. For pseudo-conversational sessions, this is the pseudo-conversational restart transaction ID.
<b>Start Date / Time</b>	Starting date and time of the Natural session.
<b>Last Act</b>	Time of last screen output.
<b>Stat</b>	Session status: see <i>Operational Status</i> .
<b>Program</b>	Natural program currently active.
<b>Library</b>	Natural library in which the user is currently working.

If you press PF10, the display of the session date and time is replaced by the following session resource data:

Column	Explanation
<b>Thrd Grp</b>	Thread group to which user is assigned.
<b>Thread</b>	Name of thread last used.
<b>Roll Fac</b>	Assigned roll facility.

This section covers the following topics:

- [Commands for Natural User Sessions](#)

- [Natural User Session Statistics](#)

## Commands for Natural User Sessions

In addition to the commands described in *Using SYSTP Utility Screens*, for each item displayed on the **Natural User Sessions** screen, you can execute one of the following line commands.

Line Command	Function
C	<p>Cancel session.</p> <p>Invokes a confirmation window where you can enter YES to mark a session for termination. The session selected is then flagged with number (#) signs that appear in the column <b>User ID</b>. Additionally, for the session concerned, the operational status Purged by Admin appears on the <a href="#">Natural User Session Statistics</a> screen of the user (administrator) who executed the cancel command.</p> <p>The session actually terminates when the owner of the session marked for termination performs the next terminal I/O, Adabas call or external program call. The session owner then receives a corresponding termination notification.</p>
F	<p>Flush session.</p> <p>Invokes a confirmation window where you can enter YES to terminate a session immediately. The session terminated is then flagged with number (#) signs that appear in the column <b>User ID</b>. The user (administrator) who terminated the session receives a termination message when trying to invoke the <a href="#">Natural User Session Statistics</a> screen for the session terminated but still listed on the <b>Natural User Sessions</b> screen. The session owner receives a corresponding termination notification.</p>
R	<p>Reactivate session.</p> <p>Reverses a C (Cancel) command as described earlier. The R command removes the termination flags set for a session and resets the session status to active. Note that you cannot reactivate a session that has been terminated with the F (Flush) command described above.</p>
W	<p>Wake up session.</p> <p>Reactivates immediately a session that has been suspended by a CMROLL call with a non-zero wait interval specified with the MAXROLL profile parameter (see the <i>Parameter Reference</i> documentation). See also the example program SUSPEND supplied in the Natural system library SYSEXTP.</p>

### ▶ To reduce the number of list items by specifying selection criteria

- On the **Natural User Sessions** screen, press PF4.

The **Selection for User Sessions** window appears where you can select user sessions by inactivity date and time, inactivity time interval, Natural server ID, CICS system ID, terminal ID, user ID or transaction ID.

Inactivity date and time and inactivity interval list all sessions that were not active before the date and time specified or before the time interval specified.

A date must be specified in the format *YYYY-MM-DD* (*YYYY* = year, *MM* = month, *DD* = day).  
The time must be specified in the format *HH:II:SS* (*HH* = hours, *II* = minutes, *SS* = seconds).

To specify a range of IDs, use the asterisk (\*) as a leading or trailing character or use the question mark (?) as a wildcard character. See also [Examples of Name Ranges](#).

## Natural User Session Statistics

For each session displayed on the **Natural User Sessions** screen shown above, additional information can be displayed by invoking the **Natural User Session Statistics** screen with the S or U line command. The following section describes the information provided on this screen.

All sizes on the **Natural User Session Statistics** screen are in KB unless otherwise indicated in the field descriptions below.

Field	Explanation
<b>Started</b>	Day, date and time when the session was started.
<b>Last Actions</b>	Date and time when the user was active last.
<b>User</b>	Natural user ID as assigned by the Natural system variable *USER (see the <i>System Variables</i> documentation).
<b>at Terminal</b>	ID of the terminal associated with the Natural session as assigned by the Natural system variable *INIT-ID (see the <i>System Variables</i> documentation).
<b>Transid</b>	Pseudo-conversational transaction ID under which Natural is running.
<b>Task # in</b>	Task number assigned by CICS followed by the ID of the CICS region.
<b>Cur Strg Used</b>	Current amount of storage used by this session.
<b>Max Strg Used</b>	Maximum amount of storage ever used by this session.
<b>Thread Size</b>	Size of this thread.
<b>Thread Name</b>	Name of the thread used last. For threads allocated by using GETMAIN, the thread name is composed of the prefix NSCP followed by the terminal ID.
<b>Thread Group</b>	Name of the associated thread group (triggered by starting the transaction ID).
<b>of Type</b>	Type of thread used for thread group:
	SHR                      Permanent storage threads.
	GETM                     Storage threads allocated by using GETMAIN.
	NONE                     No threads used; all Natural storage requests are passed to CICS.

Field	Explanation
<b>Natural Library</b>	Natural library ID as assigned by the Natural system variable *LIBRARY - ID (see the <i>System Variables</i> documentation).
<b>Natural Program</b>	Name of the Natural program currently used by the session as assigned by the Natural system variable *PROGRAM (see the <i>System Variables</i> documentation).
<b>Line No.</b>	Line number in the Natural program currently used by the session.
<b>Operational Status</b>	See <i>Operational Status</i> in the following section.
<b>Roll Facility</b>	Name of associated roll facility.
<b>Roll Recs (Last)</b>	Number of records written to roll facility for last roll-out.
<b>Roll Recs (Max)</b>	Maximum number of records ever written during roll-out.
<b>Roll Record Size</b>	Record size of this roll facility.
<b>Slot Size</b>	Number of records required to roll-out a thread completely.
<b>Restart Rec. No.</b>	<p>Number of the record that contains roll-out control information; this record must be rolled in first.</p> <p><b>VSAM Roll Files:</b></p> <p>The following applies to VSAM roll files only.</p> <p>The relationship between restart record number (RecNum), slot number (SN) and slot size (SZ) is:</p> $\text{RecNum} = (\text{SN}-1) * \text{SZ} + 2$ <p>or</p> $\text{SN} = (\text{RecNum}-2) / \text{SZ} + 1$
<b>Slot Number</b>	Number of slot in VSAM roll file belonging to this session (for VSAM only). See also <i>VSAM Roll Files</i> in <b>Restart Rec. No.</b> above.
<b>Compressed Length</b>	Amount of relevant storage currently swapped/rolled out.
<b>Session Resumes</b>	Total number of session resumes.
<b>Swap-Ins</b>	Number of session resumes with swapping in from swap pool.
<b>Thread Switches</b>	Number of session resumes with swapping/rolling into a thread which is different to the one the session had been in before.
<b>Roll-Ins</b>	Number of session resumes with rolling in from roll facility.
<b>Region Switches</b>	Number of CICS region switches.
<b>OpSys Switches</b>	Number of operating system image switches in a z/OS Parallel Sysplex environment.

## Operational Status

This field indicates any of the following operational statuses:

Status	Abbreviation	Description
Active	Act	Currently active.
Inactive	Ina	Inactive, still in thread.
Swapped	Swp	Swapped, in swap pool.
Rolled out	Rld	Rolled out, in roll facility.
Wait (Init)	WtI	Waiting for thread on session initialization.
Wait (Resume)	WtR	Waiting for thread on session resume.
Initializing	Int	Initializing session.
Resuming	Res	Resuming session, in thread, not active yet.
Suspending	Sus	Suspending session.
Terminating	Trm	Terminating session.
Swapping out	Swo	Session swapping out.
Swapping in	Swi	Session swapping in.
Rolling out	Out	Rolling out from thread or swap pool.
Rolling in	In	Rolling in from roll facility.

The following additional information can appear in **Operational Status**:

Status	Description
Conversational	Dialog-oriented session (PSEUDO=OFF) as opposed to pseudo-conversational/transaction-oriented session.  See also the PSEUDO profile parameter described in the <i>Parameter Reference</i> documentation.
Forced Conversational	Last screen I/O of a PSEUDO=ON session was conversational.  See also the PSEUDO profile parameter described in the <i>Parameter Reference</i> documentation.
No-Roll	Session is not allowed to roll.
Compressed	Session is compressed (in swap pool or roll facility).
Thread Switched	Thread currently used is not the same as used before.
Thread Locked	Session kept from switching threads (for example, RELO=OFF); can also force No-Roll/Conversational status.  See also the RELO profile parameter described in the <i>Parameter Reference</i> documentation.
Purged by Admin	Session canceled by administrator (flag set).

Status	Description
Spool Task	Task is a spool/print task.
Asynchronous Task	Task is an asynchronous task, not bound to a terminal.

## Natural Roll Facilities

This function is used to display which swap files are available for rolling out user work areas to make room in the swap pool for active users. These swap files are known as roll facilities.

When you invoke this function, the **Natural Roll Facilities** screen appears for the current CICS region (as indicated by the CICS ID in the screen title). For each roll facility, the following information is displayed:

Column	Explanation
<b>Facility Name</b>	TEMPSTOR is used for auxiliary temporary storage, MAINSTOR for main temporary storage, and remaining file names are VSAM roll files as defined in the CICS file control table (FCT). none denotes that no roll facility is used.
<b>Record Size</b>	Record size of this roll facility.
<b>Slot Size</b>	Number of records required to roll out a thread completely (maximum thread size divided by record size, rounded up).
<b>No. of Slots</b>	Number of sessions which fit into this roll file (number of file records divided by slot size, rounded down); applies to VSAM roll files only.
<b>Facility Users Cur / Max</b>	Current ( <b>Cur</b> ) and maximum ( <b>Max</b> ) number of user sessions assigned to this roll facility.
<b>Roll Counts Out / In</b>	Number of session roll operations from or into this roll facility.
<b>Status</b>	Indicates Full if the facility users equal the number of available slots.

In addition to the commands described in *Using SYSTP Utility Screens*, the **Natural Roll Facilities** screen provides the following PF key and corresponding line command:

PF Key	Line Command	Function
PF4	U	Invokes the <b>Natural User Sessions</b> screen (see the relevant section) for the marked roll facility which displays all Natural user sessions that use this facility.

## Natural Thread Groups

This function is used to display which thread groups are available to Natural.

When you invoke this function, the **Natural Thread Groups** screen appears for the current CICS region (as indicated by the CICS ID in the screen title). For each thread group, the following information is displayed on this screen:

Column	Explanation
<b>Group Name</b>	Thread group name.
<b>Group Users Cur / Max</b>	Current ( <b>Cur</b> ) and maximum ( <b>Max</b> ) number of users assigned to this thread group.
<b>Thread Type</b>	<b>Type of thread</b> used: see <b>Natural User Session Statistics</b> .
<b>TCBs</b>	Maximum number of sessions concurrently active.
<b>Thread Size</b>	Thread group's common thread size.
<b>Strg Used</b>	Maximum storage ever used by a session that uses this thread group.
<b>Queue Sizes Cur / Max / AtMax</b>	Current ( <b>Cur</b> ) and maximum ( <b>Max</b> ) queue size for the thread group's central wait queue and the number of times the maximum was reached ( <b>AtMax</b> ).  Only applies if the parameter <b>THREADS</b> has been defined as greater than zero for this thread group. See also <i>THREADS - Number of Threads or Tasks Per Thread Group</i> in the <i>TP Monitor Interfaces</i> documentation.
<b>VSAM Aux / Main</b>	Roll facilities defined for group; CICS temporary storage (auxiliary or main) always backs up VSAM if VSAM roll files are not available or full.

### Commands for Natural Thread Groups

In addition to the commands described in *Using SYSTP Utility Screens*, the **Natural Thread Groups** screen provides the following PF keys and corresponding line commands:

PF Key	Line Command	Function
PF4	U	Displays all <b>Natural user sessions</b> (see the relevant section) that use the thread group marked with the cursor/command.
PF10	T	Displays <b>Natural storage threads</b> (see below) for the thread group marked with the cursor/command.
PF11	D	Displays <b>Natural thread group definitions</b> (see the relevant section) for the thread group marked with the cursor/command.

## Natural Storage Threads

This function is used to display information about the storage threads in the Natural environment.

When you invoke this function, the **Natural Storage Threads** screen appears for the current CICS region (as indicated by the CICS ID in the screen title). The screen displays the following information:

Column	Explanation						
<b>Thread Name</b>	Name of the thread.						
<b>Grp No.</b>	Number of the group to which this thread belongs.						
<b>Thrd Size</b>	Usable thread size.						
<b>Strg Used</b>	Maximum storage ever used by a session that uses this thread.						
<b>Use Count</b>	Number of times this thread has been selected for processing.						
<b>Roll-Ins Log. / Phys.</b>	Number of roll-in operations: <table border="1" data-bbox="477 892 951 1081"> <tr> <td><b>Log.</b></td> <td>Session resumes.</td> </tr> <tr> <td><b>Phys.</b></td> <td>Roll-in from roll facility.</td> </tr> </table>	<b>Log.</b>	Session resumes.	<b>Phys.</b>	Roll-in from roll facility.		
<b>Log.</b>	Session resumes.						
<b>Phys.</b>	Roll-in from roll facility.						
<b>Queue Sizes Cur / Max / AtMax</b>	Number of users waiting in the queue: <table border="1" data-bbox="477 1186 1474 1444"> <tr> <td><b>Cur</b></td> <td>Current number of users queuing on thread. If this number <math>n</math> is greater than 1, <math>n</math> minus 1 users are waiting.</td> </tr> <tr> <td><b>Max</b></td> <td>Maximum queue count for this thread.</td> </tr> <tr> <td><b>AtMax</b></td> <td>Number of times at maximum.</td> </tr> </table>	<b>Cur</b>	Current number of users queuing on thread. If this number $n$ is greater than 1, $n$ minus 1 users are waiting.	<b>Max</b>	Maximum queue count for this thread.	<b>AtMax</b>	Number of times at maximum.
<b>Cur</b>	Current number of users queuing on thread. If this number $n$ is greater than 1, $n$ minus 1 users are waiting.						
<b>Max</b>	Maximum queue count for this thread.						
<b>AtMax</b>	Number of times at maximum.						
<b>Term ID</b>	Terminal ID belonging to the Natural session whose data are in this thread.						
<b>Task No.</b>	ID of CICS task currently active in this thread. If no ID is displayed, no session is active is this thread.						

## Commands for Natural Storage Threads

In addition to the commands described in *Using SYSTP Utility Screens*, the **Natural Storage Threads** screen provides the following line commands and PF key:

Line Command	PF Key	Function
C	-	See <i>Cancel session</i> in <i>Natural User Sessions</i> .
D	PF11	Displays <b>Natural thread group definitions</b> (see the relevant section).
F	-	See <i>Flush session</i> in <i>Natural User Sessions</i> .
G	PF10	Displays <b>Natural thread group</b> (see the relevant section).
R	-	See <i>Reactivate session</i> in <i>Natural User Sessions</i> .
U	-	Displays statistics about the Natural user sessions currently active in the thread. See also <i>Natural User Session Statistics</i> .

## NCI Global System Information

This function is used to display data on the system directory.

When you invoke this function, the **Global System Information** screen appears for the current CICS region (as indicated by the CICS ID in the screen title). The screen provides the following information:

Field	Explanation
<b>Natural User Sessions</b>	Current ( <b>Cur</b> ) and maximum ( <b>Max</b> ) number of Natural sessions in the system.
<b>Concurrent SCP Active</b>	Current ( <b>Cur</b> ) and maximum ( <b>Max</b> ) number of concurrent system control program (SCP) requests.  SCP requests are: session initialization, session suspension, session resumption and session termination.
<b>SIR Block Extensions</b>	Current ( <b>Cur</b> ) and maximum ( <b>Max</b> ) number of local SIR block extensions.
<b>Slots in 1st SIR Block</b>	Number of user sessions that fit into the primary user control block (first USERS subparameter in NCMDIR macro; see <i>USERS - Session Information Record*</i> ).
<b>Slots in SIR Block Extns</b>	Number of user sessions that fit into a secondary user control block (second USERS subparameter in NCMDIR macro; see <i>USERS - Session Information Record*</i> ).
<b>VSAM Roll File Slots</b>	Number of VSAM roll files to check (ROLLFLS).
<b>Possible Roll Facilities</b>	Number of VSAM roll files plus two for CICS (MAINSTOR and TEMPSTOR).
<b>Thread Groups</b>	Number of thread groups determined by evaluating all NCMTGD macro specifications at system startup. See also <i>NCMTGD Macro Parameters*</i> .
<b>System Recoveries</b>	Number of corrections of statistics counts and/or control block chain.

Field	Explanation
<b>Size of DIR Extension (B)</b>	Number of bytes used at system startup for thread control blocks and VSAM roll file online directories.
<b>Operating System Host ID</b>	Name of the operating system image.
<b>Assembled Last</b>	Date and time when the system directory source module was last assembled.
<b>CICS System ID</b>	ID of the CICS region.
<b>Available Resources:</b> <b>Swap Pool</b> <b>Local Buff. Pool</b> <b>Sort Buffer Pool</b> <b>DL/I Buffer Pool</b> <b>Edit Buffer Pool</b> <b>Monitor Pool</b> <b>RNM Buffer Pool</b>	Resources available in the current NCI system environment: swap pools, Natural buffer pools, monitor buffer pools and RNM buffer pools.  Type, size (in KB) and location (below or above the 16 MB line) of all buffer pools supported.
<b>Max Thread Size</b>	Largest thread size across all valid thread groups.
<b>VSAM Roll Files</b>	Number of VSAM roll files.
<b>Main/Aux TempStor</b>	Indicates whether CICS main or auxiliary temporary storage is available for the Natural/CICS roll facilities.
<b>Session Logging</b>	Indicates whether the Natural/CICS log destination is defined in the CICS DCT (destination control table) and whether the log destination is available. The log destination for sessions is defined with the LOGDEST* parameter of the NCMPRM macro.
<b>Message Logging</b>	Indicates whether the Natural/CICS error message log destination is defined in the CICS DCT and whether the log destination is available.  The log destination for messages is defined with the MSGDEST* parameter of the NCMPRM macro.
<b>Message Switching</b>	Indicates whether the message switching transaction ID is defined in CICS and whether the transaction ID is available. The transaction for switching messages is defined with the MSGTRAN* parameter of the NCMPRM macro.  If a transaction ID is not available, a SYSTP session flush (see <i>Flush session</i> in <i>Natural User Sessions</i> ) is not possible.
<b>Console Terminal</b>	Indicates whether the CICS console terminal for Natural/CICS is available. The console terminal ID is defined with the CONSOLE* parameter of the NCMPRM macro.

\* described in the *TP Monitor Interfaces* documentation

## NCI Generation Options

This function is used to display generation parameter settings for Natural running under CICS. The values of these parameters are determined in the macro `NCMPRM`, which is part of the `NCIPARM` parameter module created during installation.

When you invoke this function, the **Generation Options** screen appears for the current CICS region (as indicated by the CICS ID in the screen title). This screen displays an overview of the generation option settings for Natural.

Behind each parameter setting in the **Generation Options** screen is a parameter of the `NCMPRM` macro. These parameter names can be viewed by pressing `PF10`. Press `PF10` to toggle between the screen containing the parameter names and explanations of the parameters.

### Related Topics:

- *Installing the Natural CICS Interface* in the *Installation* documentation
- *NCMPRM Macro Parameters* in the *TP Monitor Interfaces* documentation

## Natural Thread Group Definitions

This function is used to display Natural thread group definitions.

When you invoke this function, the **Natural Thread Group Definitions** screen appears for the current CICS region (as indicated by the CICS ID in the screen title). This screen displays the following information:

Column	Explanation	
Grp No.	Thread group number.	
Group Type	Type of group definition:	
	SHR	Permanent storage threads to be used for thread group.
	GETM	Storage threads allocated by using <code>GETMAIN</code> .
	none	No threads to be used; all Natural storage requests are passed to CICS.

Column	Explanation
	Alias   Thread group redefinition to assign other primary roll facility triggered by transaction ID/task request key.
Roll Fac.	Primary roll facility assigned:  VSAM, Aux (auxiliary temporary storage), Main (main temporary storage) or none (no roll facility assigned).
Thread Size	Thread storage GETMAIN size (for thread group types GETM and SHR).
TCBs	Maximum number of Natural sessions concurrently active in this thread group.
Transaction IDs / Task Request Keys	As defined in the CICS transaction definitions for Natural.

## Commands for Natural Thread Group Definitions

In addition to the commands described in *Using SYSTP Utility Screens*, the **Natural Thread Group Definitions** screen provides the following PF keys and corresponding line commands:

PF Key	Line Command	Function
PF4	S	Displays thread group definitions for the thread group marked with the cursor/command.
PF10	G	Displays <b>Natural storage threads</b> (see the relevant section) associated with the thread group marked with the cursor/command.

## Own Natural User Session

This function invokes the **Natural User Session Statistics** screen described in *Natural User Session Statistics*.

## CICS Task Information

This function invokes the **SYSTP Task Information** window, which displays status information about the Natural task in a CICS environment.

## System Administration Facilities

---

This function is used to access facilities for debugging and tracing.

When you invoke this function, a menu appears with the following functions:

- Trace Facilities
- Debugging Facilities
- System Snapshot for Logging
- Reset System Highwater Marks
- Common Dynamic Params Control Information
- Applied NCI Source Changes
- Applied NCI Zaps

### Trace Facilities

This function reserved for internal use by Software AG personnel only.

### Debugging Facilities

This function reserved for internal use by Software AG personnel only.

### System Snapshot for Logging

This function provides complete SYSTP batch reports (see also *[SYSTP in Batch for CICS Sessions](#)*) with information about all SCP facilities, regardless of whether they have been used or not. Such facilities are:

- Thread groups
- TYPE=SHR threads
- Roll facilities

All this information is logged to the Natural/CICS log file, if available.

## Reset System Highwater Marks

This function comprises the system snapshot function previously described. In addition, all system highwater marks can be reset, for example:

- The number of user sessions.
- Every thread group and roll facility.
- The number of UCB block extensions.
- The amount of storage.
- All thread groups and TYPE=SHR threads.
- All wait queue values and counts.
- All roll facility roll counts.

## Common Dynamic Parns Control Information

This function displays common dynamic profile parameters as retrieved from the PRMDEST destination, if available. See also *PRMDEST - Name of the Natural CICS Profile Parameter Input Destination* described in the *TP Monitor Interfaces* documentation.

## Applied NCI Source Changes

This function invokes the **Applied NCI Source Changes** screen for the current CICS region (as indicated by the CICS ID in the screen title). This screen displays the numbers of all source changes that have been applied to the current Natural TP environment.

## Applied NCI Zaps

This function invokes the **Applied NCI Zaps** screen for the current CICS region (as indicated by the CICS ID in the screen title). This screen displays the numbers of all Zaps that have been applied to the current Natural TP environment.



# 118

## SYSTP Functions under IMS TM

---

▪ Broadcasting .....	800
▪ Display Environment Data .....	800
▪ Monitoring .....	801
▪ Multi Session .....	801
▪ Applied NII Zaps .....	801

The SYSTP utility provides functions that are specific to IMS TM.

▶ **To invoke specific SYSTP functions under IMS TM**

- In the **Code** field of the SYSTP **Main Menu**, enter E for **Environment-Dependent Functions**.

From the **NII Menu** displayed then, you can select the functions explained in this section.



**Note:** In the remainder of this section, the Natural IMS TM Interface is also referred to as NII.

## Broadcasting

---

This function is used to broadcast messages to specific user groups in an IMS environment.

When you invoke this function, the **Broadcasting Menu** appears from which you can select the following functions:

- **Create Broadcast Messages**
- **List all Broadcast Messages**

For further information about the broadcasting function, see the section *Natural under IMS TM - Special Functions* in the *TP Monitor Interfaces* documentation.

## Display Environment Data

---

This function is used to display environmental data on the Natural IMS TM Interface.

When you invoke this function, the **Environment Table** screen appears for the environment table used by the current Natural session. The screen displays the current parameter settings of the Natural IMS TM Interface.

The parameters cannot be updated. For more information about IMS parameters, see the section *Natural under IMS TM* in the *TP Monitor Interfaces* documentation.

## Monitoring

---

This function is used to display monitoring data about Natural user sessions that run under the same Natural subsystem.

When you invoke this function, the **Monitoring** screen appears where you can select the following functions to display monitoring data about user sessions:

- **Active Sessions**  
Displays all active Natural sessions that run under IMS TM.
- **Suspended Sessions**  
Displays all Natural sessions that are currently suspended under IMS TM.
- **User Selection**  
Invokes a window where you can specify selection criteria to display particular Natural user sessions only.

## Multi Session

---

This function is used to display or create multiple Natural sessions.

When you invoke this function, the **IMS Multi Session** screen appears where all active sessions are listed. Additionally, the settings of the create and resume keys are displayed. See also *Functionality of the Multi-Session Feature* in the *TP Monitor Interfaces* documentation.

## Applied NII Zaps

---

This function invokes the **Applied NII Zaps** screen, which displays the numbers of all Zaps that have been applied to the current Natural TP environment.



# 119

## SYSTP Functions under TIAM and UTM

---

■ P-Key Utility .....	804
■ Show Common Memory Pools .....	808

The SYSTP utility provides functions that are specific to TIAM and UTM.

▶ **To invoke specific SYSTP functions under TIAM and UTM**

- In the **Code** field of the SYSTP **Main Menu**, enter E for **Environment-Dependent Functions**.

From the menu displayed then, you can select the functions explained in this section.

## P-Key Utility

---

This function supports the loading of programmable P keys on Siemens 975X terminals under the TP monitors TIAM and UTM.

You can load either the standard Natural key settings (function-key mode KN, KO or KS) to the keys P1 to P20, or user-defined values to individual keys. See also *Function Key Support with 9750 Devices* in the *Operations* documentation.

This function invokes the **P-Key Utility** menu, which provides the following menu:

```

15:54:05          ***** NATURAL SYSTP UTILITY *****          1998-03-25
User VR000001          - P-Key Utility -                          TID 0709

          Code   Function           Parameter
          KU     Load User Values   A,H
          KS     Set KS Mode        L,N
          KN     Set KN Mode        L,N
          KO     Set KO Mode        L,N
          KF     Load F1 - F20
          ?     Help
          .     Exit

          Code .. __          Parameter A

Select function.
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Help Menu Exit KU   KS   KSN  KN   KNN  KO   KON  KF   Canc
    
```



```

Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
      Menu  Exit  Load      -      +                               Canc

```

The **Mode** field is set to HEX or ALPHA depending on the parameter you specify when invoking the function. You can switch modes by replacing the current value by A (for ALPHA) or H (for HEX).

- In ALPHA mode, you can use the left half of the screen to enter an alphanumeric value next to the key you wish it to be loaded to.
- In HEX mode, you can also assign a value to a key in hexadecimal form on the right half of the screen.

For each P key, enter an alphanumeric value in the empty input field or a hexadecimal value in the line below it (for **Parameter** value A, the hexadecimal field is input blocked).

If no value is specified for a key, the standard Natural key setting (function-key mode KN, KO or KS) applies for this key; thus, it is possible to have a mixed P-key usage; that is, some keys with user-defined functions, others with the standard Natural functions.

Load the values by pressing PF4 or by entering L in the **Function** field.

Page the screen to additional P keys by pressing PF8 or by entering a plus (+) sign in the **Function** field.



**Note:** Natural automatically converts all binary values which are smaller than H'40' to H'6F' (= question mark). So, before any binary values smaller than H'40' can be loaded, the macro NTTAB (translation table) has to be changed accordingly so as to avoid this automatic conversion. This is particularly important for H'27' (= ESCAPE) and H'19' (= Endemarke). For detailed information, see *TAB - Standard Output Character Translation* in the *Parameter Reference* documentation.

### Load User Values with LPFSUP01 Interface

The **Load User Values** function is also available to user applications as an application programming interface (API). The API consists of the Natural subprogram LPFSUP01, which performs the loading of the keys. LPFSUP01 is supplied in the system library SYSEXTP and can be copied into user libraries or steplib.

▶ **To call LPFSUP01**

- Issue the following statement:

```
CALLNAT 'LPFSUP01' P-VALUE(*)
```

where P-VALUE must be defined as an array: (A24/20).

**Example:**

```
DEFINE DATA LOCAL
  1 P-VALUE (A24/20)
END-DEFINE
* LOAD '/STA L EM DUE1' TO P1, '/STA P EM DUE1' TO P4
COMPRESS '/STA L' h'192786' INTO P-VALUE(1)
COMPRESS '/STA P' h'192786' INTO P-VALUE(4)
CALLNAT 'LPFSUP01' P-VALUE(*)
END
```

See also the example program LPFEXAM1 in the system library SYSEXTP.

### Set Key-Assignment Mode

The following functions are used to set key-assignment modes on Siemens terminals:

Mode	Function
<b>Set KS Mode</b>	Executes the terminal command %KS* and is invoked by either pressing PF5 or entering function code S in the <b>P-Key Utility</b> menu.
<b>Set KN Mode</b>	Executes the terminal command %KN* and is invoked by either pressing PF7 or entering function code N in the <b>P-Key Utility</b> menu.
<b>Set KO Mode</b>	Executes the terminal command %KO* and is invoked by either pressing PF9 or entering function code O in the <b>P-Key Utility</b> menu.

\* described in the *Terminal Commands* documentation

For detailed explanations of key-assignment modes, see *Natural under BS2000/OSD* in the *Operations* documentation.

## Load Send-Key Codes to P Keys

The **Load F1 - F20** function is used to load specific send-key (F) codes F1 to F20 to the keys P1 to P20. The function is similar to the key assignment mode **KN**, except that F codes can be selected individually.

When this function is invoked, the following screen appears:

```

15:56:34          ***** NATURAL SYSTP UTILITY *****          1998-06-25
User VR000001          - Load F-Codes -          TID VR000001

P01  _           P02  _           P03  _           P04  _           P05  _
P06  _           P07  _           P08  _           P09  _           P10  _
P11  _           P12  _           P13  _           P14  _           P15  _
P16  _           P17  _           P18  _           P19  _           P20  _

Mark P-Key to be loaded with F-Code
Command ==>
Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Load      Menu  Exit                                     Canc

```

To load P keys with F codes, mark the appropriate keys and press **ENTER**. Only the keys which are marked are loaded with F codes. Other P keys retain their original values.

## Show Common Memory Pools

This function displays a list of all common memory pools used in Natural.

The individual items of information shown for each common memory pools are explained in the online help about this function.

# 120

## SYSTP in Batch for CICS Sessions

---

- Invoking SYSTP in Batch Mode ..... 810
- Evaluating the Log File ..... 810

The SYSTP utility can also be used to obtain statistical data on Natural/CICS sessions in batch mode.

The Natural log file into which the statistical data about Natural/CICS sessions is written must be assigned to the Natural batch job as Work File 1 (that is, via CMWKF01). It must also be defined to the online system, which means in the CICS DCT (destination control table); see the LOGDEST parameter of the NCMPRM macro described in the *TP Monitor Interfaces* documentation.

## Invoking SYSTP in Batch Mode

---

### ▶ To invoke the SYSTP utility in batch mode

- In the batch job, specify either of the following commands:

```
SYSTP xxx
```

or

```
LOGON SYSTP  
SYSBATCH xxx
```

where *xxx* indicates what kind of data is to be processed; for example: *xxx=nci* indicates that the data is collected by a Natural/CICS online system.

## Evaluating the Log File

---

Data is written into the Natural log file when Natural is initialized or reset, and when a Natural session is terminated.

The Natural CICS Interface writes the following records into the Natural log file:

- A start log record whenever the Natural environment is initialized or reset.
- A session log record whenever a Natural session is terminated.

When a Natural environment is initialized, a system ID is written into the system control block. This system ID also belongs to all log records. Therefore, a Natural log file can be shared by several Natural/CICS online environments.

The information logged serves to keep track of the usage of the Natural/CICS online environment. Therefore, most of the information refers to facilities of the Natural environment. The log file is not intended to be an accounting or monitoring tool that refers to facilities of CICS.

Based on the system ID, several reports are created with data related to a Natural session:

- Log file data listed in chronological order, which means that session log records are sorted by session end date and time.
- Statistics about how the Natural environment was set up and used.
- Statistics about thread groups (if used).
- Statistics about program storage threads (if used).
- Statistics about roll facilities (if used).

This set of reports is created for all Natural environments with records about Natural/CICS sessions in the Natural log file.



**Note:** The session termination log records, of course, reflect only resources which have been used by the corresponding sessions. Therefore, these records may not reflect the full Natural environment. Reports of a full Natural environment can be obtained by making a snapshot of the whole Natural environment using the [System Administration Facilities](#) function (see the relevant section).



# Index

---

## U

utility  
    overview, 1

