Natural

Parameter Reference

Version 9.1.1

April 2019

ADABAS & NATURAL

This document applies to Natural Version 9.1.1 and all subsequent releases.

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

Document ID: NATMF-NNATPARMS-911-20211014

Table of Contents

Preface

This documentation contains detailed descriptions of all Natural profile and session parameters provided to configure your Natural environment.

If a Natural session parameter with the same name and functionality as a Natural profile parameter exists, the descriptions of both parameters are combined in a single document.

Introduction to Profile Parameters	References to documents providing detailed information on profile parameter usage.
Introduction to Session Parameters	General information on session parameter usage and evaluation.
Profile Parameters Grouped by Category	Summary of profile parameters grouped by category.
Parameters in Alphabetical Order	Descriptions of all profile parameters and session parameters in alphabetical order.

1 About this Documentation

Document Conventions

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

Online Information and Support

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at **https://documentation.softwareag.com**.

Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at **https://empower.softwareag.com/**.

You can find product information on the Software AG Empower Product Support website at **https://empower.softwareag.com**.

To submit feature/enhancement requests, get information about product availability, and download products, go to **Products**.

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the **Knowledge Center**.

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at **https://empower.softwareag.com/public_directory.aspx** and give us a call.

Software AG Tech Community

You can find documentation and other technical information on the Software AG Tech Community website at **https://techcommunity.softwareag.com**. You can:

- Access product documentation, if you have Tech Community credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

Data Protection

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

2 Introduction to Profile Parameters

You specify a Natural profile parameter either dynamically at session start, or statically in the `NTPRM` parameter macro and additional parameter macros contained in the Natural parameter module.

A profile parameter for which no macro name or macro syntax is indicated in the individual parameter description is defined in the `NTPRM` macro.

NTPRM Macro Syntax

The `NTPRM` macro is specified as follows:

```
         NTPRM profile-parameter=value, ...
```

where `value` is any parameter setting valid for the specified `profile-parameter`.

In general, you can use the default parameter values. If any of the default values do not suit your requirements, overwrite them with the required values.

For detailed information on using a profile parameter, see the following sections in the *Natural Operations* documentation:

- Profile Parameter Usage
- Natural Parameter Hierarchy
- Assignment of Parameter Values

3

Introduction to Session Parameters

Session Parameter Usage

In Natural, session parameters are used:

- to specify certain characters,
- to set processing time limits,
- to set a particular response for a given condition,
- to set various size limits,
- to determine various aspects of output reports.

At the installation of Natural, the Natural administrator sets these parameters to default values which are then valid for all users of Natural.

To see which parameter values apply to your session, you enter the system command `GLOBALS` (described in the *System Commands* documentation).

How to Set Session Parameters

Natural session parameters can be set in several ways:

- via the default Natural parameter module, which is set when Natural is installed;
- via dynamic parameters specified when invoking Natural (as described in your Natural *Operations* documentation);
- via the system command `GLOBALS`;
- via a `SET GLOBALS` statement (in reporting mode only);
- via a `FORMAT` statement;
- via parameter specification within statements where parameters also are evaluated, for example, `INPUT`, `DISPLAY`, `WRITE`;
- via terminal commands.

Instead of the parameter values `ON` and `OFF`, you can also specify `T` (true) or `F` (false) respectively.

Changing Session Parameters at Session Level Using the GLOBALS Command

For your Natural session you can change some of the parameter values set by the Natural administrator.

Within your Natural session, you can change these parameters by issuing the following system command:

GLOBALS

When you issue the `GLOBALS` command, a screen is displayed showing the parameter values that are currently in effect for your session. On this screen, you can change the values that do not suit your requirements.

A parameter value set with a `GLOBALS` command remains in effect until the end of the Natural session (and applies to every object you store during the session), unless you change it again with another `GLOBALS` command.

Changing Session Parameters at Program Level Using the FORMAT Statement

You can change certain parameters for the duration of a single program (report). This is done by using a `FORMAT` statement in the program, which will override the session-wide settings for these parameters.

Example of a FORMAT Statement:

```
FORMAT AL=10 HC=R
```

Parameters set with a `FORMAT` statement apply until the end of the executed program, unless they are changed with another `FORMAT` statement in the program.

Not all session parameters can be changed at program level, while several parameters that can be specified at program level cannot be specified at session level; most of the latter are parameters which affect the format of an output report.

Changing Session Parameters at Statement Level

Most of the parameters you can change with a `FORMAT` statement you can also change for an individual statement; for example, for a particular `DISPLAY`, `WRITE`, `INPUT` or `REINPUT` statement.

This is done by specifying the parameter (in parentheses) after the statement name.

Example:

```
DISPLAY (SF=4) NAME JOB-TITLE CURR-CODE SALARY
```

A parameter set at statement level applies only to the statement in which it is specified. The setting at statement level overrides, for that statement only, all other settings of that parameter at other levels.

Changing Session Parameters at Field Level

Within a `DISPLAY`, `WRITE`, `INPUT` or `REINPUT` statement, you can also change some parameters for an individual field or output element.

This is done by specifying the parameter (in parentheses) after the field name.

Example:

```
DISPLAY NAME (AL=10) JOB-TITLE CURR-CODE SALARY
```

The parameter value then applies only to that field. The setting at field level overrides, for that field only, all other settings of that parameter at other levels. However, only some of the parameters that can be set at statement level can also be set at field level.

Session Parameter Evaluation

Parameters specified with the statements `DISPLAY`, `FORMAT`, `PRINT`, `INPUT`, `REINPUT`, `WRITE`, `WRITE TITLE` and `WRITE TRAILER` are processed during program compilation and are therefore included in the corresponding object module for the program.

The following hierarchy is used for evaluation:

1. Parameters set at element/field (highest priority)
2. Parameters set at statement level
3. Parameters set with a `FORMAT` statement
4. The default parameter settings (lowest priority)

Parameters set with a `SET GLOBALS` statement cause the execution time environment to be modified. These modifications remain in effect until overridden by another `SET GLOBALS` statement (or `GLOBALS` system command).

4 Profile Parameters Grouped by Category

Batch Mode

The following profile parameters apply if Natural is used in batch mode:

Parameter	Brief Description
`CC`	Error processing in batch mode
`CPOBJIN`	Code page of batch input file
`CPPRINT`	Code page of batch output file
`CPSYNIN`	Code page of batch input file for commands
`ECHO`	Control printing of input data
`OBJIN`	Use of `CMOBJIN` as Natural input file
`OSP`	Parameters for z/OS batch
`READER`	System logical units for input (z/VSE only)
`VSEP`	Parameter macro for z/VSE batch

Character Assignments

The following profile parameters can be used to change default character assignments:

Parameter	Brief Description
`CVMIN`	Control variable modified at input
`FC`	Filler character for `INPUT` statement
`FCDP`	Filler character for dynamically protected input fields
`CF`	Character for terminal commands
`DC`	Decimal character
`HI`	Help character
`IA`	Input assign character
`ID`	Input delimiter character
`SOSI`	Shift-out/shift-in codes for double-byte character set
`THSEPCH`	Thousands separator character

Compiler Options

The following profile parameters can be used to control the Natural compiler:

Parameter	Brief Description
`CMPO`	Compilation options
`FS`	Default format/length setting for user-defined variables
`SM`	Programming in structured mode
`XREF`	Creation of XRef data for Natural

Database Management

The following profile parameters can be used to configure databases and control access to databases:

Parameter	Brief Description
`ADAACBX`	Use of the Extended Adabas Control Block
`ADANAME`	Name of Adabas link routine
`ADAMODE`	Adabas interface mode
`ADAPRM`	Adabas Review support
`ADASBV`	Adabas security by value
`DB`	Database types and options
`DB2`	Parameters for SQL database management system interfaces
`DBCLOSE`	Database close at session end
`DBOPEN`	Database open without `ETID`
`DBROLL`	Database calls before roll-out
`DBUPD`	Database updating
`ENDBT`	Issue `BACKOUT TRANSACTION` at session end
`ET`	Execution of `END/BACKOUT TRANSACTION` statements
`ETDB`	Database for transaction data
`ETEOP`	Issue `END TRANSACTION` at end of program
`ETID`	User identification for Adabas
`ETIO`	Issue `END TRANSACTION` upon terminal I/O
`ETSYNC`	Issue Syncpoint upon `END/BACKOUT TRANSACTION` statements
`LFILE`	Dynamic specification of logical file
`OPRB`	Database open/close processing for Adabas and VSAM databases

Parameter	Brief Description
`RCFIND`	Handling of Adabas Response Code 113 for `FIND` statement
`RCGET`	Handling of Adabas Response Code 113 for `GET` statement
`RI`	Release of Adabas ISNs
`TF`	Translation of database ID/file number
`UDB`	User database ID
`WH`	Adabas wait for record in hold status

Date and Time Settings

The following profile parameters affect the handling of date and time values by Natural as well as the internal date/time used by Natural:

Parameter	Brief Description
`DD`	Day differential
`DFOUT`	Date format for output
`DFSTACK`	Date format for stack
`DFTITLE`	Date format in default page title
`DTFORM`	Date format
`STACKD`	Stack delimiter character
`TD`	Time differential
`YD`	Year differential
`YSLW`	Year sliding window

Debugging and Tracing

The following profile parameters can be used for debugging and/or tracing purposes:

Parameter	Brief Description
`CANCEL`	Session cancellation with dump
`DBGAT`	Debug attach server for remote debugging with NaturalONE
`DBGERR`	Automatic start of debugger at runtime error
`DU`	Dump generation
`DUE`	Dump generation for specific errors only
`ETRACE`	External trace function
`ITRACE`	Internal trace function

Parameter	Brief Description
PECK	Control error processing of the ECHECK and PCHECK compilation options of the COMPOPT system command
RDC	Configure the Natural Data Collector
RELO	Storage thread relocation
TRACE	Define components to be traced
UPSI	Control of the User Program Switch Indicator (UPSI)

External Subprograms

The following profile parameters affect the dynamic loading and deletion of non-Natural programs:

Parameter	Brief Description
CDYNAM	Dynamic loading of non-Natural programs
CSTATIC	Programs statically linked to Natural
DELETE	Deletion of dynamically loaded non-Natural programs
LIBNAM	Name of external program load library (BS2000, z/OS batch and TSO only)
PGP	Properties for external subprograms
RCA	Resolve addresses of static non-Natural programs
RCALIAS	External name definition for non-Natural programs

Limits

The following profile parameters can be used to prevent a single program from consuming an excessive amount of internal resources:

Parameter	Brief Description
LE	Reaction when limit for processing loop exceeded
LT	Limit for processing loops
MADIO	Maximum database management system interface calls between screen I/O operations
MAXCL	Maximum number of program calls
MAXYEAR	Maximum year for date/time values
MT	Maximum CPU time
OVSIZE	Storage thread overflow size
PD	Number of pages captured by NATPAGE

Monitoring

The following profile parameters can be used to monitor your applications and resources:

Parameter	Brief Description
`O4I`	Collect data for Optimize for Infrastructure (only applies if Optimize for Infrastructure is used)
`RDC`	Configure the Natural Data Collector
`RDCEXIT`	Define Natural Data Collector user exits
`RDCSIZE`	Size of buffer for the Natural Data Collector

Natural System Files

Natural system files are used for the storage of data and programs as described in *Natural System Files* in the *Operations* documentation.

The following profile parameters apply to all system files:

Parameter	Brief Description
`DBID`	Default database ID of Natural system files
`FNR`	Default file number of Natural system files
`SYSPSW`	Default password for Natural system files
`SYSCIP`	Default cipher key for Natural system files
`RFILE`	File for recordings
`ROSY`	Read-only access to Natural system files (`FNAT`, `FUSER` and `FSEC` only)

With the following parameters, you can override the default values for individual system files:

Parameter	Brief Description
`FNAT`	Natural system file for system programs
`FUSER`	Natural system file for user programs
`FDIC`	Predict system file
`FSEC`	Natural Security system file
`FSPOOL`	Natural Advanced Facilities spool file
`FPROF`	Natural system file for parameter profiles
`FREG`	Natural registry system file

Natural with Other Software AG Products

The following profile parameters are used in connection with other Software AG products:

Product	Parameter	Brief Description
Adabas Text Retrieval	`TSIZE`	Size of buffer for Adabas Text Retrieval
Con-nect	`CSIZE`	Size of Con-nect/Con-form buffer area
EntireX Broker	`BSIZE`	Size of EntireX Broker buffer
Entire DB Engine	`ZSIZE`	Size of Entire DB Engine buffer
Entire System Server	`ASIZE`	Entire System Server auxiliary buffer
Natural Advanced Facilities	`NAFSIZE`	Size of Natural Advanced Facilities buffer
	`NAFUPF`	User profile for Natural Advanced Facilities
Natural Connection	`PC`	Control of PC access method with Natural Connection
Natural for DB2 Natural SQL Gateway	`DB2`	Parameters for SQL database management system interfaces
	`DB2SIZE`	Natural buffer area for DB2
Natural for DL/I	`DLISIZE`	Size of Natural buffer for DL/I
Natural for VSAM	`VSAM`	Parameters for Natural for VSAM
	`VSIZE`	Size of buffer for Natural for VSAM
NaturalONE	`PDPSIZE`	Size of data buffer for the NaturalONE Profiler
	`DBGAT`	Debug attach server for remote debugging with NaturalONE
Natural Optimizer Compiler	`OPT`	Control of Natural Optimizer Compiler
Natural RPC	`RPC`	Remote procedure call settings for Natural RPC
Optimize for Infrastructure	`O4I`	Collect data for Optimize for Infrastructure
Software AG Editor	`EDBP`	Software AG Editor buffer pool definitions
	`EDPSIZE`	Size of Software AG Editor auxiliary buffer pool
	`SSIZE`	Size of buffer for the Software AG Editor
Natural Web I/O Interface	`WEBIO`	Web I/O interface screen rendering
Natural Batch for zIIP, Natural for CICS for zIIP, Natural for Com-plete for zIIP, Natural for IMS for zIIP	`ZIIP`	zIIP (IBM's System z Integrated Information Processor) processing configuration under z/OS See also the *Natural for zIIP* documentation.

Performance Optimization

The following profile parameters are used to improve the performance:

Parameter	Brief Description
`OPT`	Control of Natural Optimizer Compiler (only applies if the Natural Optimizer Compiler is installed)
`ZIIP`	zIIP (IBM's System z Integrated Information Processor) processing configuration under z/OS (only applies if Natural Batch for zIIP, Natural for CICS for zIIP, Natural for Com-plete for zIIP and/or Natural for IMS for zIIP is installed)

Output Reports and Work Files

The following profile parameters control standard attributes used during the creation of Natural reports:

Parameter	Brief Description
`DL`	Display length for output
`EJ`	Page eject
`FAMSTD`	Overwriting of print and work file access method assignments
`HCAM`	Hardcopy access method
`HCDEST`	Hardcopy output destination
`INTENS`	Printing of intensified fields
`LS`	Line size for Natural records
`MAINPR`	Override default output report number
`MP`	Maximum number of report pages
`PCNTRL`	Print control characters
`PM`	Print mode
`PRINT`	Printer assignments
`PS`	Page size for Natural reports
`SF`	Spacing factor
`TQ`	Translate quotation marks
`TS`	Translate output from programs in system libraries
`WORK`	Work file assignments
`ZP`	Zero printing

Remote Procedure Call (RPC) Settings

The following profile parameter is used to set up and process remote procedure calls between a client and a server using Natural RPC.

Parameter	Brief Description
`RPC`	Remote procedure call settings for Natural RPC.

Session Control

The following profile parameters can be used to control Natural sessions:

Parameter	Brief Description
`CM`	Command mode
`ETA`	Error transaction program
`FREEGDA`	Release global data area in utility mode
`MENU`	Menu mode
`NC`	Use of Natural system commands
`SORT`	Control of sort program
`STEPLIB`	Additional steplib library
`SYNERR`	Control of syntax errors
`UCONMAX`	Specify the maximum number of concurrent sessions for a user
`ULANG`	User language
`ZD`	Zero-division check

Session Initialization and Termination

The following profile parameters have an influence on the initialization or termination of a Natural session:

Parameter	Brief Description
AUTO	Automatic logon
ENDMSG	Display of session-end message
IMSG	Session initialization error messages
ITERM	Session termination upon initialization error
MENU	Menu mode
NUCNAME	Name of environment-independent nucleus
PROGRAM	Program to receive control after session termination
STACK	Place data/commands on Natural stack
STEPLIB	Additional steplib library

Source Management

The following profile parameters can be used to manage Natural sources:

Parameter	Brief Description
RECAT	Dynamic recataloging
RNCONST	Renumbering of line numbers in constants
SL	Source-line length
SLOCK	Source locking
SYNERR	Control of syntax errors

Storage Management

The following profile parameters affect the Natural buffers and buffer pools:

Parameter	Brief Description
BP82	Buffer pool placeholder when object not found
BPCSIZE	Cache size of Natural buffer pool
BPC64	Cache storage type for Natural buffer pool
BPI	Buffer pool initialization
BPLIST	Name of preload list for Natural buffer pool
BPMETH	Buffer pool space search algorithm
BPNAME	Name of Natural global buffer pool
BPPROP	Global buffer pool propagation

Parameter	Brief Description
`BPSFI`	Object search sequence in buffer pool
`BPSIZE`	Size of Natural local buffer pool
`BPTEXT`	Size of text segments in Natural buffer pool
`CMPR`	General default compression optimization algorithm
`DATSIZE`	Size of buffer for local data
`DS`	Size of storage buffer
`DSIZE`	Size of debug buffer
`ESIZE`	Size of user buffer extension
`ISIZE`	Size of initialization buffer
`MAXROLL`	Number of `CMROLL` calls before roll-out
`MONSIZE`	Size of `SYSTP` monitor buffer
`PDPSIZE`	Size of data buffer for the NaturalONE Profiler (on NaturalONE servers only)
`RDCSIZE`	Size of buffer for the Natural Data Collector
`RJESIZE`	Initial size of `NATRJE` buffer
`RUNSIZE`	Size of runtime buffer
`THSIZE`	Thread size
`WPSIZE`	Sizes of Natural work pools

Terminal Communication

The following profile parameters affect the use of Natural on video terminals:

Parameter	Brief Description
`ATTN`	Attention key interrupt support
`CLEAR`	Processing of CLEAR key in `NEXT` mode
`DO`	Display order of output data
`DSC`	Data stream compression (for IBM 3270-type terminals)
`EMFM`	Edit mask free mode
`ESCAPE`	Ignore terminal commands `%%` and `%`
`IKEY`	Processing of PA keys and PF keys
`IM`	Input mode
`KEY`	Value assignments to PA, PF and CLEAR keys
`LC`	Lower to upper case translation
`ML`	Position of message line
`MSGSF`	Display system error messages in full

Parameter	Brief Description
`OPF`	Overwriting of protected fields by helproutines
`REINP`	Issue internal `REINPUT` statement for invalid data
`RM`	Retransmit modified fields
`SA`	Sound terminal alarm
`SOSI`	Shift-out/shift-in codes for double-byte character set
`TMODEL`	IBM 3270 terminal type
`TTYPE`	Terminal type

Translation Tables

The following profile parameters can be used to overwrite character translation tables used by Natural:

Parameter	Brief Description
`CCHAR`	Allow output control characters
`CCTAB`	Printer escape sequence definition
`CP`	Default code page name
`SCTAB`	Scanner character type table
`TAB`	Standard output translation table
`TABA1`	EBCDIC to ASCII translation table
`TABA2`	ASCII to EBCDIC translation table
`TABL`	Translation table for output from `SYS` libraries
`TAB1`	Alternative output translation table
`TAB2`	Alternative input translation table
`UTAB1`	Lower to upper case translation table
`UTAB2`	Upper to lower case translation table

Unicode and Code Page Support

The following profile parameter can be used for Unicode and code page support:

Parameter	Brief Description
`CFICU`	Enables/disables support for Unicode and code pages
`CPCVERR`	Conversion error
`SHAPED`	Enables/disables conversion of unshaped characters

Usage Control and Configuration Settings for Profile Parameters

The following profile parameters affect the use and the configuration of Natural profile parameters:

Parameter	Brief Description
`DYNPARM`	Control use of dynamic parameters
`PARM`	Alternative parameter module
`PLOG`	Logging of dynamic parameters
`PROFILE`	Activate dynamic parameter profile
`SYS`	Activate set of dynamic profile parameters
`USER`	Restrict use of profile parameters

Support for Internet and XML

The following profile parameters support internet and XML access:

Parameter	Brief Description
`WEBIO`	Web I/O interface screen rendering (only applies if the Natural Web I/O Interface is installed)
`XML`	Activate `PARSE XML` and `REQUEST DOCUMENT` statements

TP Monitor Interfaces

The following profile parameters apply if Natural is used with a TP monitor interface:

Parameter	Brief Description
`ASYNNAM`	Output system ID for asynchronous processing (*open*UTM only)
`CICSP`	Environment parameters for the Natural CICS Interface
`COMP`	Parameters for Com-plete
`IMSP`	General parameters for the Natural IMS TM Interface
`IMSPE`	Environment parameters for the Natural IMS TM Interface
`IMSPT`	Transaction definitions for the Natural IMS TM Interface
`OUTDEST`	Output destination for asynchronous processing under CICS, Com-plete and *open*UTM
`PSEUDO`	Pseudo-conversational mode (CICS only)
`SENDER`	Screen output destination for asynchronous processing under CICS, Com-plete, IMS TM and *open*UTM
`SKEY`	Storage protection key
`SUBSID`	Subsystem ID
`TSOP`	Parameters for the Natural TSO Interface

5 ABLOG – Log Program Execution Errors

This profile parameter is used to log all Natural errors that indicate a timeout or an abnormal termination (NAT0953 to NAT0956 errors) during program execution.

The log records include the termination code, the termination address in storage, and the current Natural library, program and statement line number where the error occurred.

Possible settings	`ON`	The Natural errors NAT0953 to NAT0956 are logged and written to the message log file.
	`OFF`	The Natural errors NAT0953 to NAT0956 are not logged.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

6

AD - Attribute Definition

With this session parameter, you specify field attributes at field/element or statement level.

Related session parameter: `CD` - Color Definition

Possible settings	See *AD Parameter Syntax*.	You can specify multiple attributes in any sequence.
Default setting	See below.	
Applicable statements	`FORMAT`	

	`DISPLAY`
	`INPUT`
	`NEWPAGE WITH TITLE`
	`PRINT`
	`REINPUT`
	`WRITE`
	`WRITE TITLE`
	`WRITE TRAILER`
		Parameter may be specified at statement level and/or at element level.

	`ASSIGN`
	`CALLNAT`
	`CALLDBPROC`
	`COMPUTE`
	`MOVE`
	`PERFORM`
	`SEND METHOD`
		Parameter may be specified at element level, however, only the attributes specified in the relevant statement description can be used.

Applicable command	none	

The following topics are covered below:

AD Parameter Syntax

AD=[*field-representation*] [*field-alignment*] [*field-i/o-characteristics*] [*interpretation-of-alphanumeric-fields*] [*mandatory-input*] [*input-value-length*] [*field-upper/lower-case*] [*filler-character*]

You can specify multiple attributes in any sequence. Possible values are:

$$\text{AD=}\left\{\begin{matrix}\text{B}\\\text{C}\\\underline{\text{D}}\\\text{I}\\\text{N}\\\text{U}\\\text{V}\\\text{Y}\end{matrix}\right\}\left\{\begin{matrix}\text{L}\\\text{R}\\\text{Z}\end{matrix}\right\}\left\{\begin{matrix}\text{A}\\\text{M}\\\text{O}\\\text{P}\end{matrix}\right\}\left\{\underline{\text{Q}}\right\}\left\{\begin{matrix}\text{E}\\\underline{\text{F}}\end{matrix}\right\}\left\{\begin{matrix}\text{G}\\\underline{\text{H}}\end{matrix}\right\}\left\{\begin{matrix}\text{T}\\\underline{\text{W}}\end{matrix}\right\}\left\{\text{'}c\text{'}\right\}$$

The meaning of the attributes and the possible values are explained below.

Examples:

```
DISPLAY #FIELDA (AD=R)
INPUT #FIELDB (AD=M)
INPUT (AD=IM) #FIELDA #FIELDB
```

Field Representation

Value	Meaning	Statements	Explanation

`B`	blinking (*)	`ASSIGN`
		`COMPUTE`
		`MOVE`
		`DISPLAY`
		`FORMAT`
		`INPUT`
		`PRINT`
		`REINPUT`
		`WRITE`
			The value of the field is displayed blinking.

`C`	cursive/italic (*)		The value of the field is displayed cursive/italic.
`D`	default intensity		The value of the field is displayed with normal intensity, that is, not highlighted in any way. This is the default value.
`I`	intensified		The value of the field is displayed intensified.
`N`	non-display		A value entered in the field will not be displayed.
`U`	underlined		The value of the field is displayed underlined.
`V`	reverse video (*)		The value of the field is displayed reverse video.

`Y`	dynamic attributes	`INPUT`
		`DISPLAY`
		`PRINT`
		`WRITE`
			Attributes are to be controlled via an attribute control variable (Format C).

* The field representation attributes marked with an asterisk (*) require corresponding hardware features, and will be ignored at runtime if these features are not available.

Field Alignment

Value	Meaning	Statements	Explanation

`L`	left-justified	`DISPLAY`
		`FORMAT`
		`INPUT`
		`PRINT`
		`REINPUT`
		`WRITE`
			The value of the field is displayed left-justified. This is the default value for alphanumeric fields.

`R`	right-justified		The value of the field is displayed right-justified. This is the default value for numeric fields.
`Z`	leading zeros		Numeric values are displayed with leading zeros, right-justified.

Field Input/Output Characteristics

Value	Meaning	Statements	Explanation
`A`	input field, non-protected	`INPUT` `FORMAT`	The value of the field is to be entered in response to the `INPUT` statement. This is the default value.

	input only	`CALLNAT`
		`CALLDBPROC`
		`PERFORM`
		`SEND METHOD`
		Function Call
			If you mark a parameter with `AD=A`, its value will not be passed to the called object (subprogram, stored procedure, subroutine, dialog, method), but it will receive a value from the called object.
			For a field defined with `BY VALUE` in the called object's parameter data area, the calling object cannot receive a value. In this case, `AD=A` only causes the field to be reset to the low value of the respective format (blanks for alphanumeric, binary zeroes for binary and zeroes for numeric fields) before the object is called.
			For `CALLNAT`, `AD=A` may be useful for remote subprograms executed via Natural RPC in a client/server environment to reduce the load of data sent. If a subprogram is executed locally, `AD=A` fields will be reset to the low value of the respective format before the object is called.
			If for `SEND METHOD`, a method is not implemented in Natural, the behavior depends on the method implementation. The parameter is then passed as an initialized variant. Whether the external component is able to return a value is described in the documentation of the external component. It can also be viewed in the Natural Component Browser.

`M`	output field, modifiable	`INPUT` `FORMAT`	The value of the field is to be displayed during `INPUT` statement execution, and a different value may be entered by the user. The field is an output field and may be modified.

	modifiable	`CALLNAT`
		`CALLDBPROC`
		`PERFORM`
		`SEND METHOD`
		Function Call
			By default, the passed value of a parameter can be changed in the called object (subprogram, stored procedure, subroutine, dialog, method) and the changed value passed back to the calling object, where it overwrites the original value.
			For a field defined with `BY VALUE` in the called object's parameter data area, no value is passed back.
			If, for `SEND METHOD`, a method is *not* implemented in Natural, the behavior depends on the method implementation. The parameter is then passed `BY REFERENCE`. Whether the external component accepts a by reference or by value parameter is described in the documentation of the external component. It can also be viewed in the Natural Component Browser.

`O`	output field, write-protected	`INPUT` `FORMAT`	The value of the field is to be displayed during `INPUT` execution. The field is an output field and may not be modified.

Value	Meaning	Statements	Explanation

	non-modifiable	`CALLNAT`
		`CALLDBPROC`
		`PERFORM`
		`SEND METHOD`
		Function Call
			If you mark a parameter with `AD=O`, the passed value can be changed in the called object (subprogram, stored procedure, subroutine, dialog, method), but the changed value cannot be passed back to the calling object; that is, the field in the calling object retains its original value.
			Internally, `AD=O` is processed in the same way as a call-by-value (see `BY VALUE` in the section Parameter Data Definition in the description of the `DEFINE DATA` statement).
			If for `SEND METHOD`, a method is implemented in Natural, the parameter is treated like it was defined `BY VALUE` in the method's parameter data area (see the *PARAMETER clause* of the `INTERFACE` statement).
			If for `SEND METHOD`, a method is *not* implemented in Natural, the behavior depends on the method implementation. The parameter is then passed `BY VALUE`. Whether the external component accepts a call by reference or by value parameter is described in the documentation of the external component. It can also be viewed in the Natural Component Browser.

`P`	temporarily protected	`INPUT` `REINPUT`	Used in conjunction with an attribute control variable (Format C), the `DY` parameter (dynamic attributes), and the `REINPUT` statement.

Note: The Field Input/Output Characteristics `A`, `M` and `O` of the `AD` parameter may be also specified in function calls.

Interpretation of Alphanumeric Fields

Value	Meaning	Statements	Explanation

`Q`	display alphanumeric field as if it were a numeric field	`ASSIGN`
		`COMPUTE`
		`MOVE`
		`DISPLAY`
		`FORMAT`
		`INPUT`
		`PRINT`
		`REINPUT`
		`WRITE`
			This attribute is available on mainframe computers only. A corresponding hardware feature is required.
			An alphanumeric field is interpreted as if it were a numeric field. If the field is displayed under the scope of profile or session parameter `PM=I`, the value of the field is interpreted from left to right instead of right to left.

Mandatory Input

Value	Meaning	Statements	Explanation
`E`	value mandatory	`INPUT` `FORMAT`	A value must be entered in the field in response to an `INPUT` statement; otherwise an error message will be issued. This is only relevant for input-only fields (`AD=A`).
`F`	value optional	`INPUT` `FORMAT`	A value can, but need not, be entered in the field in response to an `INPUT` statement. This is the default value.

Length of Input Value

Value	Meaning	Statements	Explanation
`G`	value size	`INPUT` `FORMAT`	The value entered in the field in response to an `INPUT` statement must be of the same length as the field. This is only relevant for input-only fields (`AD=A`).
`H`	value size	`INPUT` `FORMAT`	The value entered in the field in response to an `INPUT` statement may be shorter than the field. This is the default value.

Field Upper/Lower Case Characteristics

Value	Meaning	Statements	Explanation
`T`	translate lower to upper case	`INPUT` `FORMAT`	The value entered is to be translated to upper case.
`W`	accept lower case	`INPUT` `FORMAT`	Lower case values are to be accepted. `AD=W` is the default value. **Note:** To make `AD=W` effective, you have to specify the value `ON` for the Natural profile parameter `LC`.

Filler Character

Value	Meaning	Statements	Explanation
`'c'`	filler character	`INPUT` `FORMAT`	The empty field is to be filled with the specified character c (for display only) if `AD=A` (input field, non-protected) or `AD=M` (output field, modifiable) is specified.

Before the value is displayed for a modifiable field (`AD=M`), field positions that are not occupied by the value are filled with the specified filler character as follows:

- Leading or trailing positions (depending on the field alignment) are filled for format I, N and P fields.
- Trailing positions are filled for format A fields.

If the user enters a value in response to the `INPUT` statement, before the value has been assigned to the field,

- both leading and trailing filler characters are removed for format I, N and P fields,
- trailing filler characters are removed for format A fields.

Caution: Filler characters that may occur as part of the value in either leading or trailing position should be avoided to prevent undesired results. For example, if the filler character “0” (zero) is defined for a field of format N5 and the value `00100` is entered as input data, leading and trailing zeroes are removed so that only the value `1` remains, and will be assigned to the field. For the same reason, the minus sign “-” should be avoided as a filler character for numeric fields if negative values are to be entered.

If the filler character is set to blank (`X'40'`), filling blanks are replaced by `X'00'` to allow for insertion of characters without having to clear the remainder of the input field before.

In BS2000 environments, `X'00'` characters are displayed as dots on 97*xx* type terminals. Their appearance can be changed by means of the SIDA utility or with the configuration utility of the respective terminal emulation.

7 ADAACBX – Use of the Extended Adabas Control Block

This profile parameter determines whether the Natural nucleus uses ACBX (extended Adabas control block) calls to access Natural system files instead of ACB (classic Adabas control block) calls if running with Adabas Version 8 or above.

Possible settings	`ON`	The extended Adabas control block (ACBX) is used for Natural system file access.
	`OFF`	The classic Adabas control block (ACB) is used for Natural system file access.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

⚠ **Important:** If you have Adabas link routine user exits that expect the ACB layout, you must set `ADAACBX=OFF` to avoid problems.

`ADAACBX` does not affect Adabas calls submitted by Natural programs. Their use of the ACBX is controlled by the Adabas version specified as the database type with the profile parameter `DB`, for example, `DB=(ADAV8)` for Adabas Version 8. The ACBX is used since Adabas Version 8.

We recommend that you set `ADAACBX=ON`. In this case, Natural objects are loaded in chunks of 64 KB instead of 32 KB from the Natural system file thus reducing the number of Adabas calls for large objects. In consequence, you must consider the following:

- The storage requirements in the Natural address space are increased by 64 KB.
- The Adabas `LU` parameter must be set to 64 KB (default) or above.
- The number of Adabas attached buffers may have to be increased.

8 ADAMODE - Adabas Call Interface Mode

This Natural profile parameter controls the Adabas call interface mode and the number of Adabas user sessions used by Natural to issue Adabas calls.

Possible settings	See below.
Default setting	`2`
Dynamic specification	yes
Specification within session	no

The following values are possible for the `ADAMODE` parameter:

Value	Separate Adabas User Sessions for Nucleus and User Application Database Calls [1]	3GL Program Adabas Calls Use Natural's Adabas Session for User Application Calls [2]	Image Switching in a z/OS Parallel Sysplex Environment Supported [3]
`0`	No	Yes	No
`1`	No	No	Yes
`2`	Yes	No	Yes
`3`	Yes	Yes	No

Notes:

1. **Support for ADAMODE Settings**

   If the value set for `ADAMODE` is not supported by the Adabas link routine used in your environment, an error message is reported and the value of `ADAMODE` is set to `0`.

2. **Separate Adabas User Sessions for Nucleus and User Application Database Calls**

   Two Separate Adabas User Sessions

   If Natural uses two separate Adabas user sessions to issue Adabas calls, Natural uses one Adabas user session to handle Adabas calls issued by the Natural nucleus (for example, to load Natural objects from the system file), and the other Adabas user session to issue Adabas calls issued by the user application.

An Adabas timeout (leading to Natural error NAT3009) that occurs for the Adabas user session that is used to handle Adabas calls issued by the Natural nucleus does not affect the user application.

A separate Adabas user queue element (UQE) is generated for each Adabas user session, and it may be necessary to increase the Adabas `ADARUN` parameter `NU`.

Single Adabas User Session

If Natural uses only a single Adabas user session, `END TRANSACTION` and `BACKOUT TRANSACTION` statements issued by either the Natural nucleus or the user application affect transactions started by both the Natural nucleus and the user application.

An Adabas timeout (leading to Natural error NAT3009) that occurs for the Adabas user session is always reported to the user application, because it is not possible to check whether the timeout affects the application's transaction state.

If Natural uses a single Adabas user session to handle Adabas calls issued by the Natural nucleus as well as Adabas calls issued by the user application, only one UQE is necessary.

3. **3GL Program Adabas Calls Use Natural's Adabas Session for User Application Calls**

Calls Using Natural's Adabas Session

If a 3GL program, which is called from the user application, issues Adabas calls, and if these Adabas calls use Natural's Adabas session for user application calls, these Adabas calls participate in the user application's transaction handling (`END TRANSACTION` and `BACKOUT TRANSACTION` statements), and they are affected by Natural transaction processing related profile parameter settings (see the parameters mentioned below).

Calls Not Using Natural's Adabas Session

If a 3GL program, which is called from the user application, issues Adabas calls, and if these Adabas calls do not use Natural's Adabas session for user application calls, these Adabas calls will not participate in Natural's transaction handling for the Adabas user session. As a consequence, such 3GL programs need to perform their own transaction handling.

4. **Image Switching in a z/OS Parallel Sysplex Environment Supported**

If image switching in a z/OS Parallel Sysplex environment is supported, the Natural session may, after a terminal I/O operation, seamlessly continue to execute in a z/OS image that is different to the z/OS image where the Natural session has executed before the terminal I/O operation. Installation of the Natural Roll Server is required to support execution in a z/OS Parallel Sysplex environment.

To ascertain support of image switching in a z/OS Parallel Sysplex environment, even if `ADAMODE=0` or `ADAMODE=3` is set, Adabas System Coordinator (product code COR) must be installed.

Caution: Setting the value of `ADAMODE` so that image switching in a z/OS Parallel Sysplex environment is not supported may lead to unpredictable results if the Natural session continues execution in a another z/OS image after a terminal I/O operation. Depending on Natural transaction processing related profile parameter settings (see the parameters mentioned below), this may include:
- non-zero Adabas response codes (leading to, for example, Natural error NAT3021),
- database updates that have not yet been committed by an `END TRANSACTION` statement are unintentionally backed out or applied to the database.

Other transaction processing related parameters: `DBCLOSE` | `DBOPEN` | `ENDBT` | `ET` | `ETDB` | `ETEOP` | `ETIO` | `ETSYNC`

9 ADANAME - Name of Adabas Link Routine

This Natural profile parameter specifies the name of the Adabas link routine to be used.

Possible settings	1 - 8 characters	Valid module or entry name.
Default setting	`ADABAS`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. `ADANAME` does not apply to *open*UTM and Com-plete.
2. If the Adabas link routine is linked to the Natural parameter module and its entry name is the same as the one specified by `ADANAME` in the parameter module, the linked routine will be used. If not, the specified link routine will be loaded dynamically. Thus, it is no longer necessary to statically link the Adabas link module to the Natural nucleus.
3. It is possible to run the same Natural nucleus with different Adabas link modules.
4. Under CICS, the Adabas link routine must not be linked to Natural.

10 ADAPRM - Adabas Review Support

This Natural profile parameter is used to pass Natural session data to Adabas Review within the seventh Adabas buffer.

Possible settings	`ON`	Natural session data is passed. **Note:** Set `ADAPRM` to `ON` if Adabas Review is installed.
	`OFF`	No Natural session data is passed.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

11 ADASBV - Adabas Security by Setting

This Natural profile parameter can be used to prevent invalid results for accesses to Adabas files that are protected by “security-by-setting”. When a file that is protected by “security-by-setting” is accessed, invalid results may be returned in some cases where no format buffer is generated and passed to Adabas.

Possible settings	`ON`	Natural session data is passed. **Note:** It is recommended that you set `ADASBV=ON` if you access “security-by-setting”-protected Adabas files. A format buffer is then always passed to Adabas for a database access (even if this is a 2-byte dummy buffer), thus avoiding invalid results.
	`OFF`	No Natural session data is passed.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

12 AL - Alphanumeric Length for Output

With this session parameter, you specify the default output length for an alphanumeric field; that is, when it is specified shorter than the field length, the field will be right-truncated.

Possible settings	`1` to *n*	*n* = value of `LS` (line size) parameter minus 1
Default setting	none	
Applicable statements	`FORMAT`	

	`DISPLAY`
	`INPUT`
	`PRINT`
	`WRITE`
		Parameter may be specified at statement level and/or at element level.

Applicable command	none	

Notes:

1. It is not recommended to use the `AL` session parameter for input fields (**attribute definition** `AD=A` or `AD=M`) in an `INPUT` statement.
2. Any edit mask specified for a field (see session parameter `EM`) will override the `AL` session parameter for this field.

Example:

```
FORMAT AL=20
```

See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

13

ASIZE - Entire System Server Auxiliary Buffer

This Natural profile parameter determines the size of the Entire System Server auxiliary buffer.

Possible settings	`64 - 512`	Buffer size in KB. **Note:** 1. If Entire System Server is to be used, this parameter *must* be set; see the *Entire System Server* documentation. 2. The value of `96` is recommended. It should be sufficient for most applications. 3. An `ASIZE` in excess of 96 KB is needed to enable calls transporting larger amounts of data in multiple value fields to the Entire System Server. For example, the view `SEND-EMAIL` may contain 3 times 20 e-mail addresses, each address being defined to consist of 128 characters.
	`0`	If `ASIZE=0` is specified or if the requested space is not available, the Entire System Server is not activated.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. `ASIZE` only applies if Entire System Server is installed.
2. As an alternative to `ASIZE`, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.

14 ASPSIZE - Work Area Size of Adabas Stored Procedures and Triggers

This Natural profile parameter is obsolete and only accepted for compatibility reasons.

15

ASYNNAM - Output System ID for Asynchronous Processing (under openUTM)

For asynchronous processing between two Natural applications that are running under the TP monitor *open*UTM, this parameter specifies the address of the synchronous application which is used by the asynchronous application to send messages to the synchronous application.

Possible settings	1 - 8 characters.	Valid transaction name.
	blank	No asynchronous processing takes place.
Default setting	blank	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Natural under *open*UTM.
2. For further information on asynchronous processing under *open*UTM, see *Asynchronous Transaction Processing* in the Natural *TP Monitor Interfaces* documentation.

16 ATTN - Attention Key Interrupt Support

This Natural profile parameter controls the use of the attention key for IBM SNA terminals.

Possible settings	`ON`	The attention key causes Natural processing to be interrupted.
	`OFF`	The attention key is ignored.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. Pressing the attention key can interrupt Natural processing with an appropriate error message (NAT1016).
2. The availability of an attention key depends on the environment and on the terminal type.
3. This functionality is also available for Natural in batch mode under z/VSE to interrupt a batch session; see *NATVSE Attention Interrupts* in the *Operations* documentation.

17 AUTO - Automatic Logon

This Natural profile parameter causes an automatic logon to a specific library at the start of the Natural session.

Possible settings	`ON`	An automatic logon is executed at the start of the Natural session.
	`OFF`	No automatic logon is performed.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The setting contained in the system variable `*INIT-USER` is used as the user ID for the logon.
2. If used with Natural Security, `AUTO=ON` disables logons with another user ID (see the *Natural Security* documentation for further information).

18 BP82 - Buffer Pool Placeholder when Object not Found

This Natural profile parameter specifies whether or not a placeholder is put into the Natural Buffer Pool when an object was not found in a library.

Possible settings	`ON`	When an object was not found, a placeholder is put into the buffer pool. **Note:** 1. From the buffer pool manager's point of view, the placeholder is an ordinary Natural object. For the Natural object loader, it is an indicator that the object does not exist and that there is no need to look for it in the system file. 2. For further details, see ***Example***.
	`OFF`	When an object was not found, no placeholder is put into the buffer pool.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Example:

Assume the Natural buffer pool is empty, you have started Natural with the parameter `STEPLIB=XYSTEP`, made a `LOGON` to library `XYLIB`, trying to execute the program `XYPROG`, which resides in library `SYSTEM/FUSER`. While loading the program `XYPROG`, Natural searches the object firstly in library `XYLIB`, secondly in library `XYSTEP`, and finally finds it in `SYSTEM/FUSER`.

When the profile parameter `BPSFI` (*Object Search First in Buffer Pool*) is set to `OFF`, every user doing the same will also make database calls to search for the object `XYPROG` in library `XYLIB` and `XYSTEP`, but these database calls will never succeed. To prevent these unnecessary database calls, a placeholder is put into the buffer pool.

When you use the SYSBPM utility to look into the buffer pool, you will see the placeholder as an ordinary object:

```
14:34:39             ***** NATURAL SYSBPM UTILITY *****              2011-05-02
BPNAME QA82GBP                - List Objects -                  Type Global Nat
BPPROP OFF                                                        Loc DAEF QA82
C  Library  Object    DBID  FNR Loc RLD Use Max  Reuse     TotalUC ObjSize Sto
   *_______ ________ _____ _____ ___ ___ ___ ___ ______ ___________ _______ ___
__ XYLIB    XYPROG      10   32 B             1                  1      90   4
__ SYSTEM   XYPROG      10   32 B             1                  1   2,656   4
__ XYSTEP   XYPROG      10   32 B             1                  1      90   4
```

19 BPC64 - Buffer Pool Cache Storage Type

This Natural profile parameter specifies the type of storage for the buffer pool cache of a local Natural buffer pool.

Possible settings	`ON`	This indicates that virtual storage above the 2 GB line is to be used for the buffer pool cache.
	`OFF`	This indicates that a data space is to be used for the buffer pool cache.
Default setting	`OFF`	
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. This Natural profile parameter is applicable under z/OS only (not for Com-plete).
2. It corresponds to the `C64` subparameter of the `BPI` profile parameter or `NTBPI` macro.
3. Internally, the `BPC64` specification is converted into the equivalent `BPI` specification; for example: `BPC64=ON` is converted into `BPI=(TYPE=NAT,SEQ=0,C64=ON)`.
4. The `BPC64` parameter only applies to the primary Natural buffer pool (`TYPE=NAT`, `SEQ=0`). In the case of a global buffer pool, it is ignored. If there is a primary buffer pool with `SEQ=0` in the Natural parameter module, only the C64 setting of this buffer pool is updated.
5. In multi-user environments (for example, under CICS), the `BPC64` profile parameter only affects the very first Natural session which initializes the local buffer pool.
6. For general information on the Natural Buffer Pool, see *Natural Buffer Pool* in the *Operations* documentation.

20 BPCSIZE - Cache Size for Natural Buffer Pool

This Natural profile parameter specifies the size of the buffer pool cache (in KB) for a Natural local buffer pool.

Possible settings	`0`	If `BPCSIZE=0` is set, no buffer pool cache is used.
	`100 - 2097148` (2 GB - 4 KB)	The buffer pool cache size in KB for cache in data space; that is, with `C64`=`OFF`. **Note:** The specified value is rounded to the next 4 KB boundary.
	`100 - 58720256` (56 GB)	The buffer pool cache size in KB for cache “above the bar”; that is, with `C64`=`ON`. **Note:** The specified value is rounded to the next 1 MB boundary.
Default setting	`0`	By default, no buffer pool cache is used.
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. This Natural profile parameter is applicable under z/OS and z/VSE only (not for Com-plete and not for IMS TM).
2. It corresponds to the `CSIZE` subparameter of the `BPI` profile parameter or `NTBPI` macro.
3. Internally, the `BPCSIZE` specification is converted into the equivalent `BPI` specification, for example: `BPCSIZE=4000` is converted into `BPI=(TYPE=NAT,SEQ=0,CSIZE=4000)`.
4. If the value specified exceeds the possible maximum, the possible maximum value will be taken instead.

5. The `BPCSIZE` parameter applies to the primary Natural buffer pool (`TYPE=NAT`, `SEQ=0`) only. In the case of a Natural global buffer pool, it is ignored. If there is a primary buffer pool with `SEQ=0` in the Natural parameter module, only the cache size setting of this buffer pool is updated.
6. In multi-user environments (for example, under CICS), the `BPCSIZE` parameter affects the very first Natural session only, which initializes the Natural local buffer pool.
7. The type of storage to be used for the buffer pool cache is determined by profile parameter `BPC64` or subparameter `C64` of profile parameter `BPI` or macro `NTBPI`.
8. For more information, see *Buffer Pool Cache* in the *Operations* documentation.

21 BPI - Buffer Pool Initialization

This Natural profile parameter is used to assign buffer pools to a Natural session. It corresponds to the `NTBPI` macro in the Natural parameter module.

Possible settings	See *BPI Parameter Syntax*.	
Default setting	See *Keyword Subparameters*.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTBPI` is used instead.
Specification within session	no	

Notes:

1. There are several types of buffer pools for different purposes. It is possible to define backup buffer pools (see **examples** below).
2. If a buffer pool is unavailable, Natural tries to setup a backup buffer pool of the same type with the next higher sequence number.

The following topics are covered below:

BPI Parameter Syntax

The BPI parameter is specified as follows:

BPI=(*keyword-subparameter=value*,*keyword-subparameter=value*,...)

Or, to delete all buffer-pool definitions for a certain type, omit `SEQ` and specify:

BPI=(**TYPE**=*value*,OFF)

See *Keyword Subparameters*.

Notes:

1. `BPI=(TYPE=NAT,OFF)` deletes all user buffer-pool definitions for Natural; that is, the default values are used for the Natural buffer pool.
2. If `OFF` is used, it must be specified as the last value after `TYPE` and `SEQ`.
3. To dynamically deactivate a buffer-pool definition, use the special value OFF as follows:
   If you use the `BPI` parameter to override an existing buffer pool definition in the Natural parameter module, you must specify new settings in all those subparameters which are to be changed; if you do not, the old settings will still be used.
   If, for example, you want to change from a global to a local buffer pool, you must specify: `NAME=' '` (blank).

If you use the `BPI` parameter to dynamically add a new backup buffer pool definition, you must use the `SEQ` subparameter to specify a sequence number for it. If you omit the `SEQ` specification, the definition of the primary buffer pool (`SEQ=0`) will be overwritten.
The `NAME`, `SIZE`, `LIST`, `TXTSIZE`, `CSIZE`, `METHOD` and `C64` specifications for the primary buffer pool (`SEQ=0`) can also be set dynamically with the profile parameters `BPNAME`, `BPSIZE`, `BPLIST`, `BPTEXT`, `BPCSIZE`, `BPMETH` and `BPC64`.

NTBPI Macro Syntax

The `NTBPI` macro is specified as follows:

```
         NTBPI C64=value,                                                 *
               CSIZE=value,                                               *
               LIST=value,                                                *
               METHOD=value,                                              *
               NAME=value,                                                *
               SEQ=value,                                                 *
               SIZE=value,                                                *
               TXTSIZE=value,                                             *
               TYPE=value
```

See *Keyword Subparameters*.

Note: The value `OFF`, which is available in the syntax of profile parameter `BPI`, cannot be specified in the macro `NTBPI`.

Keyword Subparameters

`C64` | `CSIZE` | `LIST` | `METHOD` | `NAME` | `SEQ` | `SIZE` | `TXTSIZE` | `TYPE`

Notes:

1. The keyword subparameters `SIZE`, `CSIZE`, `TXTSIZE`, `METHOD` and `C64` do not apply to global buffer pools. They are honored only for the very first session which initializes a local buffer pool.
2. Under BS2000, the subparameters `SIZE` and `CSIZE` are ignored.

C64 - Type of Buffer Pool Cache Storage

`C64=value` determines the type of storage to be used for the buffer pool cache.

Value	Explanation
`ON`	Indicates that a memory object “above the bar” (that is, in 64-bit memory) is to be used for the buffer pool cache. Note that `C64=ON` is honored only if the prerequisites are met, namely: ■ z/... architecture hardware, ■ operating system z/OS Version 1.2 or higher. If these prerequisites are not met, the default value is taken.
`OFF`	Indicates that a data space is to be used for the buffer pool cache. This is the default value.

Notes:

1. This subparameter applies to Natural local buffer pools (`TYPE=NAT`) under z/OS only (not for Com-plete).
2. A buffer pool cache is used only if `BPI` subparameter `CSIZE` or profile parameter `BPCSIZE` is set to a non-zero value.
3. The `C64` specification can be overridden dynamically with the profile parameter `BPC64`.

CSIZE - Size of the Local Buffer Pool Cache

`CSIZE=value` determines the size of the buffer pool cache.

Value	Explanation
`0`	If `BPCSIZE=0` is set, no buffer pool cache is used. This is the default value.
`100 - 2097148` (2 GB - 4 KB)	The buffer pool cache size in KB for cache in data space; that is, with `C64=OFF`. **Note:** The specified value is rounded to the next 4 KB boundary.
`100 - 58720256` (56 GB)	The buffer pool cache size in KB for cache “above the bar”; that is, with `C64=ON`. **Note:** The specified value is rounded to the next 1 MB boundary.

Notes:

1. If the value specified exceeds the possible maximum value, the possible maximum value will be taken instead.

2. The `CSIZE` specification applies to Natural local buffer pools (`TYPE=NAT`) only (not for Complete).
3. Under CICS: The `CSIZE` specification applies to swap pools (`TYPE=SWAP`). The value of the `CSIZE` parameter for a swap pool under CICS must be at least twice the maximum thread size of the associated Natural under CICS environment (see `NCMTGD` macro in the *Natural under CICS* documentation), otherwise the `CSIZE` parameter is ignored. This maximum thread size also has to be provided as roll buffer within the swap pool size specification.
4. The `CSIZE` specification can be overridden dynamically with the profile parameter `BPCSIZE` (only in case of `TYPE=NAT`). To determine the type of storage for the buffer pool cache, subparameter `C64` can be used.
5. Under BS2000: `CSIZE` is ignored.
6. For further information, see *Buffer Pool Cache* in the *Operations* documentation.

LIST - Name of Preload List to be Used

`LIST=value` determines the name of the preload list to be used for this buffer pool.

Value	Explanation
1 - 8 characters	The name of the preload list to be used for this buffer pool.
`' '` (blank)	If `LIST=' '` (blank) is set, no preload list is used. This is the default value.

Notes:

1. This subparameter applies only to Natural local buffer pools (`TYPE=NAT`).
2. The `LIST` specification can be overridden dynamically with the profile parameter `BPLIST`.
3. For more information on preload lists, see *Preload List* in the *Operations* documentation.
4. Preload lists are maintained with the SYSBPM utility; see *Preload List Maintenance* in the *Utilities* documentation.

METHOD - Search Algorithm for Allocating Space in Buffer Pool

`METHOD=value` determines the algorithm for allocating storage in the buffer pool.

Value	Explanation
`S`	Indicates that a selection process is to be used for allocating storage. The selection process consists of browsing the whole buffer pool directory and comparing different entries in order to find the most suitable entry. This method was formerly known as "Algorithm 1+2".
`N`	Indicates that the next available unused or free space is to be used. The search for the next available space is done from a pointer to a directory entry. The pointer moves in a wrap around fashion. This method may be used in combination with a buffer pool cache. This is the default value.

Notes:

1. This subparameter applies only to Natural local buffer pools (`TYPE=NAT`).
2. The `METHOD` specification can be overridden dynamically with the profile parameter `BPMETH`.
3. For further information, see *Buffer Pool Search Methods* in the *Operations* documentation

NAME - Name of Global Buffer Pool

`NAME=value` determines the name of the global buffer pool.

Value	Explanation
1 - 8 characters	The name of the global buffer pool to be used.
' ' (blank)	A Natural local buffer pool is used. This is the default value.

Notes:

1. This subparameter applies only to Natural global buffer pools, and to swap pools (`TYPE=SWAP`) under CICS.
2. For `TYPE=SWAP`, the name is the swap pool name which is the key of the associated swap pool definitions in the Natural system file `FNAT` or `FUSER`; see parameter `SWPINIT` in *Natural Swap Pool Initialization Control* in the *Operations* documentation.
3. For a Natural local buffer pool, the name is blank.
4. Under BS2000: An `ADDON` macro with the same value for the keyword subparameter `NAME` is required in the `BS2STUB` used.
5. Under IMS TM: Because a Natural session may be executed in different regions, an editor local buffer pool is not possible, but only an editor global buffer pool.
6. The `NAME` specification can be overridden dynamically with the profile parameter `BPNAME` (with `TYPE=NAT` only).

SEQ - Sequence Number of Buffer Pool

`SEQ=`*`value`* determines the sequence number of the buffer pool.

Value	Explanation
`0` or `1` - `9`	The sequence number of the buffer pool.
`0`	This is the default value.

Notes:

1. The buffer pool defined with the lowest sequence number is called primary buffer pool.
2. For every buffer pool type, except `TYPE=SWAP`, you can define one primary buffer pool and one or more backup buffer pools; that is, alternative buffer pools (of the same type, but with a different sequence number) which will be used if the primary buffer pool is not available at session initialization or cannot be allocated.
3. Buffer pools of the same type are sorted in order of sequence numbers (should two pools of the same type have the same sequence number, they will be sorted in the order in which they are specified).
4. If a requested buffer pool is not available, the buffer pool of the same type with the next higher sequence number will be used instead. If that one is not available either, the one with the next higher number will be used, and so on.

SIZE - Size of Buffer Pool

`SIZE=`*`value`* determines the size of the buffer pool.

Value	Explanation
`256` - `2097151`	The buffer pool size in KB for Natural buffer pools.
`100` - `2097151`	The buffer pool size in KB for other buffer pool types
`256`	This is the default value.

Notes:

1. This subparameter applies to local buffer pools only.
2. Under CICS: The required “logical” minimum size of a swap pool under CICS computes as total of the `NTSWPRM` macro `SWPSLSZ` parameter slot sizes times their (implicit or explicit) factor plus 2 KB for each logical swap pool. When using a swap pool cache (see `CSIZE` subparameter), additionally the maximum thread size of the associated Natural under CICS environment is required for a roll buffer; that is, it has to be added to the computed value.
3. In case of a Natural buffer pool (`TYPE=NAT`), the `SIZE` specification can be overridden dynamically with the profile parameter `BPSIZE`.

4. Under BS2000: The `SIZE` specification is ignored.

TXTSIZE - Size of Buffer Pool Text Segments

`TXTSIZE=value` specifies the size of the segments into which the text pool area of the Natural buffer pool is divided.

Value	Explanation
`1`, `2`, `4`, `8`, `12` or `16`	The size of the buffer pool text segments in KB.
`4`	This is the default value.

Notes:

1. This subparameter applies to local buffer pools of `TYPE=NAT`, `TYPE=SORT`, and `TYPE=DLI`.
2. In multi-user environments (for example, under CICS), the `TXTSIZE` specification only affects the very first Natural session which initializes the local buffer pool.
3. In case of `TYPE=NAT`, the `TXTSIZE` specification can be overridden dynamically with the profile parameter `BPTEXT`.

TYPE - Type of Buffer Pool

`TYPE=value` determines the type of the buffer pool.

Value	Explanation
`NAT`	Natural buffer pool. This is the default value.
`DLI`	DL/I buffer pool; see *Control Blocks in Separate Buffer Pool* in the *Natural for DL/I* documentation.
`EDIT`	Software AG Editor buffer pool; see *Editor Buffer Pool* in the *Operations* documentation. Alternatively, an editor auxiliary buffer pool can be defined per session; see profile parameter `EDPSIZE`.
`SORT`	Sort buffer pool; see also keyword subparameter `STORAGE` of profile parameter `SORT` and macro `NTSORT`.
`MON`	Buffer pool for the monitoring function (`SYSMON`) of the SYSTP utility; see *Natural Monitoring (SYSMON)* in the *Utilities* documentation.
`MSG`	Message buffer pool; see *Message Buffer Pool* in the *Operations* documentation. **Note:** For this type of buffer pool, only the subparameter `NAME` will be honored.
`SWAP`	Buffer pool to hold the Natural CICS swap pool; see *Natural Swap Pool under CICS* in the *TP Monitor Interfaces* documentation.

Notes:

1. Buffer pools of type `NAT`, `DLI` and `SORT` can be managed with the `SYSBPM` utility: see *SYSBPM Utility - Buffer Pool Management* in the *Utilities* documentation.
2. For general information on the Natural buffer pool, see *Natural Buffer Pools* in the *Operations* documentation.

Examples of BPI Parameter

Example 1:

```
BPI=(NAME=' ',SIZE=2000,METHOD=N)
```

The primary buffer pool is replaced by a local buffer pool of 2000 KB. This definition is equivalent to:

```
BPNAME=' ',BPSIZE=2000,BPMETH=N
```

Example 2:

```
BPI=(SEQ=0,NAME=LBP1),BPI=(SEQ=1,NAME=LBP2),BPI=(SEQ=2,SIZE=500)
```

First, Natural tries to allocate a global Natural buffer pool with the name `LBP1`. If this buffer pool is not found, it tries to allocate `LBP2`. If this is not found, it allocates a local buffer pool with a size of 500 KB.

Example 3:

```
BPI=(SEQ=0,TYPE=EDITOR,NAME=LBPE1),BPI=(SEQ=1,TYPE=EDITOR,SIZE=500)
```

First, Natural tries to locate a global editor buffer pool with the name `LBPE1`. If this is not found, it allocates a local editor buffer pool with a size of 500 KB.

Example 4:

```
BPI=(TYPE=SWAP,SIZE=500,NAME=SWAPPOOL,CSIZE=2000)
```

A Natural local swap pool named `SWAPPOOL` having a size of 500 KB and a cache size of 2000 KB is allocated.

Examples of NTBPI Macros

```
         NTBPI TYPE=NAT,                                                         *
               SEQ=0,                                                            *
               NAME=NATBP1
         NTBPI TYPE=NAT,                                                         *
               SEQ=1,                                                            *
               NAME=NATBP2
         NTBPI TYPE=NAT,                                                         *
               SEQ=2,                                                            *
               SIZE=1000,                                                        *
               METHOD=N
```

These examples define multiple Natural buffer pools. If the global buffer pool `NATBP1` is not available, the global buffer pool `NATBP2` will be used instead. If the latter is not available either, a local buffer pool with a size of 1000 KB will be used.

22 BPLIST - Name of Preload List for Natural Buffer Pool

This Natural profile parameter specifies the name of a preload list to be used for the Natural buffer pool.

Possible settings	1 - 8 characters or `' '` (blank)	Name of a preload list to be used for the Natural buffer pool. If `BPLIST=' '` (blank) is set, no preload list is used.
Default setting	`' '` (blank)	
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. This Natural profile parameter corresponds to the `LIST` specification of the `BPI` profile parameter or the `NTBPI` macro.
2. Internally, the `BPLIST` specification is converted into an equivalent `BPI` specification; for example: `BPLIST=LIST3` is converted into `BPI=(TYPE=NAT,SEQ=0,LIST=LIST3)`.
3. It only applies to the primary Natural buffer pool (`TYPE=NAT`, `SEQ=0`). If there is a primary buffer pool with `SEQ=0` in the Natural parameter module, only the `LIST` setting of this buffer pool is updated.
4. For general information, see *Natural Buffer Pool* in the *Operations* documentation.

23 BPMETH - Buffer Pool Space Search Algorithm

This Natural profile parameter specifies the search algorithm that is to be used for allocating storage in the Natural buffer pool.

Possible settings	`S`	This indicates that a selection process is to be used for allocating storage. The selection process consists of browsing the whole buffer pool directory and comparing different entries in order to find the most suitable entry. This method was formerly known as Algorithm 1+2.
	`N`	This indicates that the next available unused or free space is to be used. The search for the next available space is done from a pointer to a directory entry. The pointer moves in a wrap around fashion. This method may be used in combination with a buffer pool cache.
Default setting	`N`	
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. This Natural profile parameter corresponds to the `METHOD` subparameter of the `BPI` profile parameter or the `NTBPI` macro.
2. Internally, the `BPMETH` specification is converted into the equivalent `BPI` specification; for example: `BPMETH=S` is converted into: `BPI=(TYPE=NAT,SEQ=0,METHOD=S)`.
3. The `BPMETH` parameter only applies to the primary Natural buffer pool (`TYPE=NAT`, `SEQ=0`). In the case of a global buffer pool, it is ignored. If there is a primary buffer pool with `SEQ=0` in the Natural parameter module, only the `METHOD` setting of this buffer pool is updated.
4. In multi-user environments (for example, under CICS), the `BPMETH` profile parameter only affects the very first Natural session which initializes the local buffer pool.

5. For general information on the Natural buffer pool, see *Natural Buffer Pool* in the *Operations* documentation.

24 BPNAME - Name of Natural Global Buffer Pool

This Natural profile parameter specifies the name of the Natural global buffer pool.

Possible settings	1 - 8 characters or `' '` (blank)	Name of the Natural global buffer pool. If `BPNAME=' '` (blank) is set, a *local* Natural buffer pool is used.
Default setting	`' '` (blank)	
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. This parameter can only be specified dynamically. It corresponds to the `NAME` specification of the `BPI` profile parameter or the `NTBPI` macro respectively.
2. Internally, the `BPNAME` specification is converted into an equivalent `BPI` specification; for example: `BPNAME=GBP1` is converted into: `BPI=(TYPE=NAT,SEQ=0,NAME=GBP1)`.
3. The `BPNAME` profile parameter only applies to the primary Natural global buffer pool (`TYPE=NAT`, `SEQ=0`).If there is a primary buffer pool with `SEQ=0` in the Natural parameter module, only the `NAME` setting of this buffer pool is updated.
4. For general information, see *Natural Global Buffer Pool* in the *Operations* documentation.

25 BPPROP - Global Buffer Pool Propagation

This Natural profile parameter controls the propagation of changes to an object in a buffer pool. If a modification occurs affecting a Natural object residing in one (global or local) buffer pool, this modification can be propagated to other global buffer pools - this will ensure the consistency of the buffer pools.

Possible settings	`OFF`	Changes are not propagated to any other global buffer pool. **Note:** Under z/OS, any setting other than `OFF` requires that the Authorized Services Manager is active.
	`GLOBAL`	Changes are propagated to all other global buffer pools. **Note:** In a z/OS Parallel Sysplex environment:, the changes are only propagated within the current z/OS image. (See **Note ***)
	`PLEX`	Changes are propagated to all other global buffer pools of the same name within the entire z/OS Parallel Sysplex environment. (See **Note ***)
	`GPLEX`	Changes are propagated to all other global buffer pools within the entire z/OS Parallel Sysplex environment. (See **Note ***) **Note:** Under BS2000, the setting `GPLEX` has the same effect as `GLOBAL`.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies under z/OS and BS2000.
2. * Under z/OS, the propagation is always restricted to the Natural subsystem in which the change has occurred; that is, the scope of the propagation, as set with the `BPPROP` parameter, applies

only within that subsystem, but not to other subsystems. For details, see *Natural Subsystem* in the *Operations* documentation.

3. For further information on the propagation, see *Natural Global Buffer Pool* in the *Operations* documentation.

26 BPSFI - Object Search First in Buffer Pool

This Natural profile parameter determines the sequence in which a requested object that is to be executed is searched for in the buffer pool and in the system file(s).

You can choose between three search sequences:

Possible settings	`ON`	Search Sequence 1 is used (search buffer pool first for all libraries, then the system file(s)). Natural looks for the object in the following sequence until it is found: 1. in the buffer pool, first in the current library, then in one steplib after another, then in the two `SYSTEM` libraries; 2. in the system file(s), first in the current library, then in one steplib after another, then in the two `SYSTEM` libraries. For performance reasons, it is recommended that you set `BPSFI=ON` in production environments. **Caution:** If you set `BPFSI=ON`, make sure that object names are unique across all libraries that are involved in the search. If objects with the same name exist in different libraries being searched, unpredictable results may occur.
	`OFF`	Search Sequence 2 is used (alternating search in buffer pool and system file(s) for each library). Natural looks for the object in the following sequence until it is found: 1. in the current library, first in the buffer pool, then in the system file(s); 2. in one steplib after another, first in the buffer pool, then in the system file(s) for each steplib; 3. in the two `SYSTEM` libraries, first in the buffer pool, then in the system file(s) for each library.

		`BPSFI=OFF` is recommended in development environments to always get the most current object from your own current library.
	`LIB`	Search Sequence 3 is used. Natural looks for the object in the following sequence until it is found: 1. in the current library, first in the buffer pool, then in the system file(s); 2. in the buffer pool in one steplib after another, then in the two `SYSTEM` libraries; 3. in the system file(s) in one steplib after another, then in the two `SYSTEM` libraries.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. For further information, see *Steplib Libraries* and *Search Sequence for Object Execution* in the *Using Natural* documentation.
2. See also profile parameter `BP82` (*Buffer Pool Placeholder when Object not Found*).

27 BPSIZE - Size of Natural Local Buffer Pool

This Natural profile parameter specifies the size of the Natural local buffer pool.

Possible settings	`256 - 2097151`	Size of the Natural local buffer pool in KB.
Default setting	`256`	
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. This Natural profile parameter corresponds to the `SIZE` specification of the `BPI` profile parameter or the `NTBPI` macro.
2. `BPSIZE` only applies to the primary Natural local buffer pool (`TYPE=NAT`, `SEQ=0`). For a global buffer pool, it is ignored. If there is a primary buffer pool with `SEQ=0` in the Natural parameter module, only the `SIZE` setting of this buffer pool is updated.
3. In multi-user environments (for example, under CICS), the `BPSIZE` parameter only affects the very first Natural session, which initializes the local buffer pool.
4. Under Com-plete, the size of a local buffer pool is set as described in the corresponding Natural installation documentation.
5. Under BS2000, the size of a local buffer pool is specified with the parameter `SIZE` of the `ADDON` macro.
6. Internally, the `BPSIZE` specification is converted into an equivalent `BPI` specification; for example: `BPSIZE=1500` is converted into `BPI=(TYPE=NAT,SEQ=0,SIZE=1500)`.
7. For general information, see *Natural Buffer Pool* in the *Operations* documentation.

28 BPTEXT - Size of Text Segments in Natural Buffer Pool

This Natural profile parameter specifies the size of the segments into which the text pool area of the Natural buffer pool is divided.

Possible settings	`1`, `2`, `4`, `8`, `12` or `16`	Size of the buffer pool text segments in KB.
Default setting	`4`	
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. This Natural profile parameter corresponds to the `TXTSIZE` specification of the `BPI` profile parameter or the `NTBPI` macro.
2. Internally, the `BPTEXT` specification is converted into an equivalent `BPI` specification; for example: `BPTEXT=4` is converted into `BPI=(TYPE=NAT,SEQ=0,TXTSIZE=4)`.
3. The `BPTEXT` parameter only applies to the primary Natural buffer pool (`TYPE=NAT`, `SEQ=0`). In the case of a global buffer pool, it is ignored. If there is a primary buffer pool with `SEQ=0` in the Natural parameter module, only the `TXTSIZE` setting of this buffer pool is updated.
4. In multi-user environments (for example, under CICS), the `BPTEXT` parameter only affects the very first Natural session, which initializes the local buffer pool.
5. For general information on the Natural buffer pool, see *Natural Buffer Pool* in the *Operations* documentation.

29 BSIZE - Size of EntireX Broker Buffer

This Natural profile parameter specifies the size of the EntireX Broker buffer.

Possible settings	`1 - 64`	Buffer size in KB.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies if EntireX Broker is installed.
2. Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.
3. Currently, if EntireX Broker is used, EntireX Broker specifies the buffer size automatically.

30 BX - Box Definition

With this session parameter, you specify which parts of a box are to be displayed for field outlining.

Notes:

1. Outlining ("boxing") is the capability to generate a line around certain fields when they are displayed on the terminal screen. Drawing such boxes around fields is another method of showing the user the lengths of fields and their positions on the screen. The outlining feature is only available on certain types of terminals, usually those which also support the display of double-byte character sets. If the terminal used does not support outlining, this parameter will be ignored at execution time.
2. See also terminal command `%D=`.

Possible settings	`T`	Top horizontal line. See Note 1.
	`B`	Bottom horizontal line. See Note 1.
	`L`	Lefthand vertical line. See Notes 1 and 2.
	`R`	Righthand vertical line. See Notes 1 and 2.
	`ON`	Corresponds to `BX=TBLR`.
	`OFF`	Causes no boxes to be drawn around the fields concerned.
Default setting	`OFF`	
Applicable statements	`FORMAT`	

	`DISPLAY`
	`INPUT`
	`WRITE`
		Parameter may be specified at statement level and/or at element level.

Applicable command	none	

Notes:

1. You can specify the values `T`, `B`, `L`, `R` in any order.

2. If you use the session parameter settings `BX=L` or `BX=R`, you should switch off Natural's screen optimization using the profile parameter setting `DSC=OFF` or the Natural terminal command `%RO`.

Example:

```
DISPLAY #FIELD1 (BX=RLT) /
        #FIELD2 (BX=TLRB)
```

31 CANCEL - Session Cancellation with Dump

This Natural profile parameter can be used to specify a character string that will cause the Natural session to be terminated with a dump. This may be useful for debugging purposes.

Possible settings	1 - 8 characters	When you enter this character string in any input field within your Natural session (beginning in the first input field), the session will be terminated immediately and a dump will be produced.
Default setting	`*CANCEL`	
Dynamic specification	yes	
Specification within session	no	

32

CC - Error Processing in Batch Mode

This Natural profile and session parameter specifies the action to be taken if an error is detected during the compilation/execution of a Natural program in batch mode.

Possible settings	`ON`	Natural flushes the input data stream for the batch input files `CMSYNIN` and `CMOBJIN` until a line containing `%%` in the first two positions is encountered or until an end-of-file condition is detected. If more data are available in the input stream, Natural resumes reading after the line containing `%%`.	
	`OFF`	Natural attempts to process the next program (or command) in the input stream.	
Default setting	`OFF`		
Dynamic specification	yes		
Specification within session	yes	Applicable Statements:	`SET GLOBALS`
		Applicable command:	`GLOBALS`
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.	

Notes:

1. This Natural profile and session parameter only applies in batch mode.
2. It does not apply if user-written error-handling routines are used.
3. Within a Natural session, the profile parameter `CC` can be overridden by the session parameter `CC`.
4. When a Natural session terminates, Return Code 4 is passed to the invoking program with Register 15 if an error is detected (regardless of the `CC` setting).

33

CCHAR - Allow Output Control Characters

To avoid screen I/O errors, Natural automatically translates the output control characters `x'01'` through `x'3F'` to `'?'`. In some cases, however, certain control characters are required for special purposes

This Natural profile parameter allows you to define hexadecimal control characters for primary I/O to be passed through unchanged. This overwrites the definitions in the output translation tables `NTTAB`, `NTTAB1` and `NTTABL` as contained in the configuration module `NATCONFG` or defined by the corresponding dynamic profile parameter or by the corresponding macro in the Natural parameter module. It avoids warning message NAT7021 during session initialization if profile parameter `CFICU` is set to `ON`.

Possible settings	`a1`
	`(a1,a2,…)`
		The following can be specified:
		a single hex character or
		a list of hex characters enclosed in brackets can be specified. The hex characters must be within `x'01'` through `x'3F'`. A hex character range `a1-a2` is allowed instead of a hex character.

	`OFF`	`OFF` resets any previous `CCHAR` definitions.
Default setting	no	
Dynamic specification	yes	
Specification within session	no	

Note: For additional print files, the subparameter `CCHAR` of profile parameter `PRINT` can be used.

Examples:

```
CCHAR=17
CCHAR=19-1B
CCHAR=(03-06,0A,1B,3A-3F)
```

34 CCTAB - Printer Escape Sequence Definition

This Natural profile parameter is used to set up a table of printer-control sequences, which is used for printing additional reports and hardcopies. It corresponds to the `NTCCTAB` macro in the Natural parameter module.

- It is possible to either translate Natural field attributes into escape sequences or specify special characters to be translated into escape sequences.
- In addition, strings can be specified which are always sent as the first output record after an open operation or as the last output record before a close operation.
- This means that by using the right profile name, you can activate your printout either in portrait mode or in landscape. Then you can use all print features of this device by using simple attributes in Natural. This makes even bar-code printing or double-height printing possible.
- `CCTAB` defines tables which are used to recognize special characters in output fields and replace them with the defined control sequences. The parameter also defines the Natural attributes which are used to insert the defined control sequences.

Possible settings	See *CCTAB Parameter Syntax*.	
Default setting	As specified within the macro `NTCCTAB` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTCCTAB` is used instead.
Specification within session	no	

The following topics are covered below:

CCTAB Parameter Syntax

For each profile, a separate `CCTAB` parameter must be specified. The `CCTAB` parameter can be specified in three variants:

1st Variant

CCTAB=(*name*,**OPN**='*xxxxx*',**CLS**='*yyyyy*')

Where:

`name`	is the name of the profile form; that is, the `DEFINE PRINTER (n) OUTPUT 'nnnnn' PROFILE 'name'`, which is required and which has a maximum length of 8 bytes.
`OPN='xxxxx'`	is optional and defines a data string (up to 250 bytes) which is sent to the printer with each open operation.
`CLS='yyyyy'`	is optional and defines a data string (up to 250 bytes) which is sent to the printer before each close operation.

Note: `OPN` and `CLS` can be specified in any sequence.

2nd Variant

CCTAB=(*name*,**CODE**='*n*',**CS**='*xxxx*')

Where:

`CODE='n'`	is a character which is recognized by Natural once it appears in the output string.
`CS='xxxx'`	is the string to replace the `CODE` character.

Note: The `CS` subparameter must follow the `CODE` subparameter.

3rd Variant

CCTAB=(*name*,**ATR**=*nnnn*,**CSS**='*xxxx*',**CSE**='*yyyy*')

Where:

`ATR='nnnn'`	is the Natural internal field attribute. The name is defined by the macro `NAMATR`.
`CSS='xxxx'`	is the string (up to 20 bytes) which is inserted before the field. `CSS` is mandatory.
`CSE='yyyy'`	is the string (up to 20 bytes) which is inserted after the field. `CSE` is mandatory.

Note: The `CSS` and `CSE` subparameters must follow the `ATR` subparameter.

NTCCTAB Macro Syntax

The `NTCCTAB` macro can be specified in three variants:

1st Variant

```
         NTCCTAB name,                                                    *
               OPN='xxxxx',                                               *
               CLS='yyyyy'
```

For details, refer to the `CCTAB` parameter syntax, ***1st Variant***.

2nd Variant

```
         NTCCTAB name,                                                    *
               CODE='n',                                                  *
               CS='xxxx'
```

For details, refer to the `CCTAB` parameter syntax, ***2nd Variant***.

3rd Variant

```
         NTCCTAB name,                                                    *
               ATR=nnnn,                                                  *
               CSS='xxxx',                                                *
               CSE='yyyy'
```

For details, refer to the `CCTAB` parameter syntax, ***3rd Variant***.

String Syntax for OPN, CLS, CODE, CS, CSS or CSE

You specify character strings either as characters (enclosed in apostrophes) or as the corresponding hexadecimal representation of the characters (without apostrophes).

Proportional Fonts

If you use proportional fonts, be sure to return to a fixed-spacing font before using tables where you need correct positioning.

Examples of CCTAB Parameter

```
CCTAB=(DBCST,CODE=0E,CS=400E,CODE=0F,CS=0F40,ATR=P5DBCS,CSS=0E,CSE=0F)
```

```
CCTAB=(OPN=27C5274DA2F1F188275093F0D6,CLS='LAST LINE')
```

Examples of NTCCTAB Macros

```
         NTCCTAB DBCST
         NTCCTAB CODE=0E,CS=400E
         NTCCTAB CODE=0F,CS=0F40<
         NTCCTAB ATR=P5DBCS,CSS=0E,CSE=0F
```

```
         NTCCTAB TEST,OPN=27C5274DA2F1F188275093F0D6,CLS='LAST LINE'
         NTCCTAB CODE='<',CS=' B(S0B'
         NTCCTAB CODE='>',CS='B(S3B '
         NTCCTAB CODE='(',CS=' B(S1S'
         NTCCTAB CODE=')',CS='B(S0S '
         NTCCTAB ATR=P2UL,CSS=' B&&DD',CSE='B&&D§'
         NTCCTAB ATR=P2UL,CSS=405FF1C25084C4,CSE=5FF1C250847C
         NTCCTAB ATR=P2ITAL,CSS=' B(S1S',CSE='B(S0S'
         NTCCTAB ATR=P1HIGH,CSS=' B(S3B',CSE='B(S0B'
         NTCCTAB ATR=P2RVID,CSS=' B(S-3B',CSE='B(S0B'
```

35 CD - Color Definition

With this session parameter, you specify the color attributes for fields. If no color screen is used, this parameter will be ignored at runtime.

Related session parameter: `AD` - Attribute Definition

Possible settings	`BL`	blue
	`GR`	green
	`NE`	neutral
	`PI`	pink
	`RE`	red
	`TU`	turquoise
	`YE`	yellow
Default setting	`NE`	
Applicable statements	`FORMAT`	

	`DISPLAY`
	`INPUT`
	`PRINT`
	`WRITE`
		Parameter may be specified at statement level and/or at element level.

	`ASSIGN`
	`MOVE`
	`REINPUT`
		Parameter may be specified at statement level.

Applicable command	none	

Example:

```
INPUT (CD=RE) #A #B
```

36 CDYNAM - Dynamic Loading of Non-Natural Programs

This Natural profile parameter sets the limit for the number of non-Natural programs allowed to be loaded simultaneously during a Natural session. If the specified limit is reached, Natural returns error message NAT0920 indicating that the requested non-Natural programs cannot be loaded dynamically.

Non-Natural programs can only be loaded again after the previously loaded non-Natural programs are deleted. The profile parameter `DELETE` determines when non-Natural programs are deleted after the dynamic loading.

Possible settings	`0`	Dynamic loading of non-Natural programs is not allowed.
	`1 - 1024`	Non-Natural programs can be loaded dynamically up to at least the specified number. The number of programs actually loaded may exceed the specified number depending on the space available in internal program tables.
Default setting	`5`	
Dynamic specification	yes	
Specification within session	no	

37

CF - Character for Terminal Commands

This Natural profile and session parameter specifies the control character for Natural terminal commands; that is, the character which is to be used as the first character of any terminal command.

Possible settings	any special character	A terminal command must begin with the character specified here. The character specified with the `CF` parameter ■ must not be the same as the one specified with the `HI` parameter (help character) or `IA` parameter (input assign character). ■ should not be the same as the one specified with the `DC` parameter (decimal character) or `ID` parameter (input delimiter character). ■ In the map editor, the control character for terminal commands is always "%" (so as to avoid conflicts with delimiter characters used in maps), no matter which character is defined with the `CF` parameter.	
	`OFF`	No control character for terminal commands is available. Terminal commands issued with `SET CONTROL` statements, however, are still accepted.	
Default setting	`%`	A terminal command must begin with the character "%".	
Dynamic specification	yes		
Specification within session	yes	Applicable statements:	`SET GLOBALS`
		Applicable command:	`GLOBALS`
Application programming interface	`USR0350N`, `USR1005N` *	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation. * Recommended.	

Notes:

1. Within a Natural session, the profile parameter `CF` can be overridden by the session parameter `CF`.

2. Under Natural Security: the setting of this parameter can be overridden by the *Session Parameters* option of the Library Profile.

38 CFICU - Unicode and Code Page Support

This Natural profile parameter is required to enable Unicode and code page support for various Unicode settings, for example, if variables with format U or the statement `MOVE ENCODED` are to be used. It corresponds to the `NTCFICU` macro in the Natural parameter module.

Possible settings	See ***CFICU Parameter Syntax***.	
Default setting	`ON` or `OFF`	Enables or disables the Unicode and code page support. The default value is `OFF` when profile parameter `CP` is set to `OFF`. Otherwise, the default value is `ON`.
Dynamic specification	yes	The parameter `CFICU` can only be specified dynamically. In the Natural parameter module, the macro `NTCFICU` is used instead.
Specification within session	no	

Notes:

1. `CFICU=ON` is enforced when profile parameter `CP` is set to any value other than `OFF`.
2. For further information, see *Profile Parameters and Macros* in the *Unicode and Code Page Support* documentation.

The following topics are covered below:

CFICU Parameter Syntax

The `CFICU` profile parameter is specified as follows:

```
CFICU=(ON,keyword-subparameter=value,keyword-subparameter=value,...)
```

Or:

```
CFICU=ON
```

Or:

```
CFICU=(OFF,keyword-subparameter=value,keyword-subparameter=value,...)
```

Or:

```
CFICU=OFF
```

See *Keyword Subparameters*.

NTCFICU Macro Syntax

The `NTCFICU` macro is specified as follows:

```
         NTCFICU ON,                                                   *
               CNVNORM=value,                                          *
               COLLATE=value,                                          *
               COLNORM=value,                                          *
               CPOPT=value,                                            *
               DATFILE=value,                                          *
               DATITEM=value,                                          *
               LOCALE=value1_value2
```

Or:

```
         NTCFICU ON
```

Or:

```
         NTCFICU OFF
```

See *Keyword Subparameters*.

Keyword Subparameters

`CNVNORM` | `COLLATE` | `COLNORM` | `CPOPT` | `DATFILE` | `DATITEM` | `LOCALE`

CNVNORM - Normalization before Conversion

`CNVNORM=value` activates/deactivates normalization before conversion.

Value	Explanation
`ON`	Enable normalization before conversion.
`OFF`	Disable normalization before conversion. **Note:** When `CNVNORM=OFF`, the `MOVE NORMALIZED` statement can be used to normalize selected strings. This is the default value.

Notes:

1. The German character "ä", for example, can be represented in Unicode as `U+00E4` or by using a combining character as `U+0061`, `U+0308`. Conversion to a code page considers the combined "ä" (`U+0061 U+0308`) as two code points and produces an "a" and a substitution character, if `U+0308` is no valid character of the target code page. Normalization before conversion creates one code point `U+00E4` from the combined code points `U+0061 U+0308` and the subsequent conversion will deliver the result "ä".
2. The parameter is honored whenever a conversion from U to A format is performed, for example `MOVE U TO A` or `DISPLAY U`, when the output device is a terminal emulation. The additional operation consumes of course additional storage as well as additional CPU time.

COLLATE - Collation Services

`COLLATE=`*`value`* determines the collation service used.

Value	Explanation
`ON`	Use Locale ID and ICU's collation services to compare Unicode strings. This is the default value.
`OFF`	Use ICU's simple Unicode compare.

Note: Collation is the process of ordering units of textual information (alphabetic sorting). Collation is usually specific to a particular language.

Examples:

- The character "Ä" is sorted in German locale between *A* and "B", but in Swedish locale it is sorted after "Z".
- In Lithuanian, "y" is sorted between "i" and "k".

COLNORM - Normalization Check of Collation Services

`COLNORM=`*`value`* can be used to enable or disable the normalization check.

Value	Explanation
`ON`	Check for un-normalized text.
`OFF`	Disable check for un-normalized text. This is the default value.

Notes:

1. Normalization is the process of removing alternate representations of equivalent sequences from textual data, to convert data into a form that can be binary-compared for equivalence. The ICU Collation Service handles un-normalized text properly, producing the same results as if the text were normalized. This maximizes performance for the majority of text that does require normalization. If Unicode data is known with certainty not to contain un-normalized text, then even the overhead checking for normalization can be eliminated.
2. This subparameter is honored only if subparameter `COLLATE` is set to `ON`.

CPOPT - Fast Code Page Conversion

`CPOPT=`*`value`* can be used to optimize the conversion performance.

Value	Explanation
`ON`	Use internal translation tables instead of ICU functions, if possible.
`OFF`	Use ICU functions in any case. This is the default value.

Note: By default, a conversion from alpha to Unicode format and vice versa is performed by calling ICU functions. Certain code pages are mapping characters to Unicode with 1:1 relationship. In this case, the conversion performance can be increased by using internal translation tables rather than ICU functions.

DATFILE - Additional Data Libraries

`DATFILE=`*`value`* can be used to define the name of an optional ICU data library.

Value	Explanation
1 - 8 characters or `OFF`	The name of the ICU data library.
`OFF`	No additional ICU data library is defined. The default ICU data library is used, which is part of the ICS module (see the *Unicode and Code Page Support* documentation). This is the default value.

Notes:

1. The ICU data library must be eligible for dynamic loading (see profile parameters `RCA` and `RCALIAS` for more information).
2. The ICU data library contains the converter mapping tables, collation rules, break iterator rules and other locale data.

3. The ICU development kit provides tools to build data libraries that comply with particular requirements. Refer to the chapter *ICU Data* in the *ICU User Guide* at ***http://userguide.icu-project.org/icudata*** for more information.
4. You can assign only one ICU data library to a Natural session, but you can assign different data libraries to different Natural sessions. The ICS module supports up to ten different ICU data libraries. ICS searches all available ICU data libraries for a requested item, for example, a converter.
5. The version of the ICU data library and the ICU version must match. If the data library does not match the ICU version, Natural issues the error message NAT3418 with return code 80 during session initialization.

DATITEM - Load Method for ICU Data Items

`DATITEM=`*`value`* can be used to determine the method to load ICU data items under CICS and Complete. For information on ICU data items, see the relevant section in the *Unicode and Code Page Support* documentation.

Value	Explanation
`SVC`	The ICU data items are loaded with the SVC instruction of the operating system.
`NONE`	The ICU data items are loaded with the functions provided by the TP system. This is the default value.

Notes:

1. If you use `DATITEM=NONE` under Com-plete, you must set the keyword parameter `THREAD-ESQA-SIZE=15K` (or a size greater than 15 KB) in the startup options for your Com-plete.
2. If you use `DATITEM=NONE` under CICS, you must add one PPT entry for each ICU data item.

LOCALE - Locale ID

`LOCALE=`*`value1_value2`* determines the Locale ID.

Value	Explanation
`value1_value2`	*`value1`* is a 2- or 3-byte language code of lower-case characters. If specified in upper case, it will be translated into lower case automatically. *`value2`* is a 2- or 3-byte region code of upper-case characters to classify the language.
`en_US`	This is the default value.

Note: The Locale ID is used by ICU's Collation Service to consider language and even region-dependent features of collation. The language code of the Locale ID follows ISO639, and the region code follows ISO 3166.

Examples of Language Code and Reason Code Pairs:

`en_US`	English (United States)
`en_UK`	English (United Kingdom)
`de_DE`	German (Germany)
`de_AT`	German (Austria)
`de_CH`	German (Switzerland)
`sv_SE`	Scandinavian (Sweden)

Example of CFICU Parameter

```
CFICU=(COLNORM=ON,LOCALE='de_DE',DATFILE=TEST15)
```

Example of NTCFICU Macro

```
         NTCFICU COLNORM=ON,                                         *
               LOCALE=de_DE,                                         *
               DATFILE=TEST15
```

39 CFWSIZE (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

40 CICSP - Environment Parameters for Natural CICS Interface

Note: This parameter is only available with Natural CICS Interface Version 8.3.

This Natural profile parameter can only be specified with the `NTCICSP` macro, dynamic parameter specification is not possible yet.

The `NTCICSP` macro is used to define environment-specific parameters for Natural session options relevant in a CICS environment.

Possible settings	See below.
Default setting	See below.
Dynamic specification	no
Specification within session	no

This section covers the following topics:

NTCICSP Macro Syntax

The `NTCICSP` macro is specified as follows:

```
         NTCICSP BACKEND=value,                                       *
               BACKOUT=value,                                         *
               BACKRPL=value,                                         *
               CALLRPL=value,                                         *
               CHAP=value,                                            *
               CNTCALL=value,                                         *
               COMARET=value,                                         *
               DIRNAME=value,                                         *
               DUPTID=value,                                          *
               FDTPX=value,                                           *
               LOGDEST=value,                                         *
               MEMOBJR=value,                                         *
               MSGDEST=value,                                         *
               MSGPFX=value,                                          *
               MSGTRAN=value,                                         *
               PREFIX=value,                                          *
               PRMDEST=value,                                         *
               PSTRNID=value,                                         *
               RCVASYN=value,                                         *
               RESENDC=value,                                         *
               RESENDS=value,                                         *
               RJEDEST=value,                                         *
               SLCALL=value,                                          *
               SLNOHLD=value,                                         *
               SNDLAST=value,                                         *
               STORVIO=value,                                         *
               TERMVAR=value,                                         *
               TIOBSZ=value,                                           *
               TRANCHK=value,                                          *
               TTYCNSL=value,                                          *
               UCTRAN=value,                                           *
               UNITID=value,                                           *
               USERID=value
```

See *Keyword Subparameters*.

Keyword Subparameters

`BACKEND` | `BACKOUT` | `BACKRPL` | `CALLRPL` | `CHAP` | `CNTCALL` | `COMARET` | `DIRNAME` | `DUPTID` | `FDTPX` | `LOGDEST` | `MEMOBJR` | `MSGDEST` | `MSGPFX` | `MSGTRAN` | `PREFIX` | `PRMDEST` | `PSTRNID` | `RCVASYN` | `RESENDC` | `RESENDS` | `RJEDEST` | | `SLCALL` | `SLNOHLD` | `SNDLAST` | `STORVIO` | `TERMVAR` | `TIOBSZ` | `TRANCHK` | `TTYCNSL` | `UCTRAN` | `UNITID` | `USERID`

BACKEND - Back-End Program Invocation Control

`BACKEND=value` defines whether a specified back-end program or transaction is to be invoked after the session has terminated (normally or abnormally).

The `BACKEND` parameter has two sub-parameters. The second sub-parameter is optional. It controls if a back-end program is to be invoked in the event of a terminal error. This also includes session clean-up tasks started by NEP.

Possible values are `ON/OFF` for both sub-parameters, but the default values are different.

Value	Explanation
`ON`	Same as `BACKEND=(ON,OFF)`. This is the default if the `BACKEND` parameter is omitted. A potential back-end program or transaction is always invoked, particularly after task abends, but not in the case of terminal errors. When a back-end program is invoked, the Natural termination message and return code are passed to the CICS transaction work area (TWA). In addition, the same information can be passed to a CICS COMMAREA, as described with the `BACKRPL` parameter.
`(ON,ON)`	Same as `BACKEND=(,ON)`. A potential backend program or transaction is always invoked, particularly after abends including terminal errors.
`OFF`	Forces `BACKEND=(OFF,OFF)`. A potential back-end program or transaction is only invoked if the Natural session has been terminated normally; that is, with a Natural termination message.

BACKOUT - Backout Transaction in the Case of Unrecoverable Abends

`BACKOUT=value` defines whether the Natural CICS Interface is to perform a transaction backout by means of an `EXEC CICS SYNCPOINT ROLLBACK` call or not.

Value	Explanation
`ON`	All pending file updates are backed out. This is the default value.
`OFF`	All pending file updates are committed.

Because of its abnormal termination exit, the Natural CICS Interface intercepts all abends. If an abend is not recoverable, all resources of the abending session are released and the session is terminated via `EXEC CICS RETURN`; that is, it is terminated “normally” in terms of CICS. Thus, at the end of the task, “pending” file updates are not automatically backed out by CICS.

BACKRPL - Location of Parameter List for Back-End Program

`BACKRPL=value` controls where and how the back-end parameters are passed to a back-end program.

Value	Explanation
`ALL`	This is the default. The Natural back-end parameter area mapped by macro `NAMBCKP` is passed both in the CICS TWA and in a CICS COMMAREA (including potential termination data).
`COMA`	The Natural back-end parameter area mapped by macro `NAMBCKP` (including potential termination data) is passed in a CICS COMMAREA only, not in the CICS TWA.
`DATA`	The Natural back-end parameter area mapped by macro `NAMBCKP` is passed in the CICS TWA only, a CICS COMMAREA just holds potential termination data; if no termination data is available, no COMMAREA is passed.
`TWA`	The Natural back-end parameter area mapped by macro `NAMBCKP` is passed in the CICS TWA only, no CICS COMMAREA is passed.

Note: This parameter applies to back-end programs only, not to back-end transactions.

CALLRPL - Location of Parameter List for External Subroutine Programs CALLed by Natural via EXEC CICS LINK

`CALLRPL=value` controls where and how the `CALL` parameter lists are passed to external subroutine programs.

Value	Explanation
`ALL`	This is the default. The Natural parameter list addresses are passed both in the CICS TWA and in a CICS COMMAREA; the length of the passed COMMAREA is controlled by the second sub-parameter.
`COMA`	The Natural parameter list addresses are passed in a CICS COMMAREA only, not in the CICS TWA; the length of the passed COMMAREA is controlled by the second sub-parameter.
`TWA`	The Natural parameter list addresses are passed in the CICS TWA only, not in a CICS COMMAREA; that is, the COMMAREA length then is `0`.

Possible values for the second sub-parameter are:

Value:	Explanation:
`2`	This is the default. Only the parameter address list address and the field description list address (R1 and R2, as described with the `CALL` statement) are passed in a CICS COMMAREA; that is, the COMMAREA length is `8`.
`3`	The field length list address (R3, as described with the `CALL` statement) is passed in addition in a COMMAREA; that is, the COMMAREA length is `12`.
`4`	The field length list address and the large field length list address (R4, as described with the `CALL` statement) are passed in addition in a COMMAREA; that is, the COMMAREA length is `16`.

Example:

```
CALLRPL=(ALL,2)
```

This is the default setting.

Notes:

1. The second sub-parameter applies only if the first sub-parameter is `ALL` or `COMA`.
2. If the CICS TWA is to be used, it always holds all 4 parameter list addresses.
3. If the CICS COMMAREA length is greater than `0`, the last parameter address passed gets a flag saying it is the last address in the list. This flag is set in the high order bit in the address field.
4. The `CALLRPL` parameter does not apply, when passing parameter values in a CICS COMMAREA (`%P=C`); a CICS COMMAREA then is used regardless of the `CALLRPL` parameter setting.

CHAP - Change Task's Dispatching Priority

`CHAP=`*`value`* defines how the Natural CICS Interface is to treat long-running tasks reaching the `DBROLL` and/or `MAXROLL` call limits.

Value	Explanation
`ON`	The task's dispatching priority is decremented by 1 every time it reaches the `DBROLL` and/or `MAXROLL` call limits. The original task dispatching priority is re-established at the next screen I/O.
`OFF`	The session is suspended. This is the default value.

CNTCALL - CICS Call Passing Automatically Data in Container

With `SET CONTROL 'P=C'` the `CALL` statement parameter data is passed in a CICS COMMAREA on the `EXEC CICS LINK` rather than parameter data pointers. As a CICS COMMAREA is limited to 32 KB, `EXEC CICS LINK` with a COMMAREA greater than 32 KB will fail due to a LENGERR condition.

`CNTCALL=`*`value`* enables you to automatically use a container on `EXEC CICS LINK` when the data to be passed exceeds the maximum COMMAREA length of 32 KB. This functionality only works if the CICS Transaction Server in your z/OS environment supports channels and containers.

The default container name then is `NCI-COMMAREA` unless explicitly specified via the application programming interface `USR4204N` prior to the Natural `CALL` statement.

Value	Explanation
`ON`	When the COMMAREA data would exceed 32 KB, the Natural CICS Interface automatically uses a CICS container on the EXEC CICS LINK, using `NCI-COMMAREA` as default name.
`OFF`	When the COMMAREA data would exceed 32 KB, the Natural `CALL` statement fails with a `NAT0920` message and reason code `LENGERR` (hexadecimal 16).

COMARET - CICS COMMAREA Usage for Task Control

`COMARET=`*`value`* defines whether the Natural CICS Interface is to take advantage of the CICS command level COMMAREA facility when terminating and restarting pseudo- conversational tasks.

Value	Explanation
`ON`	A pseudo-conversational Natural task saves its restart information into a CICS COMMAREA, unless it has been invoked with `EXEC CICS LINK`. This is the default value.
`OFF`	Forces Natural to place its restart information into CICS main temporary storage, which results in more overhead because of additional CICS service calls necessary to place and retrieve this information. The CICS temporary storage key used consists of a prefix string (as defined with the `NCMDIR` parameter `TSKEY` (see the *TP Monitor Interfaces* documentation) and of the terminal ID. If running in a CICSplex environment, the CICS temporary storage key prefix must be defined in a CICS TST as `REMOTE/SHARED` to be accessible in all participating CICS regions.

DIRNAME - Name of Natural CICS Interface System Directory Module

`DIRNAME=`*`value`* specifies the name of the Natural CICS Interface system directory module.

Value	Explanation
(see below)	Any valid module name.
`prefix``CB`	*`prefix`* is the common prefix for programs and files, see `PREFIX` parameter. This is the default value.

The first 5 characters of the directory module name are also used as part of CICS temporary storage queue names related to the relevant NCI environment. So when running more than one Natural CICS environment in a CICS region, the relevant system directory module names must be different in the first 5 characters.

Note that the specified or defaulted Natural CICS Interface system directory module name may be modified at run-time via the NCI system directory module name exit interface `NCIDIREX` (see the *TP Monitor Interfaces* documentation). This makes it possible to use the same NCI driver/Natural parameter module, but use different NCI environments (thread groups/thread sizes, etc.) depending for example on CICS system ID, transaction ID.

DUPTID - Handle Duplicate Terminal ID

The Natural CICS Interface requires unique terminal IDs, because the terminal ID is the key for its session information records (SIRs). This is normally guaranteed for a single CICS region, but not necessarily over several CICS regions sharing the same SIP server.

`DUPTID=`*`value`* determines how the Natural CICS Interface has to deal with duplicate terminal IDs, that is, when a new session is to be started and an SIR already exists for this terminal ID.

Value	Explanation
`ON`	If a duplicate terminal ID is encountered, the Natural CICS Interface internally forces the old session to terminate and, after that, starts a new session. This is the default value.
`OFF`	When an SIR already exists for the new session's terminal ID, the Natural CICS Interface terminates the new session and issues the message `NS19`. For an explanation and remedial actions, see *Natural under CICS Messages*, *SCP Processing Errors* in the *Messages and Codes* documentation.

A terminal ID exit interface is available to create unique 8-character terminal IDs, for example, by appending the 4-character CICS system ID to the physical 4-character CICS terminal ID, which results in a logical Natural terminal ID.

FDTPX - Force Use of NCIDTPEX Exit for all Terminal Types

`FDTPX=`*`value`* determines whether the `NCIDTPEX` terminal I/O exit interface (see the *TP Monitor Interfaces* documentation) is called for all types of terminal used in your environment.

Value	Explanation
`ON`	The `NCIDTPEX` interface is called for all terminal types.
`OFF`	The `NCIDTPEX` interface is only called for distributed transaction processing (DTP) using APPC or MRO conversions. This is the default value.

LOGDEST - Natural CICS Logging Destination

`LOGDEST=`*`value`* specifies the name of a CICS destination, where the Natural CICS Interface writes its session log records to.

Value	Explanation
`name`	Any valid destination name.
`NLOG`	This is the default value.

A CICS destination control table entry must be defined for the optional Natural CICS log data set.

MEMOBJR - Roll To Memory Object When Possible

In later versions of CICS TS, main temporary storage queues are allocated above the bar. When using CICS main temporary storage, roll data is split into temporary storage queue items with a maximum size of 32 KB each.

Depending on the CICS TS version installed at your site, CICS also supports the usage of memory objects for CICS applications. Allocating CICS memory objects for roll data means less overhead since the roll data must not be split.

Possible values for `MEMOBJR=`*`value`* are:

Value	Explanation
`ON`	Data is rolled by using CICS memory objects if supported by the CICS version installed at your site. This is the default value.
`OFF`	Data is rolled into CICS main temporary storage.

MSGDEST - Destination ID for Natural Error Message Logging

`MSGDEST=`*`value`* specifies the name of the CICS destination to be used by the Natural CICS Interface for NCI informational messages and to log the Natural session termination message if a session terminates abnormally.

Value	Explanation
`name`	Any valid destination name
`NERR`	This is the default value.

Since these messages are in character format, any already available CICS destination (for example, `CSSL`) can be used rather than defining a new one.

MSGPFX - Generate NCI Message Prefix for WTL Messages

The Natural CICS Interface uses a prefix for all messages it sends to the `MSGDEST` destination. This prefix has a length of approximately 48 bytes and comprises the following information:

- NCI message number,
- CICS region `SYSID`,
- terminal ID or the string `ASYN` for non-terminal tasks,
- user ID,
- transaction ID,
- date and time.

By default, the message prefix is also appended to those messages which are output through `CMWTL`.

Possible values for `MSGPFX=`*`value`* are:

Value	Explanation
`ON`	The `NCI` message prefix is appended to all messages which are issued through `CMWTL`. This is the default value.
`OFF`	The `NCI` message prefix is not appended to the messages which are issued through `CMWTL`. The messages are issued unchanged.

MSGTRAN - Internal Message Switching Transaction ID

`MSGTRAN=`*`value`* specifies the transaction ID internally used by the Natural message switching and asynchronous session flushing facilities.

This parameter has the same meaning as the `MSGTRAN` parameter in the `NCIZNEP` module (see the Natural *Installation* documentation) and must be specified identically.

Value	Explanation
(see below)	Any valid CICS transaction ID.
`NMSG`	This is the default value.

This transaction ID must be different from any transaction ID used to invoke Natural, and it must be defined in CICS.

PREFIX - Common Prefix for Programs and Files

`PREFIX=`*`value`* defines a common module prefix for the Natural CICS components as the Natural CICS system directory *`prefix`*`CB`, the CICS 3270 Bridge `XFAINTU` exit *`prefix`*`XFA`, the VSAM roll files *`prefix`*`R`*`n`*, where *`n`*=1 - 9, and system control records in CICS main temporary storage holding information about all permanent GETMAIN storages by the Natural CICS Interface as local pools and shared threads. The TS control record keys are of the form *`prefix`**`X`*`CR`, where *`X`* is an unprintable character. In general, it is good practice to use this common prefix for all programs that relate to the Natural CICS Interface, for example, *`prefix`*`DRV` for the Natural CICS Interface module, *`prefix`*`NEP` for the Natural CICS Interface node error program.

Value	Explanation
`XXXXX`	The *`prefix`* can be 1 to 5 bytes long and must conform to the naming conventions for programs and files.

No default value is provided.

PRMDEST - Name of the Natural CICS Profile Parameter Input Destination

`PRMDEST=value` specifies the name of a CICS destination containing Natural dynamic profile parameters.

Value	Explanation
`name`	Any valid destination name
`NPRM`	This is the default value.

At system initialization time, the Natural CICS Interface retrieves Natural dynamic profile parameters and saves them in its environment. At session start, potential other profile parameters (entered by way of terminal input or by a front-end caller) are concatenated at the end of the parameter string which was retrieved from the `PRMDEST` destination, that is, explicit dynamic profile parameters can be used to overwrite the Natural CICS Interface system profile parameters read from `PRMDEST`.

A CICS destination control table entry must be defined for the optional Natural CICS Interface profile parameter input destination, normally an extra partition destination.

PSTRNID - Control of *INIT-PROGRAM Variable Setting

When a Natural task is activated by a front-end program, the `PSTRNID` parameter determines, how the Natural system variable `*INIT-PROGRAM` is set.

Possible values for `PSTRNID=value` are:

Value	Explanation
`ON`	`*INIT-PROGRAM` is set to the actual transaction ID used for Natural CICS pseudo-conversational task processing, which is not necessarily the transaction ID of the task which originally started the Natural session. This is the default value.
`OFF`	`*INIT-PROGRAM` is set to the transaction ID of the task, which originally started the Natural session.

RCVASYN - Recover Asynchronous Session

`RCVASYN=value` defines how the Natural CICS Interface treats asynchronous sessions.

Value	Explanation
`ON`	This is the default value. The Natural CICS Interface forces some Natural profile parameter settings for non-terminal sessions to prevent unexpected input or abends due to NT06, NT11 or other I/O errors. `RCVASYN=ON` forces the following parameter settings: ■ `CM=OFF,MENU=OFF,PC=OFF` ■ `TTYPE=ASYL` if the `SENDER` specification is blank, not specified or a CICS transient data queue, or if `CONSOLE` is specified (see *Asynchronous Natural Sessions under CICS* in the *TP Monitor Interfaces* documentation). ■ `SENDER='`*`msgdest`*`'` if the `SENDER` specification is blank or not specified. ■ `OUTDEST='`*`sender`*`'` if the `OUTDEST` specification is blank or not specified. ■ `INTENS=1,EJ=OFF` if the `SENDER` specification is `CONSOLE` or a CICS transient data queue which is not set up for print control characters.
`OFF`	The Natural CICS Interface does not do anything specific for non-terminal sessions; it is the user's responsibility to set appropriate Natural profile parameters for an asynchronous Natural session; see *Asynchronous Natural Processing*.

RESENDC - Check for Screen Re-sending after Subroutine Calls

Natural optimizes the 3270 output data stream by default. The screen imaging technique used by Natural makes it possible for Natural to always remember the map most recently sent. Thus, when sending a new map, Natural actually sends “updates” of the old map only. With this logic, a screen image can get destroyed by 3GL programs called by Natural which perform screen I/O themselves.

Possible values for `RESENDC=`*`value`* are:

Value	Explanation
`ON`	The Natural CICS Interface checks whether any called 3GL programs have performed screen I/O. If so, the Natural CICS Interface causes Natural to send a full screen with the next screen I/O. This is the default value.
`OFF`	The Natural CICS Interface causes Natural to send only updates.

RESENDS - Screen Re-send Check after Pseudo-Conversational Session Resume

Natural optimizes the 3270 output data stream by default. The screen imaging technique used by Natural makes it possible that Natural always remembers the map most recently sent. Thus, Natural only sends “updates” when sending a new map, too. With this logic a screen image can get destroyed, for example, by message switching (CICS CMSG transaction) during pseudo-conversational screen I/O.

Possible values for `RESENDS=`*`value`* are:

Value	Explanation
`ON`	During the Natural session, the Natural CICS Interface also recognizes screen I/O from outside and causes Natural to re-send the screen most recently issued. This is the default value.
`OFF`	Natural only sends “updates” when sending a new map.

RJEDEST - Name of the Natural CICS Submit Destination

`RJEDEST=`*`value`* applies to z/OS-type operating systems only.

Value	Explanation
(see below)	Destination name.
`NRJE`	This is the default value.

`RJEDEST` specifies the *destination name* of the CICS extra partition destination used by the `NATRJE` utility for submitting jobs via the JES internal reader facility.

Caution: An appropriate CICS destination must be defined in the CICS DCT and start-up JCL; see also the corresponding step in *Installing Natural CICS Interface on z/OS* in the Natural *Installation* documentation.
Function code `L` or `B` (*`parm3`* of the `NATRJE CALL` statement) must be set for the last `NATRJE` call.
When `L` is specified and *`nrje`* is an extra partition destination, the destination is closed, which in turn triggers the start of the internal reader.
When `B` is specified and *`nrje`* is an indirect destination, the destination is not closed; in this case, a trailing `/*EOF` card must be submitted in order to trigger the start of the internal reader.

For further information on the Natural `NATRJE` utility, refer to the Natural *Utilities* documentation.

SLCALL - Standard Linkage Call

The Natural `CALL` statement invokes a dynamic non-Natural program using CICS conventions, that is, via an `EXEC CICS LINK`. A dynamic non-Natural program can also be invoked with standard linkage conventions (for example, `BALR/BASR/BASSM 14,15`) if an appropriate indicator is set in the Natural program before the `CALL` statement is executed; see also the terminal command `%P=S`, `%P=SC`, `%P=L` and `%P=LS`.

Caution: The terminal commands `%P=S`, `%P=SC`, `%P=L` and `%P=LS` bypass the `SLCALL` automatism of using a certain linkage convention.

`SLCALL=`*`value`* enables you to automatically use a certain linkage convention.

Value	Explanation
`ON`	The Natural CICS Interface determines whether the module to be called is a valid CICS command level program by looking for the string `DFH` at the module's load point. If `DFH` is found, the program is invoked via an `EXEC CICS LINK`. If `DFH` is not found, the module is treated according to standard linkage conventions and is invoked via `BALR/BASSM 14,15`.
`OFF`	The linkage convention is not used. This is the default value.

SLNOHLD - Load Option for External Programs to Be Invoked via Standard Linkage Conventions

`SLNOHLD=`*`value`* defines how the Natural CICS Interface treats non-LE external programs to be invoked via standard linkage conventions (that is, dynamic non-CICS programs and RCA programs) in a non-CICSPlex environment.

Value	Explanation
`ON`	This is the default value. The Natural CICS Interface loads all non-LE external programs to be invoked via standard linkage conventions (including RCA programs) via `EXEC CICS LOAD` without the `HOLD` option, thus allowing these programs to be `NEWCOPY`ed while the Natural session is suspended/waiting in a pseudo-conversational screen I/O. `SLNOHLD=ON` corresponds to the processing which the Natural CICS Interface does for LE programs in general and for non-LE programs in a CICSPlex environment anyhow.
`OFF`	This is how Natural worked eversince. The Natural CICS Interface loads all non-LE external programs to be invoked via standard linkage conventions (including RCA programs) via `EXEC CICS LOAD HOLD`, that is, such a program is fixed in storage for some time depending on the `DELETE` profile parameter setting, RCA programs until session end.

SNDLAST - LAST Option Usage for EXEC CICS SEND Commands

`SNDLAST=`*`value`* is useful for SNA terminals (`LUTYPE2`) with bracket protocol to force “end bracket” for pseudo-conversational screen I/O.

Value	Explanation
`ON`	The `LAST` option is used for `EXEC CICS SEND` commands before the task terminates in pseudo-conversational mode. This is the default value.
`OFF`	The `LAST` option is not used.

STORVIO - Storage Violation Trap

`STORVIO=`*`value`* provides for a storage violation trap for external program calls with call option `%P=C(C)`.

<table>
<tr><th>Value</th><th colspan="2">Explanation</th></tr>
<tr><td>`OFF`</td><td colspan="2">The storage violation trap is deactivated.
This is the default value.</td></tr>
<tr><td rowspan="4">(*mm*,*nn*)</td><td colspan="2">The storage violation trap is activated by specifying any `STORVIO` sub-parameter.
The first sub-parameter specifies a tolerance value in the range from 0 to 16777215: the storage size for the extra %C(C) GETMAIN is increased by this value to try to prevent real CICS storage violations.
The second sub-parameter specifies, how to react on a detected storage violation. Possible values:</td></tr>
<tr><td>`0`</td><td>Just a NCI0250 storage violation message is issued, no other special interaction.
This is the default value for sub-parameters.</td></tr>
<tr><td>`1 - 32767`</td><td>In addition to the NCI0250 message, a NAT0920 condition is raised with the specified value passed as reason code; as in the CICS world the NAT0920 reason code normally holds the EIBRESP value of a failing `EXEC CICS LOAD` or `LINK` request, it is recommended *not* to specify a value in the range of valid CICS EIBRESP values, that is, better leave values `1` to `255` to CICS.</td></tr>
<tr><td>`32768` or higher</td><td>In addition to the NCI0250 message, an S0C3 abend is forced, which raises a NAT0954 condition.</td></tr>
</table>

TERMVAR - Terminal ID Variable for Natural Work Files

`TERMVAR=`*`value`* enables a Natural user to have exclusive Natural work files under CICS without having to know the terminal ID.

Value	Explanation
`xxxx`	Variable *`xxxx`* is a four-character string. See explanation below.
`&TID`	This is the default value.

As terminal IDs are unique in a CICS system, exclusive work files in CICS temporary storage usually contain the CICS terminal ID. The parameter `TERMVAR` allows you to define a variable. If this variable is found in a work file name, it will be replaced by the actual terminal ID. Strings with non-alphanumeric characters must be enclosed in apostrophes (').

Note that for non-terminal sessions the packed CICS task number is used as a *logical* terminal ID.

Caution: The variable string must not contain the substring `'**'`, because Natural will replace this substring with the work file number, which makes it impossible to insert the terminal ID.

TIOBSZ - Size of the Natural Terminal I/O Buffer

`TIOBSZ=`*`value`* specifies the size of the Natural terminal I/O buffer.

Value	Explanation
`8 - 60`	Size of the terminal I/O buffer in KB.
`16`	This is the default value.

TRANCHK - Check Input Map for Transaction ID

If a connection to a CICS session gets lost or dropped (for example under VM or, when a session manager is installed) without having terminated the session, another user can get into this open session when calling CICS. Usually, the first action of a user in a CICS environment is to enter a transaction ID.

Possible values for `TRANCHK=`*`value`* are:

Value	Explanation
`ON`	The Natural CICS Interface checks whether the first 4 bytes of the transaction ID entered by the user matches the Natural transaction ID. If so, the Natural CICS Interface assumes a “restart” after a connection has been lost or dropped. All resources of the “old” session are freed and a new session is started.
`OFF`	Data entered by the user are not checked for the Natural transaction ID. This is the default value.

TTYCNSL - Control Console Communication

This parameter is for compatibility with previous versions of the Natural CICS Interface. `TTYCNSL=`*`value`* controls session and device characteristics for Natural sessions started through a console device by using, for example, the `MODIFY` command.

Value	Explanation
`ON`	The `*DEVICE` system variable is set to `TTY`: communication with the console is in 3270 data stream holding TTY control orders. The `PSEUDO` profile parameter is evaluated allowing or disallowing the session to run in pseudo-conversational mode.
`OFF`	The `*DEVICE` system variable is set to `BATCH` forcing batch/command-line mode: each line is output separately to the console by an `EXEC CICS WRITE OPERATOR` command. The `PSEUDO` profile parameter is ignored: the session runs in conversational mode to indicate that a session is pending. This is the default value.

UCTRAN - Lower/Mixed Case Support in Natural

`UCTRAN=`*`value`* enables or disables the lower/mixed case support by the Natural CICS Interface.

Value	Explanation
`ON`	Same as `UCTRAN=(ON,ON)`. NCI lower/mixed case support is fully enabled. This is the default value.
`OFF`	Same as `UCTRAN=(OFF,ON)`. NCI lower/mixed case support is disabled for pseudo-conversational screen I/O.

The first subparameter controls NCI mixed case support after a pseudo-conversational screen I/O, while the second subparameter controls NCI mixed case support after a conversational screen I/O; the latter also includes NTC uploads.

First Subparameter (pseudo-conversational screen I/O)

To accomplish lower/mixed case support for pseudo-conversational Natural sessions, it is necessary that the terminal input be not already translated to upper case before the Natural nucleus gets control. Therefore, the Natural CICS Interface by default switches terminals defined with `UCTRAN(ON)` into mixed mode `(UCTRAN(TRANID))` for the lifetime of the Natural session.

As for security reasons any modification of CICS definitions/control blocks may not be desired, the Natural CICS Interface can be prevented from modifying a terminal's upper case translation status by setting this `NTCICSP` parameter `UCTRAN` to `OFF`. If so, the user must define a terminal as running in “lower case” (CICS `TYPETERM` parameter `UCTRAN(TRANID/OFF)`) to be able to use the Natural lower/mixed case support.

As all CICS versions supported by the current Natural Version provide “case switching” on transaction level via the `UCTRAN` parameter in a transaction's `PROFILE`, this `NTCICSP` parameter should be set to `OFF`, thus leaving lower/mixed case support to CICS.

Note: In CICS, the combination of the `UCTRAN` parameters in both `TYPETERM` and `PROFILE` definitions determine how CICS treats the terminal input of a pseudo-conversational transaction (for details see CICS Resource Definition Manual or others). Therefor it is always advisable that mainly the `PROFILE` associated to a transaction defines the required upper case translation status thus making an application unaffected by any `TYPETERM` upper case translation mode changes.

Second Subparameter (conversational screen I/O)

Lower/mixed case support for conversational I/O means that the Natural CICS Interface uses the “as is” option on the CICS terminal input requests (`CONVERSE/RECEIVE ASIS`). If the second subparameter is set to `OFF`, the Natural CICS Interface does the conversational CICS terminal input requests without the “as is” option,

UNITID - Establish Unique Terminal IDs

`UNITID=`*`value`* helps make the CICS terminal ID for Natural purposes unique over more than one CICS region.

Value	Explanation
`ON`	The Natural CICS Interface appends a CICS system ID (local `SYSID` if no MRO, otherwise `TOR SYSID`) to the 4-byte CICS terminal ID, thus creating an 8-byte logical terminal ID.
`OFF`	The Natural CICS Interface uses the CICS terminal ID as it is. This is the default value.

This parameter is of interest when resources are shared as SIP server or roll server by several CICS regions, particularly in non-CICSplex: If the same terminal IDs are used in several CICS environments, this parameter helps to provide unique terminal IDs for Natural. Inside the Natural CICS

Interface, Natural terminal IDs are 8-byte fields, and a combination of 8-byte terminal ID and 8-byte CICS user ID is taken as key for SIP and the roll server.

The result of this parameter is used by the Natural CICS Interface for the session key and the roll server key and by Natural for the system variable `*INIT-ID`.

Notes:

1. A terminal ID exit (`NCITIDEX`) possibly will post-process that logical terminal ID. (`NCITIDEX` is described in the *TP Monitor Interfaces* documentation.)
2. Also a user ID exit (`NCIUIDEX` and `NATUEX1`) may post-process the `*INIT-ID` system variable. (`NCIUIDEX` is described in the *TP Monitor Interfaces* documentation, `NATUEX1` in the *Operations* documentation.)
3. This parameter also applies to *Natural Advanced Facilities* (NAF) printers, that is, the printers have to be defined appropriately in the NAF spooling and report management system `NATSPOOL`, or a user ID exit should be used to post-process the `*INIT-ID` for printers.
4. For non-terminal sessions, the Natural CICS Interface always sets up an 8-byte logical *terminal ID* consisting of the packed CICS task number and the CICS system ID; that is, `UNITID=ON` is forced for asynchronous tasks with the CICS task number taken as terminal ID.

USERID - Deal with CICS User ID

`USERID=`*`value`* defines how Natural under CICS should deal with a CICS user ID for a Natural session.

The first subparameter is for terminal bound CICS sessions, the second subparameter for non-terminal, that is, asynchronous, DPLEd, etc. CICS sessions, the third subparameter is for program-to-program sessions, that is, DTP, APPC.

Value	Explanation
`ANY`	Any non-blank value returned by `EXEC CICS ASSIGN USERID (..)` is considered to be valid. This is the default value.
`ON`	A non-blank value returned by `EXEC CICS ASSIGN USERID (..)` is considered to be valid if it is different from the CICS default user ID and, for terminal bound sessions only, if the user has signed on in CICS.
`OFF`	The value returned by `EXEC CICS ASSIGN USERID (..)` is ignored.

Further Processing

When a CICS user ID is invalid or ignored, the edited (unpacked) CICS task number is taken for non-terminal, that is, asynchronous or DPLed, etc., CICS sessions; for terminal bound sessions, the 3-byte CICS operator ID is taken when it is non-blank, otherwise the CICS terminal ID is taken; for DTP sessions the pseudo terminal ID is taken.

Notes:

1. CICS terminal IDs are unique within a CICS region, while CICS user IDs and operator IDs are not necessarily unique. However, CICS terminal IDs may have duplicates in other CICS regions resulting in duplicate user IDs in Adabas.
2. Natural user ID exit `NATUEX1` (see the *Operations* documentation) or Natural CICS user ID exit interface `NCIUIDEX` (see the *TP Monitor Interfaces* documentation) may be used to customize the content of the system variable `*INIT-USER`.

Example of NTCICSP Macro

```
         NTCICSP PREFIX=NCI83,                                          *
               DIRNAME=NCI83,                                           *
               BACKRPL=COMA,                                            *
               CALLRPL=COMA
```

41 CLEAR - Processing of CLEAR Key in NEXT Mode

This Natural profile parameter causes Natural to execute a specific Natural terminal command whenever CLEAR is pressed during program execution in `NEXT` mode.

Possible settings	any character	The default action can be overridden by supplying a character which, when appended to the terminal-command control character (as specified with the `CF` parameter), forms a valid Natural terminal command.
Default setting	`%`	By default, when the CLEAR key is pressed, Natural responds as if the user had entered the terminal command `%%`.
Dynamic specification	yes	
Specification within session	no	

Note: Under Natural Security:, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.

Example:

```
CF=%
CLEAR=R
```

Natural executes the terminal command `%R` when the CLEAR key is pressed in `NEXT` mode.

42 CM - Command Mode

This Natural profile parameter can be used to suppress Natural command mode (`NEXT` and `MORE`).

Possible settings	`ON`	`NEXT` and `MORE` are available for command input.
	`OFF`	The Natural session will be terminated whenever `NEXT` is encountered; the `MORE` line will be write-protected (no input possible).
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

43 CMPO - Compilation Options

This Natural profile parameter and its keyword subparameters can be used at session start to specify dynamically or to override the same compiler options which you can specify statically with the `NTCMPO` macro in the Natural parameter module or, during an active session, with the system command `COMPOPT`.

Possible settings	See *CMPO Parameter Syntax*.	
Default setting	Identical to the corresponding `COMPOPT` options. See *Keyword Subparameters* or `COMPOPT` options.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTCMPO` is used instead.
Specification within session	yes	Use system command `COMPOPT`.

The following topics are covered below:

CMPO Parameter Syntax

The parameter syntax of `CMPO` is as follows:

CMPO=(*keyword-subparameter=value*,*keyword-subparameter=value*,...)

Notes:

1. The **keyword subparameters** are functionally identical to the compiler options that can be specified within the session, using the system command `COMPOPT`.
2. Each keyword subparameter can take the value `ON` or `OFF`, except for `GFID`, which can also take the value `VID`.
3. For further information such as default values and functional descriptions, refer to system command `COMPOPT` in the *System Commands* documentation.

NTCMPO Macro Syntax

The syntax of the `NTCMPO` macro in the Natural parameter module is as follows:

```
NTCMPO CHKRULE=value,                                      *
       CPAGE=value,                                        *
       DB2ARRY=value,                                      *
       DB2BIN=value,                                       *
       DB2PKYU=value,                                      *
       DB2TSTI=value,                                      *
       DBSHORT=value,                                      *
       ECHECK=value,                                       *
       GDASC=value,                                        *
       GFID=value,                                         *
       KCHECK=value,                                       *
       LOWSRCE=value,                                      *
       LUWCOMP=value,                                      *
       MASKCME=value,                                      *
       MAXPREC=value,                                      *
       MEMOPT=value,                                       *
       NMOVE22=value,                                      *
       PCHECK=value,                                       *
       PSIGNF=value,                                       *
       THSEP=value,                                        *
       TQMARK=value,                                       *
       TSENABL=value
```

Notes:

1. The `DB2PKYU` option is only available if it is supported by the Natural for DB version installed at your site.
2. The keyword subparameters are functionally identical to the compiler options that can be specified within the session, using the system command `COMPOPT`.
3. Each keyword subparameter can take the value `ON` or `OFF`, except for `GFID`, which can also have the value `VID`.
4. In the `NTCMPO` macro, the keyword subparameters can be specified in any sequence.
5. For further information such as default values and functional descriptions, refer to system command `COMPOPT` in the *System Commands* documentation.

Keyword Subparameters

`CHKRULE` | `CPAGE` | `DB2ARRY` | `DB2BIN` | `DB2PKYU` | `DB2TSTI` | `DBSHORT` | `ECHECK` | `GDASC` | `GFID` | `KCHECK` | `LOWSRCE` | `LUWCOMP` | `MASKCME` | `MAXPREC` | `MEMOPT` | `NMOVE22` | `PCHECK` | `PSIGNF` | `THSEP` | `TQMARK` | `TSENABL`

For a complete description of the compiler options, refer to the system command `COMPOPT` in the *System Commands* documentation. The default values indicated there also apply to the corresponding keyword subparameters of `CMPO` and `NTCMPO`.

Example of CMPO Parameter

```
CMPO=(KCHECK=ON,PCHECK=ON)
```

Example of NTCMPO Macro

```
         NTCMPO KCHECK=ON,                                                    *
               PCHECK=ON
```

44 CMPR - General Default Compression Optimization Algorithm

This Natural profile parameter enables the Natural administrator to define the general default compression optimization in order to preserve main storage for the sessions which are currently processing and to improve the performance of Natural.

In addition, the type of storage compression optimization can be defined specifically for individual buffer types, using the parameter `CMPR` of the `NTBUFID` macro in the `NATCONFG` module. The setting of this macro parameter overrides the general default setting of profile parameter `CMPR`. For further information, see *Customization of Buffer Characteristics* in the *Operations* documentation.

Possible settings	`OPT0`	Compression without optimization
	`OPT1`	Compression with optimization of identical characters from the buffer used low end and high end.
	`OPT2`	Compression with optimization by tiles with identical characters. The tile size is 128 bytes.
	`(OPT2,`*`nnn`*`)`	Compression with optimization by tiles with identical characters. The tile size is a multiple of 128 bytes. *`nnn`* determines the tile size by multiplying with 128. Possible values: `1-255`. Example: `(OPT2,5)` yields a tile size of 640 bytes.
Default setting	`OPT2`	This is a synonym for `(OPT2,1)`.
Dynamic specification	yes	
Specification within session	no	

45 COMP - Parameters for Natural Com-plete/SMARTS Interface

This Natural profile parameter is used to specify the parameters for the Natural Com-plete/SMARTS Interface. It corresponds to the `NTCOMP` macro in the Natural parameter module.

Possible settings	See *COMP Parameter Syntax*.	
Default setting	See *Keyword Subparameters*.	
Dynamic specification	yes	The parameter `COMP` can only be specified dynamically. In the Natural parameter module, use the macro `NTCOMP`.
Specification within session	no	

Note: For information on the Natural Com-plete/SMARTS Interface, see *Natural under Com-plete/SMARTS* in the *TP Monitor Interfaces* documentation.

COMP Parameter Syntax

The `COMP` parameter is specified as follows:

```
COMP=(keyword-subparameter=value,keyword-subparameter=value,...)
```

See *Keyword Subparameters*.

NTCOMP Macro Syntax

The `NTCOMP` macro is specified as follows:

```
         NTCOMP EXIT=value,                                            *
               HCDTID=value,                                           *
               INITID=value,                                           *
               LC=value,                                               *
               LE370=value,                                            *
               MSGHDR=value,                                           *
               NTHSIZE=value,                                          *
               SERVER=value,                                           *
               SPIEA=value,                                            *
               THABOVE=value,                                          *
               TTYxxx=value,                                           *
               UCTRAN=value,                                           *
               U2PRINT=value
```

See *Keyword Subparameters*.

Keyword Subparameters

`EXIT` | `HCDTID` | `INITID` | `LC` | `LE370` | `MSGHDR` | `NTHSIZE` | `SERVER` | `SPIEA` | `THABOVE` | `TTYxxx` | `UCTRAN` | `U2PRINT`

EXIT - User Exit Module Name

`EXIT=value` defines a user exit module name which can be called during a session initialization before Natural is initialized.

Value:	Explanation:
1 - 8 characters, or ' ' (blank)	Name of user exit.
' ' (blank)	No user exit is used. This is the default value.

HCDTID - Initialization of Hardcopy Destination

`HCDTID=value` controls the initialization of the hardcopy destination.

Value:	Explanation:
`ON`	The hardcopy destination is initialized with the terminal ID.
`OFF`	The hardcopy destination corresponds to the logical terminal name. This is the default value.

INITID - Content of *INIT-ID

`INITID=value` controls the content of the system variable `*INIT-ID`.

Value:	Explanation:
`TIBNAM`	`*INIT-ID` contains the logical unit name of the user's terminal.
`TID`	`*INIT-ID` contains the string *lbnnnnnn*, where *l* is the stack level on which the session is running, *b* is blank and *nnnnnn* is the TID number, right justified without leading zeroes. This is the default value (Natural terminal ID).
`CPATCH`	`*INIT-ID` contains the same string as with `INITID=TID`, except that *b* is the Com-plete patch character instead of a blank.

LC - Enable Lower-Case Mode

`LC=`*`value`* can be used to switch the terminal between lower-case and upper-case mode.

Value:	Explanation:
`ON`	Lower-case mode. This is the default value.
`OFF`	Upper-case mode.

LE 370 - LE/370 Environment Usage

`LE370=`*`value`* specifies the usage of LE/370 as preinitialized environment (CEEPIPI interface) under Complete/SMARTS.

Value:	Explanation:
`ON`	All 3GL calls are handled in the preinitialized LE/370-enclave.
`OFF`	This is the default value.

MSGHDR - Activation of Message Header

`MSGHDR=`*`value`* activates or deactivates a message header for Natural error and termination messages using Com-plete's message switching facility for asynchronous Natural transactions.

Value:	Explanation:
`ON`	The message header is activated. This is the default value.
`OFF`	The message header is deactivated.

NTHSIZE - Natural Thread Size

`NTHSIZE=`*`value`* specifies the size of the storage area used for Natural's buffers, data areas and thread.

Value:	Explanation:
`256 - 2097151`	Size in KB. The actual upper limit is determined by the size of the Com-plete thread.
`1024`	This is the default value.

Note: This storage area is allocated within the physical Com-plete thread. The remaining area (Com-plete region size RG for the Natural transaction minus `NTHSIZE` value) is available for dynamically loading non-Natural subroutines, increasing of variable Natural thread buffers or for Natural work pools, for example.

SERVER - Name of Natural Server

`SERVER=value` defines the name of the Natural server which is initialized during Com-plete startup.

Value:	Explanation:
1 - 8 characters	Name of the Natural Server.
`NCFNAT82`	This is the default value.

Notes:

1. The specified server is used to maintain common storage and tables across Natural sessions, for example, local buffer pools. The server must be defined in the Com-plete startup.
2. It is possible to copy the supplied server module `NCFNAT82` under a different name and to link and run different Natural Com-plete interfaces with different servers, that is, with different sets of local buffer pools in the same Com-plete.

SPIEA - Activation of ABEXIT Exits

`SPIEA=value` activates or deactivates the `ABEXIT` exits.

Value:	Explanation:
`ON`	Activates the `ABEXIT` exit. This is the default value.
`OFF`	Deactivates the `ABEXIT` exit. Should be used for test purposes only.

THABOVE - Location of Natural Thread

`THABOVE=value` determines the location of the Natural thread (see `NTHSIZE` parameter).

Value:	Explanation:
`ON`	The Natural thread is allocated in the Com-plete thread extension above the 16 MB line. This is the default value (use Com-plete thread extension).
`OFF`	The Natural thread is allocated in the physical Com-plete thread below the 16 MB line

TTYxxx - TTY Device Control Characters

`TTYxxx=value` sets teletypewriter (TTY) device control characters. The following hexadecimal values can be set:

Value:	Explanation:
`TTYCR=0D`	TTY carriage return
`TTYLF=15`	TTY line feed
`TTYIC=00`	TTY idle character
`TTYNIC=00`	TTY number of idle characters
`TTYBS=16`	TTY backspace
`TTYAL=07`	TTY alarm

Note: There is no default value.

UCTRAN - Lower-Case to Upper-Case Translation of Com-plete/SMARTS Error Messages

`UCTRAN=value` controls the lower-case to upper-case translation of the Com-plete/SMARTS error messages.

Value:	Explanation:
`ON`	Upper-case translation enabled.
`OFF`	Upper-case translation disabled. This is the default value.

U2PRINT - Dynamic Printer Allocation

`U2PRINT=value` controls Com-plete's dynamic printer allocation feature for hardcopy requests.

Value:	Explanation:
`ON`	Natural calls for hardcopy requests Com-plete's `U2PRINT` routine to specify a printer destination.
`OFF`	Disables the dynamic hardcopy printer allocation. Natural uses the default value from Natural profile parameter `HCDEST`. This is the default value.

Example of COMP Parameter

```
COMP=(LE370=ON,INITID=TIBNAM,NTHSIZE=2000)
```

Example of NTCOMP Macro

```
         NTCOMP LE370=ON,                                                      *
               INITID=TIBNAM,                                                  *
               NTHSIZE=2000
```

46 CP - Default Code Page Name

This Natural profile parameter defines the default code page for Natural data and Natural sources.

<table>
<tr>
<td rowspan="2"><b>Possible settings</b></td>
<td>1 - 64 characters</td>
<td>The name of the desired code page.

<b>Note:</b>

1. Any character string is possible, but must be predefined by one of the code page parameters <code>CCSID</code>, <code>CCSN</code>, <code>IANA</code> or <code>ALIAS</code> of the macro <code>NTCPAGE</code> in the source module <code>NATCONFG</code>.
2. <code>UTF-32</code> is not allowed.
3. For information on multi-byte code page support, see <i>Support of Multi-Byte Code Pages</i> in the <i>Unicode and Code Page Support</i> documentation.</td>
</tr>
<tr>
<td><code>ON</code></td>
<td>Set the default code page for the mainframe as follows:

For BS2000, the code page is <code>EDF03IRV</code>.

For z/OS and z/VSE, it depends on the setting of Natural profile parameter <code>ULANG</code>:
<table>
<tr><td><code>ULANG</code> Setting:</td><td>Code Page Used:</td></tr>
<tr><td><code>ULANG=1</code> (English)</td><td><code>IBM01140</code></td></tr>
<tr><td><code>ULANG=2</code> (German)</td><td><code>IBM01141</code></td></tr>
<tr><td><code>ULANG=3</code> (French)</td><td><code>IBM01147</code></td></tr>
<tr><td><code>ULANG=4</code> (Spanish)</td><td><code>IBM01145</code></td></tr>
</table>
For other languages, <code>IBM01140</code> is used as default code page.

<b>Note:</b>

1. The language code related adaptation of the profile parameter <code>CP</code> applies only to the <code>ULANG</code> profile parameter active at session time.
2. Any subsequent language code modification(s) in Natural Security or by terminal command <code>%L=</code> do not influence the initial definition of the default code page.</td>
</tr>
</table>

	`OFF`	Disable code page support.
	`' '` (blank)	Same as `ON`.
	`AUTO`	The code page name from the user terminal is taken, if available. This applies to the following online environments only: TSO, CICS, Com-plete. **Note:** 1. For information on multi-byte code page support, see *Support of Multi-Byte Code Pages* in the *Unicode and Code Page Support* documentation. 2. `CP=AUTO` is not supported in a Natural Single Point of Development environment.
Default setting	`OFF`	Disable code page support.
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If no code page is specified for a code page sensitive operation such as data conversions to and from Unicode (for example, by means of a statement specific `ENCODED` option or by another profile parameter), the default code page applies.
2. For the current Natural session, it is assumed that all code page data, for example, Natural sources, contents of A-format fields, etc., are stored in this code page format.
3. See also *Profile Parameters and Macros* in the *Unicode and Code Page Support* documentation.
4. If the `CP` profile parameter is set to a value other than `OFF`, the value of the `CFICU` profile parameter will change to `ON`.
5. If the `CP` profile parameter is set to a value other than `OFF`, values specified with the profile parameters `TAB`, `UTAB1`, `UTAB2` and `SCTAB` during the start of a Natural session are ignored. See also *Translation Tables* in the *Unicode and Code Page Support* documentation.
6. If the profile parameter `CP` is set to a multi-byte code page (MBCS), the logical shift-in and shift-out characters will be supplied with the code page and therefore `SOSI` will be ignored.

Tips:

- You can find out the default code page that is the result of the evaluation of the `CP` parameter by viewing the content of the system variable `*CODEPAGE` (see system command `CPINFO`) or by using the *Unicode Properties* function of the SYSCP utility.

- You can use the `LIST DIRECTORY` system command or the SYSCP utility to find out the default code page used for encoding a Natural source object. The SYSCP utility can also be used to change the code page for a source object.

47 CPCVERR - Code Page Conversion Error

This Natural profile and session parameter specifies whether a conversion error that occurs when converting

- from Unicode to code page or
- from code page to Unicode or
- from one code page to another code page

results in a Natural error or not. Anyway, after the conversion, the target operand will contain the conversion result where all characters which cannot be converted will be replaced by a substitution character which is defined by ICU for the affected code page.

Notes:

1. This parameter is not regarded for the conversion of Natural sources when loading them into the source area or during catalog.
2. On mainframe platforms, it is not regarded whether a Unicode field is converted into the code page before an I/O on a terminal emulation. In this case, the substitution character is replaced by the placeholder character which is defined in `NATCONFG`.

Possible settings	`ON`	A Natural error NAT3413 is issued, if at least one code point could not be translated correctly during ICU conversion. For output statements, no error message is issued.
	`OFF`	No error is generated if one or more code points could not be translated correctly during ICU conversion.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	

Applicable command	`GLOBALS`	

See also:

- *Profile Parameters and Macros* in the *Unicode and Code Page Support* documentation.
- *Code Page Support for Editors, System Commands and Utilities on the Mainframe* in the *Unicode and Code Page Support* documentation.
- *Using an Error Transaction Program* in the *Programming Guide*

48 CPOBJIN - Code Page of Batch Input File

This Natural profile parameter specifies the code page of the batch input file `CMOBJIN` (see *Natural in Batch Mode*).

Possible settings	1 -64 characters	The name of the desired code page. **Note:** Any character string is possible, but must be predefined by one of the code page parameters `CCSID`, `CCSN`, `IANA` or `ALIAS` of the macro `NTCPAGE` in the source module `NATCONFG`.
	' ' (blank)	The code page resulting from the evaluation of the profile parameter `CP` is used.
Default setting	' ' (blank)	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If Natural code page support is disabled (for example, by parameter `CP=OFF`), any value specified for this parameter is ignored.
2. See also *Profile Parameters and Macros* in the *Unicode and Code Page Support* documentation.

49 CPPRINT - Code Page of Batch Output File

This Natural profile parameter specifies the code page of the batch output file `CMPRINT` (see *Natural in Batch Mode*).

Possible settings	1 - 64 characters	The name of the desired code page. **Note:** Any character string is possible, but must be predefined by one of the code page parameters `CCSID`, `CCSN`, `IANA` or `ALIAS` of the macro `NTCPAGE` in the source module `NATCONFG`.
	' ' (blank)	The code page resulting from the evaluation of the profile parameter `CP` is used.
Default setting	' ' (blank)	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If Natural code page support is disabled (for example, by parameter `CP=OFF`), any value specified for this parameter is ignored.
2. See also *Profile Parameters and Macros* in the *Unicode and Code Page Support* documentation.

50

CPSYNIN - Code Page of Batch Input File for Commands

This Natural profile parameter specifies the code page of the batch input file for commands `CMSYNIN` (see *Natural in Batch Mode*).

Possible settings	1 - 64 characters	The name of the desired code page. **Note:** 1. Any character string is possible, but must be predefined by one of the code page parameters `CCSID`, `CCSN`, `IANA` or `ALIAS` of the macro `NTCPAGE` in the source module `NATCONFG`. 2. UTF-32 is not allowed.
	' ' (blank)	The code page resulting from the evaluation of the profile parameter `CP` is used.
Default setting	' ' (blank)	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If Natural code page support is disabled (for example, by parameter `CP=OFF`), any value specified for this parameter is ignored.
2. See also *Profile Parameters and Macros* in the *Unicode and Code Page Support* documentation.

51 CSIZE - Size of Con-nect/Con-form Buffer Area

This Natural profile parameter determines the size of the Con-nect/Con-form buffer area.

Possible settings	`1-512`	Buffer size in KB.
	`0`	If `CSIZE=0` is specified or if the requested space is not available, Con-nect/Con-form cannot be used.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies if Con-nect/Con-form is installed.
2. Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the buffer size.
3. See the Con-nect/Con-form *Installation* documentation for further information.

52 CSTATIC - Statically Linked Modules

This Natural profile parameter can be used to define a list of module names which are to be linked statically together with the Natural parameter module. It corresponds to the `NTCSTAT` macro in the Natural parameter module.

Possible settings	See *CSTATIC Parameter Syntax*.	
Default setting	none	
Dynamic specification	no	
Specification within session	no	

Notes:

1. Each module specified and linked to the Natural parameter module can be called from a Natural object using a `CALL` statement.
2. As the length of a value of a profile parameter is limited to 256 characters, the number of module names specified with the `CSTATIC` parameter is limited. Alternatively, the macro `NTCSTAT` may be used to define more statically linked modules.
3. Modules which have been statically linked can be replaced during session initialization by loading them dynamically (see the profile parameter `RCA` for details). Modules which are linked neither statically nor loaded dynamically are loaded when they are first invoked by a `CALL` statement.
4. If you want to link modules to an environment-independent nucleus, you have to define them with the `CSTATIC` parameter in two Natural parameter modules: One parameter module has to be linked to the environment-independent nucleus and the other to the environment-dependent nucleus. Note that for all other parameter definitions only the parameter module linked to the environment-dependent nucleus is used.
5. For further information, see *Modules for Static Linking* in the *Installation* documentation.

The following topics are covered below:

CSTATIC Parameter Syntax

For each module name (1-8 characters) an external reference is generated for the linkage editor.

```
CSTATIC=module-name
```

Or, if the external reference (*`entry-name`*) is different from the module name, the entry name can be specified, enclosed in brackets, after the module name:

CSTATIC=*module-name(entry-name)*

NTCSTAT Macro Syntax

`NTCSTAT` allows just one module specification per macro call. For each module name (1-8 characters) an external reference is generated for the linkage editor.

```
         NTCSTAT module-name
```

Or, if the external reference (*entry-name*) is different from the module name, the entry name can be specified, separated by a comma, after the module name:

```
         NTCSTAT module-name,entry-name
```

Example of Parameter CSTATIC

```
CSTATIC=(MOD1,MOD7(ENTRY2),MOD12,MOD27($MAIN))
```

Examples of NTCSTAT Macro

```
         NTCSTAT MOD1
         NTCSTAT MOD7,ENTRY2
         NTCSTAT MOD12
         NTCSTAT MOD27,$MAIN
```

53 CV - Attribute Control Variable

This session parameter is used to reference an attribute control variable.

Possible settings	`B, C, D, I, N, U, V`	Field representation attributes (see session parameter `AD`).
	`P`	Field protection (see session parameter `AD`).
	`BL, GR, NE, PI, RE, TU, YE`	Color (for an explanation of the color codes, see the session parameter `CD`).
Default setting	none	

Applicable statements	`DISPLAY`
	`INPUT`
	`PRINT`
	`PROCESS PAGE`
	`REINPUT`
	`WRITE`
		Parameter may be specified at statement level and/or at element level.

Applicable command	none	

Notes:

1. An attribute control variable is defined with Format C (see *Special Formats* in the *Programming Guide*) and is used to assign field attributes dynamically and/or check the "modified" status of a field in conjunction with an `INPUT` or `PROCESS PAGE` statement; see also *Logical Condition Criteria, MODIFIED Option - Check whether Field Content has been Modified* in the *Programming Guide*.
2. By specifying the `MODIFIED` option of the `IF` statement, the attribute control variable can be used to check whether the contents of a field has been modified during the execution of an `INPUT` or `PROCESS PAGE` statement: `IF #ATTR MODIFIED ...`
3. A single attribute control variable can be applied to several input fields by specifying it once at statement level or multiple times at element level, in which case the "modified" status indication is set if any of the fields referencing the control variable has been modified. If the `CV` parameter is specified both at statement level and at field level and the attribute control variable

for the individual field is empty, the attribute control variable for the statement will be used for the field.

4. The attribute control variable can be expanded up to three dimensions, for example, `CONTR(*)`, `CONTR(*,*)`, `CONTR(*,*,*)`, depending on the rank of the corresponding array.

Example:

```
DEFINE DATA LOCAL
1 #ATTR(C)
1 #A   (N5)
END-DEFINE
...
MOVE (AD=I CD=RE) TO #ATTR
INPUT #A (CV=#ATTR)
...
```

54 CVMIN - Control Variable Modified at Input

This Natural profile parameter determines whether or not an attribute control variable is assigned the status `MODIFIED` when the setting of the field to which the attribute control variable is attached is overwritten by an *identical* setting.

Possible settings	`ON`	If a field setting is overwritten by the same setting, the corresponding control variable will be assigned the status `MODIFIED`.
	`OFF`	If a field setting is overwritten by the same setting, the corresponding control variable will *not* be assigned the status `MODIFIED`.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Note: If an attribute control variable has been assigned the status `MODIFIED`, the `MODIFIED` option evaluates this as `TRUE`. This applies regardless of whether the input was entered manually, read from the stack or supplied in batch mode.

55 DATSIZE - Minimum Size of Buffer for Local Data

This Natural profile parameter can be used to set the minimum size of the local data buffer (`DATSIZE`).

Possible settings	`10 - 2097151`	Minimum buffer size in KB.
Default setting	`32`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.
2. The `DATSIZE` buffer is a "variable size" buffer. If more storage for local data areas is required during the session, the `DATSIZE` buffer is expanded dynamically. In a thread environment, the `DATSIZE` may be temporarily allocated outside the storage thread if it becomes too large. The size of the `DATSIZE` buffer is reduced back to the minimum size when the application does not need the space any longer.

Function of the DATSIZE Buffer

At execution time, the `DATSIZE` buffer holds the local data used by the Natural main program being executed and the local data of all subordinate objects (except "FETCHed" programs) invoked by this program.

When you use Natural in a development environment, the minimum `DATSIZE` required is the default setting (that is, 32 KB). A smaller `DATSIZE` is only possible when using Natural as a runtime-only environment without any Natural utilities being available.

Calculating the DATSIZE Requirement

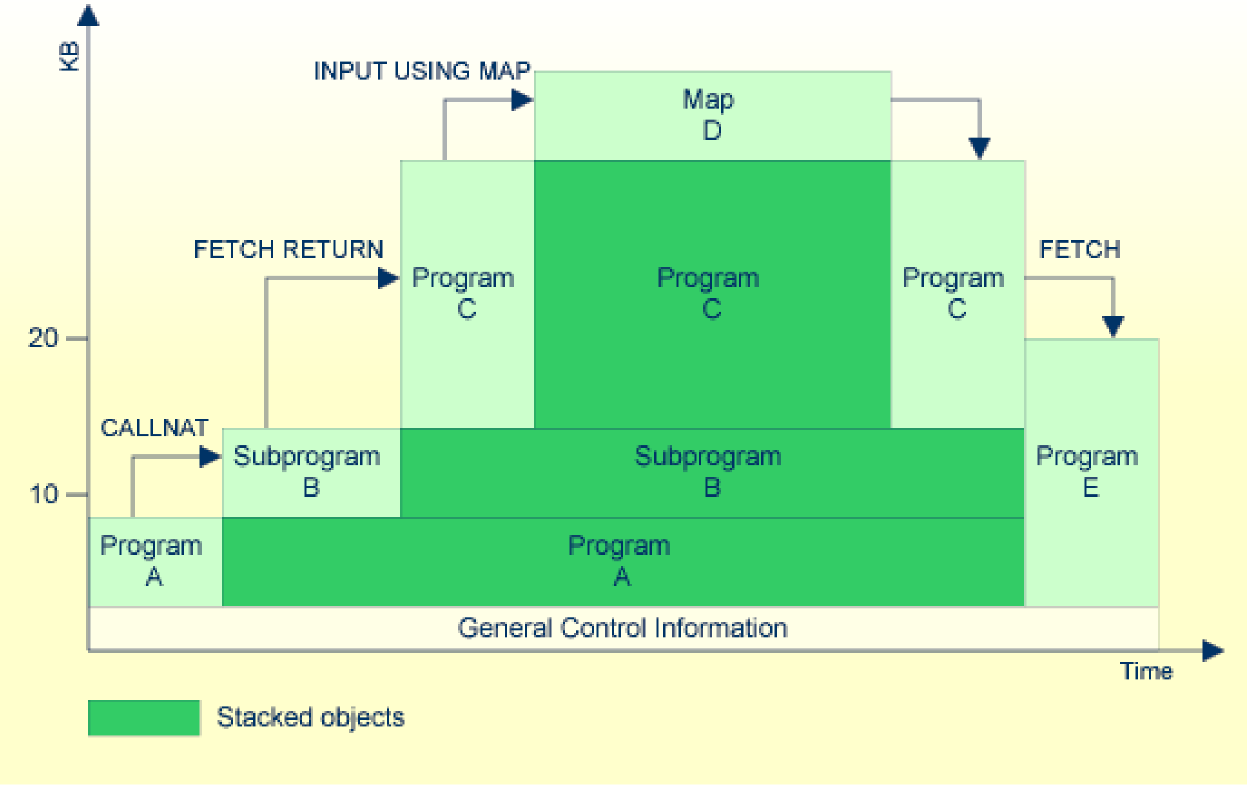
The actual `DATSIZE` requirement can be calculated as follows (refer to the illustration below):

If another object is invoked by the main program, the local data of this object are also held in the `DATSIZE` buffer.

If other objects are invoked from the invoked object (with a `CALLNAT`, `PERFORM`, `FETCH RETURN`, `INPUT USING MAP` statement, a helproutine/help map being invoked), their local data are also held in the `DATSIZE` buffer; the local data of an invoked object is held in the `DATSIZE` buffer until control is returned from the invoked object to the invoking object.

If another main program is invoked with a `FETCH` statement, the local data of all previously invoked objects are deleted from the `DATSIZE` buffer and the local data of the "FETCHed" program are held in the `DATSIZE`.

In addition, an amount of approximately 128 bytes of general control information for execution are held in the `DATSIZE` buffer, plus approximately 128 bytes of control information for each object whose local data are being held in the `DATSIZE` buffer. This is illustrated in the figure below.



The system command `LIST` provides an option to display directory information about an object. This information includes the object's `DATSIZE` storage requirement (not including the control information).

56 DB - Database Types and Options

This Natural profile parameter can be used to define database types and options for all and for specific database IDs. It corresponds to the `NTDB` macro in the Natural parameter module.

Possible settings	See *DB Parameter Syntax*.	
Default setting	`ADABAS,*`	The default database type is Adabas.
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTDB` is used instead.
Specification within session	no	

Notes:

1. For information on the Natural database management interface, see the *Database Management System Interfaces* documentation.
2. For the supported versions of the database management systems, refer to *Database Management Systems* in the current *Natural Release Notes for Mainframes*.
3. At compile time, Natural Data Manipulation Language (DML) statement functionality will be limited to the functionality that is available for the specified database type.
4. At runtime, the specified database type defines which Natural database management interface is called for a database ID.

The following topics are covered below:

DB Parameter Syntax

The DB parameter is specified as follows:

1. Default Database Definition

The default database type and its default options is specified as follows. It applies to all database IDs not explicitly specified by the `DB` parameter or `NTDB` macro. If there are no options, the commas and the asterisk can be omitted.

```
DB=(database-type,*,options)
```

2. Single Database Definition

A single database ID is specified as follows:

DB=(*database-type*,*database-ID*,*options*)

3. Multiple Database Definition

Multiple database IDs of the same database type with the same options can be specified together, enclosed in parentheses:

DB=(*database-type*,(*database-ID1*,*database-ID2*,...),*options*)

Where:

Syntax Element	Value	Explanation
database-type	See ***Possible Database Types***.	Database type. The default value is `ADABAS`. This subparameter is mandatory for the `NTDB` macro.
database-ID	`0 - 65535`	Database identification. Database ID `255` must not be specified, because it is reserved for internal use. You can specify a single database ID, a list of database IDs enclosed in parentheses, or an asterisk (*) to indicate the default for all databases not specified explicitly.
options		See ***Possible Database Options***.

NTDB Macro Syntax

The `NTDB` macro is specified as follows:

1. Default Database Definition

The default database type and its default options is specified as follows. It applies to all database IDs not explicitly specified by the DB parameter or `NTDB` macro. If there are no options, the commas and the asterisk can be omitted.

```
         NTDB database-type,*,options
```

2. Single Database Definition

A single database ID is specified as follows:

```
        NTDB database-type,database-ID,options
```

3. Multiple Database Definition

Multiple database IDs of the same database type with the same options can be specified together, enclosed in parentheses:

```
        NTDB database-type,(database-ID1,database-ID2,...),options
```

Possible Database Types

The database types that can be specified with the `DB` parameter or the `NTDB` macro are listed in the following table.

A database type and version for Adabas indicates the functional level of database features that are to be used by Natural.

All database types except Adabas require that the appropriate database handler module is installed. For more information, see the appropriate documentation about the database handler required in your environment.

Database Type	Adabas Features or Database Handler to be Used

`ADABAS`
or
`ADAV82`
	Adabas Version 8.2 features
	`ADABAS` is the default value.

`ADAV7`	Adabas Version 7 features
`ADAV8`	Adabas Version 8 features
`VSAM`	Natural for VSAM
`DLI`	Natural for DL/I

`DB2`
or
`SQL`
	Natural for DB2

`CXX`	Natural SQL Gateway
`PROCESS`	Entire System Server
`TRS`	Adabas Text Retrieval
`INCORE`	Natural ISPF

Possible Database Options

The following options can be specified with the `DB` parameter or the `NTDB` macro:

Option	Explanation
`ACODE`	The Natural application must communicate to Adabas whether code page or Unicode support is desired if the Adabas DBID used is enabled for character encoding and data conversion. Therefore, the `ACODE` setting specifying the application-specific code page for all A fields and/or the `WCODE=4095` (UTF-16) setting for all W fields must be sent with the OP call. See also *Unicode and Code Page Support*.
`WCODE`	
`ENTIRE`	The database is to be handled by Entire DB.
`OPEN`	This option applies to Adabas databases only, for which Adabas requires an open request to be issued. If `OPEN` is specified for such a database, an open request is always issued (even if the `ETID` is blank).
`READ`	The database is to be read-only.

The following options can be specified with the dynamic parameter `DB` only.

Option	Explanation
`NOENTIRE`	Resets the `ENTIRE` option.
`NOOPEN`	Resets the `OPEN` option.
`NOREAD`	Resets the `READ` option.
`OFF`	Removes any `DB` or `NTDB` definition for the specified databases, see ***Examples of DB Parameter*** below.

Examples of DB Parameter

`DB=(VSAM,(22,26,33))`	This defines Databases 22, 26 and 33 as VSAM databases.
`DB=(,*,READ)`	This sets all databases for which the default database definition applies to read-only.
`DB=(,(8,9),NOREAD)`	This removes the read-only option for Databases 8 and 9.
`DB=(,17,OFF)`	This resets the database definition of Database 17 to defaults.

Examples of NTDB Macro

`NTDB DLI,7`	This defines Database 7 as DL/I database.

57 DB2 - Parameters for SQL Database Management Interfaces

This Natural profile parameter is used to specify the parameters for the database management interfaces Natural for DB2 and Natural SQL Gateway. It corresponds to the `NTDB2` macro in the Natural parameter module.

Possible settings	See *DB2 Parameter Syntax*.	
Default setting	See *Keyword Subparameters*.	
Dynamic specification	yes	The parameter `DB2` can only be specified dynamically. In the Natural parameter module, use the macro `NTDB2`.
Specification within session	no	

The following topics are covered:

DB2 Parameter Syntax

The `DB2` parameter is specified as follows:

DB2=(*keyword-subparameter=value*,*keyword-subparameter=value*,...)

See *Keyword Subparameters*.

NTDB2 Macro Syntax

The `NTDB2` macro is specified as follows:

```
         NTDB2 BTIGN=value,                                         *
               CONVERS=value,                                       *
               CONVRS2=value,                                       *
               DB2COLL=value,                                       *
               DB2GROV=value,                                       *
               DB2PLAN=value,                                       *
               DB2SSID=value,                                       *
               DB2XID=value,                                        *
               DDFSERV=value,                                       *
               DELIMID=value,                                       *
               EBPFSRV=value,                                       *
               EBPMAX=value,                                        *
               EBPPRAL=value,                                       *
               EBPSEC=value,                                        *
               ETIGN=value,                                         *
               FSERV=value,                                         *
               MAXLOOP=value,                                       *
               MF=value,                                            *
               MAXSTMT=value,                                       *
               NNPSF=value,                                          *
               NSBDATE=value,                                        *
               NSBHOST=value,                                        *
               NSBPORT=value,                                        *
               PSCIGN=value,                                         *
               REFRESH=value,                                        *
               RETRYPO=value,                                        *
               RWRDONL=value,                                        *
               SMFSRV=value,                                         *
               STATDYN=value
```

See *Keyword Subparameters*.

Keyword Subparameters

There are two groups of keyword subparameters:

General Keyword Subparameters

`BTIGN` | `CONVERS` | `CONVRS2` | `DDFSERV` | `DELIMID` | `EBPFSRV` | `EBPPRAL` | `EBPSEC` | `EBPMAX` | `ETIGN` | `FSERV` | `MAXLOOP` | `MAXSTMT` | `MF` | `NNPSF` | `NSBDATE` | `NSBHOST` | `NSBPORT` | `PSCIGN` | `REFRESH` | `RETRYPO` | `RWRDONL` | `SMFSRV` | `STATDYN`

Special Keyword Subparameters

`DB2COLL` | `DB2GROV` | `DB2PLAN` | `DB2SSID` | `DB2XID`

Notes:

1. These special keyword subparameters belong together. They apply only to DB2 for z/OS in environments using CAF or RRSAF.
2. They provide DB2 connecting and resource functionality, which in earlier versions was only supplied by the `NATPLAN` program. These parameters apply only to DB2 for z/OS and to environments where either the DB2 Call Attachment Facility (CAF) or the DB2 Resource Recovery Services Attachment Facility (RRSAF) is used. An exception to this is the Natural for DB2 stored procedure environment, where DB2 already provides the DB2 resources based on the stored procedure creation parameter `COLLID`. Therefore, the keyword subparameters mentioned here are not used in a Natural for DB2 stored procedure environment. Before the very first DB2 SQL access is performed by Natural for DB2 in a CAF or RRSAF environment, Natural connects to the desired DB2 subsystem (`DB2SSID`), and the desired plan (`DB2PLAN`) is allocated.
3. If the application already connected to DB2 by `NATPLAN` before the first SQL request, the `NTDB2` or `DB2` parameters are ignored and the connection already established is used.

BTIGN - Ignore BACKOUT TRANSACTION Error

`BTIGN=`*`value`* enables you to ignore the error which results from a `BACKOUT TRANSACTION` statement that was issued too late for backing out the current transaction, because an implicit Syncpoint has previously been issued by the TP monitor.

Value	Explanation
`ON`	The error after a late `BACKOUT TRANSACTION` is ignored. This is the default value.
`OFF`	The error after a late `BACKOUT TRANSACTION` is *not* ignored.

Notes:

1. This parameter is relevant in CICS and IMS TM environments only.
2. This subparameter is ignored by Natural SQL Gateway.

CONVERS - Conversational Mode under CICS

`CONVERS=`*`value`* is used to allow conversational mode in CICS environments where no Natural file server is used.

Value	Explanation
`ON`	Conversational mode is allowed. This is the default value.
`OFF`	Conversational mode is *not* allowed.

Notes:

1. If this subparameter is set to `OFF` and no Natural file server is used, you cannot continue database loops across terminal I/O; if so, the following codes may occur.
2. With Natural for DB/2, DB2 SQLCODE -501, 504, 507, 514 or 518 applies.
3. With Natural SQL Gateway, Connex SQLCODE -4011 applies.
4. If, in a CICS environment, you are using the function *SQL Services (NDB)* or *SQL Services (NSB)* (described in the *Database Management System Interfaces* documentation) without Natural for DB2 file server, you must specify `CONVERS=ON`, otherwise the error mentioned above can occur.

CONVRS2 - Conversational Mode 2 under CICS

`CONVRS2=`*`value`* allows/disallows the conversational mode 2 in CICS environments.

Value	Explanation
`ON`	Conversational mode 2 is allowed.
`OFF`	Conversational mode 2 is *not* allowed. This is the default value.

Notes:

1. This subparameter is used to control conversational mode 2 in CICS environments. Conversational mode 2 means that update transactions are spawned across terminal I/O until either an explicit `COMMIT` or explicit `ROLLBACK` has been issued (Caution: DB2 and CICS resources are kept across terminal I/O.). This means `CONVRS2=ON` has the same effect as the Natural profile parameter `PSEUDO=OFF`, except that the conversational mode is entered after a DB2 update statement (`UPDATE`, `DELETE`, `INSERT`) and left again after a `COMMIT` or `ROLLBACK`, while `PSEUDO=OFF` causes conversational mode for the total Natural session.
2. See also `CALLNAT` subprogram `NDBCONV` (described in the *Database Management System Interfaces* documentation), which allows setting or resetting conversational mode 2 dynamically.

DB2COLL – DB2 Collection Name

`DB2COLL=`*`value`* specifies the collection name of DB2 packages used by the application in an environment where the RRSAF interface is used.

Value	Explanation
`value`	Any valid 18 character DB2 collection name.
' ' (blank)	No name is specified. This is the default value.

Notes:

1. This subparameter is only honored by Natural for z/OS.
2. This parameter is only honored by the RRSAF interface if the `DB2PLAN` character contains as first character a question mark.
3. See also ***Special Keyword Subparameters***.

DB2GROV – DB2 Group Override

`DB2GROV=value` specifies whether the connection to the DB2 system identified by `DB2SSID` is to be made to the single DB2 subsystem or to the DB2 sharing group, in case there exists a DB2 sharing group and a single DB2 with the identical `DB2SSID`.

Value	Explanation
' ' (blank)	Connection will be made to the DB2 sharing group identified by `DB2SSID`. This is the default value.
`NOGROUP`	Connection will be made to the DB2 subsystem identified by `DB2SSID`.

Notes:

1. This subparameter is only honored by Natural for z/OS.
2. See also ***Special Keyword Subparameters***.

DB2PLAN – DB2 Plan Name

`DB2PLAN=value` specifies the plan name used by the application.

Value	Explanation
value	Any valid 8 character DB2 plan name. If the first character is a question mark (?) and the RRSAF interface is used by the application, the packages identified by the collection name specified with the subparameter `DB2COLL` will be used by the application.
' ' (blank)	No name is specified. This is the default value.

Notes:

1. This subparameter is only honored by Natural for z/OS.
2. See also ***Special Keyword Subparameters***.

DB2SSID – DB2 Subsystem Identifier

`DB2SSID=`*`value`* specifies the name of the DB2 sharing group or the name of the DB2 subsystem to be connected to.

Value	Explanation
value	Any valid 4 character DB2 sharing group or DB2 subsystem name.
' ' (blank)	No name is specified. This is the default value.

Notes:

1. This subparameter is only honored by Natural for z/OS.
2. See also *Special Keyword Subparameters*.

DB2XID – DB2 Global Transaction ID

`DB2XID=`*`value`* specifies whether the RRSAF interface should use a global transaction ID or not.

Value	Explanation
`ON`	RRSAF will create a global transaction ID. This is the default value.
`OFF`	RRSAF will not create a global transaction ID.

Notes:

1. This subparameter is only honored by Natural for z/OS.
2. See also *Special Keyword Subparameters*.

DDFSERV - Alternate DD Name for Natural File Server

`DDFSERV=`*`value`* specifies either a DD name for the Natural file server (VSAM) or the name of the shared memory object used for a Shared Memory Objects File Server (`SMFSRV=ON`).

Value	Explanation
ddname	Any valid 8-character DD name or a shared memory object name.
`CMFSERV`	This is the default name.

DELIMID - Escape Character for Delimited Identifiers

`DELIMID=value` specifies the escape character to be used for generating delimited SQL identifiers for the column names and table names in SQL statements.

Value	Explanation
`DQ`	Double quotation mark (")
`SQ`	Single quotation mark (')
`OFF`	Delimited identifiers are not enabled. This is the default value.

Notes:

1. A delimited identifier is a sequence of one or more characters enclosed in escape characters. You must specify a delimited identifier if you use SQL-reserved words for column names and table names, as demonstrated in ***Example of DELIMID***.
2. To enable generation of delimited identifiers, `DELIMID` must be set to double quotation mark (`''`) or single quotation mark (`'`).
3. The escape character specified for `DELIMID` and the SQL `STRING DELIMITER` are mutually exclusive. This implies that the mark (double or single quotation) used to enclose alphanumeric strings in SQL statements must be different from the value specified for `DELIMID`.
4. If you enable delimited identifiers, ensure that the value specified for `DELIMID` also complies with the SQL string delimiter value of your DB2 installation.
5. See also the `RWRDONL` subparameter to determine which delimited identifiers are generated in the SQL string.

Example of DELIMID

In the following example, a double quotation mark (`''`) has been specified as the escape character for the delimited identifier:

Natural statement:

```
SELECT FUNCTION INTO #FUNCTION FROM XYZ-T1000
```

Generated SQL string:

```
SELECT "FUNCTION" FROM XYZ.T1000
```

EBPFSRV - Editor Buffer Pool for Natural File Server

`EBPFSRV=value` specifies whether the Natural file server uses the Software AG Editor buffer pool as the storage medium.

Value	Explanation
`ON`	The Software AG buffer pool is to be used as the storage medium for the Natural file server. `ON` *must* be set if the file server is to be used in a Parallel Sysplex environment. In this case, your Natural session must use the auxiliary editor buffer pool (see also *Support of a z/OS Parallel Sysplex Environment* in *Installing Software AG Editor*).
`OFF`	A VSAM file or a shared memory object (`SMFSRV=ON`) is to be used as the storage medium for the Natural file server (`SMFSRV=ON`). This is the default value.

EBPMAX - Editor Buffer Pool Maximum Allocation

`EBPMAX=value` specifies the maximum number of blocks to be allocated to each user of the Natural file server if the Software AG Editor buffer pool is used as the storage medium.

Value	Explanation
`0 - 32676`	Maximum number of blocks to be allocated.
`100`	This is the default value.

Notes:

1. This subparameter defines the upper limit for the allocation of buffer pool blocks to a single user.
2. If the `EBPFSRV` subparameter is set to `OFF`, `EBPMAX` is not used at runtime.

EBPPRAL - Editor Buffer Pool Primary Allocation

`EBPPRAL=value` specifies the number of blocks to be allocated primarily to each user of the Natural file server if the Software AG Editor buffer pool is used as the storage medium.

Value	Explanation
`0 - 32676`	Number of blocks to be allocated primarily.
`20`	This is the default value.

Notes:

1. If the `EBPFSRV` subparameter is set to `OFF`, `EBPPRAL` is not used at runtime.

EBPSEC - Editor Buffer Pool Secondary Allocation

`EBPSEC=`*`value`* specifies the number of blocks to be allocated secondarily to each user of the Natural file server if the Software AG Editor buffer pool is used as the storage medium.

Value	Explanation
`0 - 32676`	Number of blocks to be allocated secondarily.
`10`	This is the default value.

Notes:

1. The secondary allocation is used to allocate buffer pool blocks to the user if the primary allocation amount is already exhausted.
2. If the `EBPFSRV` subparameter is set to `OFF`, `EBPSEC` is not used at runtime.

ETIGN - Ignore END TRANSACTION Error

`ETIGN=`*`value`* is used to handle `END TRANSACTION` statements in a message-driven IMS region (MPP or message-oriented BMP).

Value	Explanation
`ON`	The `END TRANSACTION` error is ignored and processing is continued. This is the default value.
`OFF`	The `END TRANSACTION` error is *not* ignored.

Notes:

1. This subparameter is relevant in IMS MPP and message-oriented BMP environments only.
2. In such a region, an `END TRANSACTION` cannot be executed by the Natural IMS TM Interface and is therefore ignored without any notification. In such situations, the `ETIGN` subparameter can be used to issue an error message instead.
3. This subparameter is ignored by Natural SQL Gateway.

FSERV - Activate Natural File Server

`FSERV=`*`value`* specifies whether the Natural file server is to be used and whether it can be disabled in the case of an initialization error.

Value	Explanation
`ON`	Natural file server is to be used.
`OFF`	Natural file server is not to be used. This is the default value.
`DIS`	Natural file server is to be used but is to be disabled if it cannot be initialized.

Notes:

1. If `FSERV` is set to `ON` and the Natural file server is not operational, the initialization of Natural for DB2 is terminated with a corresponding Natural error message. The Natural interface to DB2 is disabled, and any SQL call is rejected with a corresponding error message.

MAXLOOP - Maximum Number of Nested Program Loops

`MAXLOOP=`*`value`* specifies the maximum possible number of nested SQL database access statements.

Value	Explanation
`1 - 99`	Maximum possible number of nested database access loops.
`10`	This is the default value.

MAXSTMT - Maximum Number of Allocated Dynamic SQL Statements

`MAXSTMT=`*`value`* specifies the maximum possible number of allocated dynamic SQL statements for the Natural SQL Gateway.

Value	Explanation
`1 - 99`	Maximum possible number of allocated dynamic SQL statements.
`10`	This is the default value.

Note: This subparameter is ignored by Natural for DB2.

MF - Multi-Fetch Row Count

`MF=`*`value`* specifies the number of rows to be fetched by DB2 in one `FETCH` operation. This subparameter can be used to enable multi-fetch operations by DB2 on a global basis. Changes to the application program are not required.

During static generation, `MF` also determines whether a generated `DECLARE CURSOR` statement contains the `WITH ROWSET POSITIONING` clause:

- If `MF` is set to zero (`0`), `DECLARE CURSOR` will not contain `WITH ROWSET POSITIONING`.
- If `MF` is set to a value greater than zero (`0`), `DECLARE CURSOR` will contain `WITH ROWSET POSITIONING`.

If a Natural for DB2 program already uses multi-fetch syntax in a `FIND`, `READ` or `SELECT` statement, this statement is executed as specified in the program and not affected by the `MF` subparameter. Irrespective of whether one of these statements already has a multi-fetch specification, the statement will use a multi-fetch buffer holding space for the number of rows specified in the `MF` subparameter.

`FIND`, `READ` and `SELECT` statements associated with a positioned `UPDATE` or `DELETE` do not use multi-fetch operations even if the `MF` subparameter is set to a value greater than zero (`0`).

If the `MF` subparameter is set to a value greater than zero (`0`) and one or more rows encounter a warning condition (for example, due to row-value truncation), DB2 can return SQLCODE +354. In this case, either set `PSCIGN=ON` to ignore the return code as positive SQLCODE, or increase the size of the receiving fields in the program so that the warning condition no longer occurs. In non-multi-fetch mode, the warning condition does not result in SQLCODE greater than zero (`0`).

Value	Explanation
`1 -32767`	The number of rows to be fetched by DB2 in one `FETCH` operation.
`0`	This is the default value.

Note: This subparameter is used by Natural for DB2 only.

NNPSF - Set Natural Numerics' Positive Sign to F

`NNPSF=`*`value`* is used to change the sign character of positive Natural variables which have format `N`, if they are filled from the SQL database system. Usually, these variables have the `C` as the positive sign character. If the subparameter `NNPSF` is set to `ON`, `F` is used as the positive sign character.

Value	Explanation
`ON`	Positive numbers put into Natural numeric variables by the SQL database system get the sign `F`.
`OFF`	Positive numbers put into Natural numeric variables by the SQL database system remain unchanged. This is the default value.

NSBDATE - Set Natural SQL Gateway Date String Format

`NSBDATE=`*`value`* specifies the format in which the Natural SQL Gateway server returns SQL date strings to the application. The application returns an SQL date as a string in the ISO format (`YYYY-MM-DD`) by default. `NSBDATE=E` enables the application to return SQL date strings in the EUR format (`DD.MM.YYYY`).

Value	Explanation
`' '`	Natural SQL Gateway returns SQL date strings in the ISO format (`YYYY-MM-DD`). This is the default value.
`E`	Natural SQL Gateway returns SQL date strings in the EUR format (`DD.MM.YYYY`).

Note: This subparameter is ignored by Natural for DB2.

NSBHOST - Set Natural SQL Gateway Server Host Name

`NSBHOST=`*`value`* specifies the Natural SQL Gateway server TCP/IP host name used to communicate from TP-monitor environments such as CICS or Com-plete to CONNX JDBC in order to access SQL databases.

Value	Explanation
`hostname`	This host name designates the TCP/IP address of the Natural SQL Gateway server that communicates with the CONNX JDBC server.
`LOCALHOST`	This is the default value, meaning the Natural SQL Gateway server resides on the local host.

Notes:

1. If the host name specification contains dots (.), the host name should be surrounded with single quotes.
2. This subparameter is ignored by Natural for DB2.

Example:

```
NSBHOST='IBM2.HQ.SAG'
```

NSBPORT – Set Natural SQL Gateway Server TCP/IP Port Number

`NSBPORT=`*`value`* specifies the TCP/IP port number to which the Natural SQL Gateway server listens.

Value	Explanation
`integer`	Specifies the port number to which the Natural SQL Gateway server listens.
`0`	This is the default value, meaning no Natural SQL Gateway server port number is specified.

Note: This subparameter is ignored by Natural for DB2.

Example:

```
NSBPORT=4713
```

PSCIGN - Treat Positive SQLCODEs as SQLCODE 0

`PSCIGN=`*`value`* influences the treatment of positive SQLCODEs returned from the SQL database system.

Value	Explanation
`ON`	Positive SQLCODEs are treated as zero.
`OFF`	Positive SQLCODEs cause a NAT3700 error message. This is the default value.

Notes:

1. If the subparameter `PSCIGN` is set to `OFF`, a NAT3700 error message is issued.
2. If the subparameter `PSCIGN` is set to `ON`, positive SQLCODEs are treated as if they were zero; that is, no NAT3700 error message is issued.

REFRESH - Refresh Setting of DB2 Server and Package Set

`REFRESH=`*`value`* is used to automatically set the DB2 server and package set to the values that applied when the last transaction was executed.

Value	Explanation
`ON`	An automatic refresh is performed every time before a database transaction starts and if a server or package set has been specified.
`OFF`	No automatic refresh is performed. This is the default value.

Notes:

1. Server and package set are refreshed by using the `CONNECT TO` *`server-name`* and `SET CURRENT PACKAGESET = '`*`package-name`*`'` SQL statements of DB2.
2. This subparameter is ignored by Natural SQL Gateway.

RETRYPO - Number of Positioning Retries

`RETRYPO=`*`value`* delimits the number of retries done by Natural for DB2 in order to reposition a dynamic scrollable cursor in a pseudo-conversational environment (IMS MPP or CICS).

Value	Explanation
`1 - 2147483648`	Number of retries done by Natural for DB2.
`0`	No retries are done if `RETRYPO` is set to `0`.
`10`	This is the default value.

Notes:

1. This subparameter is ignored by Natural SQL Gateway.
2. This subparameter applies only to dynamic scrollable cursors.
3. In pseudo-conversational environments, cursors are closed at terminal I/O. For dynamic scrollable cursors the current absolute position number and the current key column values are saved. After terminal I/O, the dynamic scrollable cursor is opened again and positioned absolutely to the position of the saved absolute position. The contents of the key columns are compared with the saved values. If they match, processing continues with the next database operation requested.
4. If the contents of the key columns do not match the saved values, the next rows are fetched and compared with the saved values until either the values match or no row is found or the `RETRYPO` count is exhausted. In the latter cases, the cursor is repositioned to the saved position and the prior rows are fetched and compared until either the values match or no row is found or the `RETRYPO` count is exhausted. In the latter cases, a NAT3703 error message is issued. If a row is

fetched whose key columns match the saved values, processing continues with the next database instruction.

5. `RETRYPO` delimits the retries in each direction (*`next`* or *`prior`*).

RWRDONL - Generate Delimited Identifiers for Reserved Words Only

`RWRDONL=`*`value`* determines which identifiers are generated as delimited identifier in an SQL string.

Value	Explanation
`ON`	Only identifiers that are reserved words are generated as delimited identifiers. The list of reserved words is contained in the `NDBPARM` module. This list was merged from the lists of reserved words for DB2 for z/OS, DB2 for VSE & VM, DB2 for LINUX, OS/2, Windows and UNIX, and ISO/ANSI SQL99. This is the default value.
`OFF`	All identifiers are generated as delimited identifiers.

Note: `RWRDONL` only takes effect if the setting of the `DELIMID` subparameter allows delimited identifiers.

SMFSRV - Use Shared Memory Object File Server (FSSM)

`SMFSRV=`*`value`* specifies whether a Shared Memory Objects File Server (FSSM) is used. For more information, see *File Server – Shared Memory Object* in the section *Natural for DB2* in the *Database Management System Interfaces* documentation

Value	Explanation
`ON`	The shared memory object above the bar specified in the `DDFSERV` parameter is used for the FSSM.
`OFF`	The shared memory object is not used. Either a VSAM file or the Software AG Editor buffer pool is used as the storage medium for the file server. This is the default value.

STATDYN - Allow Static to Dynamic Switch

`STATDYN=`*`value`* is used to allow dynamic execution of statically generated SQL statements if the static execution returns an error.

Value	Explanation
`NEVER`	Dynamic execution is never allowed. This is the default value.
`ALWAYS`	Dynamic execution is always allowed after an error.
`SPECIAL`	Dynamic execution is allowed after special errors only. These special errors are: ■ NAT3706: Load module not found ■ SQL -805: DBRM (database request module) does not exist in plan ■ SQL -818: Mismatch of time stamps

Note: This subparameter is ignored by Natural SQL Gateway.

Example of DB2 Parameter

```
DB2=(FSERV=DIS,DELIMID=DQ,RWRDONL=ON,STATDYN=ALWAYS)
```

Example of NTDB2 Macro

```
         NTDB2 FSERV=ON,                                              *
             DELIMID=DQ,                                              *
             NSBHOST=LOCALHOST,                                       *
             NSBPORT=4851,                                            *
             RWRDONL=ON
```

58 DB2SIZE - Natural Buffer Area for DB2

This Natural profile parameter sets the maximum size of the buffer area required by some Natural DBMS interface products.

Possible settings	`0 - 64`	Maximum size of the buffer area in KB. **Note:** 1. If the requested space is not available, the DBMS interface cannot be used. 2. Set `DB2SIZE` to `0` if Natural is *not* to be used for DB2. 3. If Natural is to be used for DB2, `DB2SIZE` must be set to at least 40 KB.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter applies to Natural for DB2 and Natural SQL Gateway.
2. See also the following sections in the *Installation* documentation:

   Natural Parameter Modifications for Natural for DB2
   Natural Parameter Modifications for the Natural SQL Gateway

59 DBCLOSE - Database Close at Session End

This Natural profile parameter determines whether or not Natural closes all databases that it has accessed during a session at the end of this session.

Possible settings	`ON`	Natural closes all databases.
	`OFF`	Natural closes only those databases which had been opened by an explicit open command. An explicit open command will be issued in the following cases: - profile parameter `ETID` is not set to `' '` (blank), - profile parameter `DBOPEN`=`ON`, - the open is forced by the `OPEN` option of macro `NTDB` or profile parameter `DB`.
	`ETDB`	Natural closes only the database specified with the profile parameter `ETDB`.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Other transaction processing related parameters: `ADAMODE` \| `DBOPEN` \| `ENDBT` \| `ET` \| `ETDB` \| `ETEOP` \| `ETIO` \| `ETSYNC`

60 DBGAT - Debug Attach Server for NaturalONE

This Natural profile parameter allows debugging of an external Natural application from a z/OS or z/VSE host with NaturalONE. `DBGAT` corresponds to the `NTDBGAT` macro in the Natural parameter module.

Possible settings	See *DBGAT Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

For detailed information on how to debug external Natural applications, see the *NaturalONE* documentation.

This section covers the following topics:

DBGAT Parameter Syntax

The `DBGAT` parameter is specified as follows:

```
DBGAT=(keyword-subparameter=value,...)
```

Where:

Keyword Subparameter	Value	Explanation
`ACTIVE`	`ON`	`ON` means that the debug attach mechanism is active. The Natural runtime is ready for debugging.
	`OFF`	
`HOST`	1 - 64 characters	Name of the debug attach server that is to be connected.
`PORT`	0 - 65535	Number of the port to which the debug attach server listens.
`CLID`	1 - 64 characters	Client ID of the NaturalONE project that is to be debugged.
`TRACE`	1 - 8 hexadecimal digits	Trace level for attached debugging. See also *Trace Level*.

Trace Level

Each bit of `TRACE` represents certain trace information as indicated in the following table:

Bit	Information
00000001	Trace error events.
00000002	Trace main events, such as attach debug initialization and termination.
00000008	Dump internal storage areas.
00000004	Trace detailed events, such start and end of functions with return code.
00000010	Dump request and reply buffers in EBCDIC.
00000020	Dump request and reply buffers in ASCII.
00000040	Request processing main events.
00000080	Request processing detailed events.

You can combine different trace options, for example, `TRACE=00000007` traces error events, main events and detailed events.

NTDBGAT Macro Syntax

The `NTDBGAT` macro is specified as follows:

```
         NTDBGAT ACTIVE=value,                                                *
               HOST=value,                                                    *
               PORT=value,                                                    *
               CLID=value,                                                    *
               TRACE=value
```

Example of DBGAT Parameter

```
DBGAT=(ACTIVE=ON,HOST=MYHOST,PORT=9999,CLID=MYCLIENTID,TRACE=00000097)
```

Example of NTDBGAT Macro

```
         NTDBGAT ACTIVE=ON,                                                 *
               HOST=MYHOST,                                                 *
               PORT=50882,                                                  *
               TRACE=00000097
```

61 DBGERR - Automatic Start of Debugger at Runtime Error

This Natural profile parameter enables the Natural Debugger to be started automatically if a Natural error occurs at runtime.

Possible settings	`ON`	The Debugger is started automatically and produces a screen which enables you to get further information on the existing error. **Note:** 1. The runtime environment will cede control to the Debugger in the event of a Natural error, independent of whether the Debugger is already on or not. This measure avoids the manual control effort of using the Natural system command `TEST ON` in such a case. 2. For further information, see *Start the Debugger* in the Natural *Debugger* documentation.
	`OFF`	The Debugger is *not* started automatically.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

62 DBID - Default Database ID for Natural System Files

This Natural profile parameter identifies the default database in which the Natural system files (`FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF`, `FREG`) are located.

Possible settings	`0 - 254`
	`256 - 65535`
		The default database ID for Natural system files.
		Note: Database ID `255` is reserved for internal use.

Default setting	`0`	
Dynamic specification	yes	**Note:** If you specify the `DBID` parameter dynamically, the database ID for all system files is set to this setting. Therefore, you must specify the `DBID` parameter *before* any individual system file specific profile parameter (`FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF`, `FREG`) if you want to specify any of these parameters, too.
Specification within session	no	

Notes:

1. The database ID specified with the `DBID` parameter applies to all Natural system files for which no individual database IDs are specified.
2. Database IDs for individual system files can be specified with the subparameter *`database-ID`* of the profile parameters `FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF` and `FREG`.
3. The type of database system is determined by the specification in the `NTDB` macro.

63

DBOPEN - Database Open without ETID

This Natural profile parameter controls the database open handling of Natural.

Note: `DBOPEN` overrules the setting `ETID=' '` (blanks).

Possible settings	`ON`	A database open will be issued even if the `ETID` parameter is set to blanks.
	`OFF`	No database open will be issued if the `ETID` parameter is set to blanks. **Exception**: One open command will always be sent to the database specified as `ETDB`, even if `ETID` is set to blanks and `DBOPEN` is set to `OFF`.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Other transaction processing related parameters: `ADAMODE` | `DBCLOSE` | `ENDBT` | `ET` | `ETDB` | `ETEOP` | `ETIO` | `ETSYNC`

64 DBROLL - Database Calls before Session Suspension

This Natural profile parameter determines the number of database calls after which a Natural session is suspended, that is, a potential roll-out of the Natural thread is to be performed.

Possible settings	`0 - 32767`	Number of database calls.
Default setting	`0`	No session suspension for database calls.
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies under CICS and Com-plete.
2. When the non-zero `DBROLL` count is reached, Natural issues a conditional `CMROLL` request. (see *Note Concerning CMROLL* in the description of profile parameter `MAXROLL`); that is, when other sessions are waiting for a thread, the session is suspended, which may result in a roll-out of the Natural thread.
3. In CICS, if no other session is waiting, just an `EXEC CICS SUSPEND` is executed to relinquish control to other tasks of higher or equal dispatching priority.

65 DBUPD - Database Updating

This Natural profile parameter indicates whether database updating is to be permitted during the Natural session.

Possible settings	`ON`	Database update is permitted.
	`OFF`	Database update is not permitted. When compiling a program (`CHECK`, `CATALOG` or `STOW` command), a NAT0105 error message (`Database updating not permitted`) is issued if the program contains one of the following statements: `UPDATE`, `STORE`, `DELETE`, `INSERT` or `MERGE`. A database update is not performed when a program with an `UPDATE`, `STORE` or `DELETE` statement executes. Instead, a NAT1010 warning message is issued during the next screen I/O. In addition, a database loop that contains an `UPDATE` or `DELETE` statement does not place the records in hold status (no `read with hold`).
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation. * Recommended.
	`USR1042N` *	

Note: The `DBUPD` setting has no effect on the execution of Natural system commands.

66

DC - Character for Decimal Point Notation

This Natural profile and session parameter determines the character to be used as decimal separator, that is, a point or a comma.

<table>
<tr><td><b>Possible settings</b></td><td>any character (except numeric characters)</td><td colspan="3">You specify the <code>DC</code> parameter as <code>DC='<i>c</i>'</code> where <code><i>c</i></code> represents the character to be used as decimal separator.
The character specified with the <code>DC</code> parameter must not be the same as the one specified with the <code>IA</code> (input assign character) or <code>ID</code> (input delimiter character) parameter. In addition, we recommend that this character is not the same as the one specified with the <code>CF</code> (control character for terminal commands) or <code>HI</code> (help character) parameter.</td></tr>
<tr><td><b>Default setting</b></td><td><code>.</code> (period)</td><td colspan="3"></td></tr>
<tr><td><b>Dynamic specification</b></td><td>yes</td><td colspan="3"></td></tr>
<tr><td rowspan="2"><b>Specification within session</b></td><td rowspan="2">yes</td><td>Applicable statements:</td><td><code>SET GLOBALS</code></td><td>Parameter is evaluated at runtime.</td></tr>
<tr><td>Applicable command:</td><td><code>GLOBALS</code></td><td>Parameter may be specified dynamically with the <code>GLOBALS</code> system command.</td></tr>
<tr><td><b>Application programming interface</b></td><td><code>USR0350N</code>,
<code>USR1005N</code> *</td><td colspan="3">See <i>SYSEXT - Natural Application Programming Interfaces</i> in the <i>Utilities</i> documentation.
* Recommended.</td></tr>
</table>

Notes:

1. Within a Natural session, the profile parameter `DC` can be overridden by the session parameter `DC`.
2. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the *Library Profile*.

67 DD - Day Differential

This Natural profile parameter is used to adjust the current machine date (as read by using the internal machine time) by adding/subtracting any number of days to/from it. This makes it possible to re-run an application that was to be run at a certain date but for some reason could not be run at that date.

The `DD` profile parameter is specified as follows:

DD=+*nn*

or

DD=-*nn*

where *nn* is the number of days.

Possible settings	`-32767` to `+32767`	Machine date is adjusted. Specification of the plus sign (+) is optional.
	`0`	No adjustment is made.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

See also the profile parameters `TD` and `YD`.

68 DELETE - Deletion of Dynamically Loaded Programs

This Natural profile parameter determines whether dynamically loaded non-Natural programs are to be deleted on the completion of the Natural program in which they are loaded.

You can use the profile parameter `CDYNAM` to limit the number of non-Natural programs that can be loaded simultaneously.

Possible settings	`ON`	Dynamically loaded non-Natural programs are deleted at the end of the Natural program in which they were loaded.
	`OFF`	Dynamically loaded non-Natural programs are not deleted at the end of the Natural program in which they were loaded; they are kept until the current Level 1 program terminates.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

The following platform-specific requirements apply:

Platform:	Comment:
Under CICS	In a CICS environment, this parameter applies only if the non-Natural program is invoked via standard linkage conventions (`SET CONTROL 'P=S'`).
Under z/OS Batch, TSO, z/VSE Batch and IMS TM	This parameter does not apply in an IBM Language Environment (LE). All dynamic programs loaded during a Natural session are deleted upon LE environment termination, that is, during the termination of the Natural session. For more information about Natural in an LE environment, see *Support of IBM LE Subprograms* in the *Operations* documentation.

69 DF - Date Format

With the `DF` session parameter, you determine the length of a date when converted into alphanumeric representation without an edit mask being specified.

Possible settings	`S`	8-byte representation with 2-digit year component and delimiters (*`yy-mm-dd`*). With `DF=S`, only 2 digits are provided for the year information; this means that if the date value contained the century, this information would be lost during the conversion.
	`I`	8-byte representation with 4-digit year component and no delimiters (*`yyyymmdd`*). See **Note**.
	`L`	10-byte representation with 4-digit year component and delimiters (*`yyyy-mm-dd`*). See **Note**.
Default setting	`S`	
Applicable statements	`FORMAT`	

	`INPUT`
	`DISPLAY`
	`WRITE`
	`PRINT`
		Parameter may be specified at statement level and/or at element level.

	`MOVE`
	`COMPRESS`
	`STACK`
	`RUN`
	`FETCH`
		Parameter may be specified at element level.

Applicable command	none	

Notes:

1. The `DF` parameter is evaluated at compilation time.

2. The sequence of the day, month and year components and the delimiter characters used are determined by the profile parameter `DTFORM`.
3. When the value of a date field is converted into alphanumeric format (for example, in a `MOVE`, `DISPLAY`, `WRITE` or `INPUT` statement) and no edit mask is specified for the conversion, the default date format as determined by the profile parameter `DTFORM` is used as edit mask.
4. The same is true for the input validation of a date variable used in an `INPUT` statement: If no edit mask is specified, the input is validated according to the date format determined by the `DTFORM` parameter.
5. By using `DF=I` or `DF=L`, you can gradually change your applications to use 4-digit year representations and at the same time continue to make use of the flexibility provided by the profile parameter `DTFORM`.
6. See also *Date Format for Alphanumeric Representation - DF Parameter* in the *Programming Guide*.

70 DFOUT - Date Format for Output

This Natural profile and session parameter determines the format in which the settings of date variables are displayed by `INPUT`, `DISPLAY`, `PRINT` and `WRITE` statements.

Possible settings	`S`	Date variables are displayed with a 2-digit year component, and delimiters as determined by the profile parameter `DTFORM`. Example: *yy-mm-dd*	
	`I`	Date variables are displayed with a full 4-digit year component and no delimiters. Example: *yyyymmdd*	
Default setting	`S`		
Dynamic specification	yes		
Specification within session	yes	Applicable statements:	`SET GLOBALS`
		Applicable command:	`GLOBALS`
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.	

Notes:

1. Within a Natural session, the profile parameter `DFOUT` can be overridden by the session parameter `DFOUT`.
2. The *profile parameter* `DFOUT` is evaluated at runtime.
3. It applies to date fields in `INPUT`, `DISPLAY`, `PRINT` and `WRITE` statements for which no explicit edit mask is specified and for which the *session parameter* `DF` is not set.

4. The sequence of the day, month and year components in the date settings is determined by the `DTFORM` profile parameter.
5. See also *Processing of Date Information* in the *Programming Guide*.

71 DFSTACK - Date Format for Stack

This Natural profile and session parameter determines the format in which the settings of date variables are placed on the stack via a `STACK`, `RUN` or `FETCH` statement.

Possible settings	`S`	Date variables are placed on the stack with a 2-digit year component, and delimiters as determined by the profile parameter `DTFORM`. Example: *`yy-mm-dd`*	
	`C`	Same as `DFSTACK=S`. In addition, if the century used when the setting is read from the stack is not the same as that of the original date setting, Natural will issue a runtime error.	
	`I`	Date variables are placed on the stack with a full 4-digit year component and no delimiters. Example: *`yyyymmdd`*	
Default setting	`S`		
Dynamic specification	yes		
Specification within session	yes	Applicable statement:	`SET GLOBALS`
		Applicable command:	`GLOBALS`
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.	

Notes:

1. Within a Natural session, the profile parameter `DFSTACK` can be overridden by the session parameter `DFSTACK`.
2. The profile parameter `DFSTACK` does not apply to `STACK`, `RUN` or `FETCH` statements for which the session parameter `DF` is set.
3. See also *Processing of Date Information* in the *Programming Guide*.

72 DFTITLE - Output Format of Date in Standard Report Title

This Natural profile and session parameter determines the output format of the date in the default title line of a report page (as output with a `DISPLAY`, `WRITE` or `PRINT` statement).

Possible settings	`S`	The date is output with a 2-digit year component and delimiters. Example: *`yy-mm-dd`*	
	`L`	The date is output with a 4-digit year component and delimiters. Example: *`yyyy-mm-dd`*	
	`I`	The date is output with a 4-digit year component and no delimiters. Example: *`yyyymmdd`*	
Default setting	`S`		
Dynamic specification	yes		
Specification within session	yes	Applicable statement:	`SET GLOBALS`
		Applicable command:	`GLOBALS`
Application programming interface	`USR1005N`	See *SYSEXT Utility - Natural Application Programming Interfaces* in the *Utilities* documentation.	

Notes:

1. Within a Natural session, the profile parameter `DFTITLE` can be overridden by the session parameter `DFTITLE`.
2. `DFTITLE` is evaluated at runtime and determines whether the date is displayed with a 2-digit or 4-digit year component with or without delimiters.

3. It has no effect on a user-defined page title (as specified with a `WRITE TITLE` statement).
4. The sequence of the day, month and year components and the delimiter characters used are determined by the profile parameter `DTFORM`.
5. See also *Processing of Date Information* and *Date Format for Default Page Title - DFTITLE Parameter* in the *Programming Guide*.

73 DL - Display Length for Output

With this session parameter, you specify the display length for a field of format A or U. The default display length is the length of the field.

Possible settings	`1` to *n*	*n* = value of `LS` (line size) parameter minus 1
Default setting	none	
Applicable statements	`FORMAT`	

	`DISPLAY`
	`INPUT`
	`PRINT`
	`WRITE`
		Parameter may be specified at statement level and/or at element level.

Applicable command	none	

Example:

```
FORMAT DL=20
```

For further information and an example of the `DL` session parameter usage, see the following topics in the *Programming Guide*:

- *Parameters to Influence the Output of Fields*
- *Output Length - AL and NL Parameters*
- *Display Length for Output - DL Parameter*

74

DLISIZE - Size of Natural Buffer Area for DL/I

This Natural profile parameter determines the maximum size of the buffer area required by Natural for DL/I.

Possible settings	`26 - 512`	Buffer size in KB. **Note:** 1. The size actually required depends on the specifications in the `NDLPARM` macro (see the *Natural for DL/I* documentation). 2. If you use the default specifications in `NDLPARM, DLISIZE=26` is sufficient.
	`0`	If you do not need DL/I support during a Natural session, you are recommended to invoke Natural with `DLISIZE=0` to avoid overhead caused by handling of unused buffers.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Natural for DL/I.
2. If the size specified with the `DLISIZE` parameter is not sufficient, an appropriate error message at initialization of Natural for DL/I will tell you what size to specify.
3. If Natural for DL/I is installed, the corresponding Natural buffers are requested at the initialization of the Natural session.
4. If the requested space is not available, Natural for DL/I cannot be used.

75 DO - Display Order of Output Data

This Natural profile and session parameter specifies how fields are to be interpreted for display on terminals that support bidirectional data.

Possible settings	`L`	Specifies that the data from the application is in logical display order. **Note:** The field characters are displayed according to their character property (left-to-right or right-to-left).	
	`V`	Specifies that the data from the application is in visual order. **Note:** 1. All fields are inverted by Natural before they are sent to the terminal. 2. This option is required for old applications written for terminals which support inverse (right-to-left) print mode, activated by profile parameter `PM=I` or terminal command `%VON`.	
Default setting	`L`		
Dynamic specification	yes		
Specification within session	yes	Applicable statement:	`SET GLOBALS`
		Applicable command:	`GLOBALS`

Notes:

1. The I/O device must be able to create the correct display order depending on the character properties. This is for instance the case if an application runs in a browser under Natural Web I/O Interface. For other terminal types, this parameter does not have any effect.
2. For detailed information on how to use the setting `PM=I`, see *Bidirectional Language Support* in the *Unicode and Code Page Support* documentation.

76

DS - Define Size of Storage Buffer

This Natural profile parameter defines the default initial size of various Natural storage buffers. It corresponds to the `NTDS` macro in the Natural parameter module.

Possible settings	See *DS Parameter Syntax*.	
Default setting	See *Table of Buffer Sizes*.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the corresponding macro `NTDS` is used instead.
Specification within session	no	

Notes:

1. In previous versions of Natural, individual profile parameters (for example, `SSIZE`) were used to define the sizes of the buffers. The `DS` profile parameter is a universal parameter to specify all buffer sizes.
2. You may continue using the individual parameters or you may use the individual parameters in parallel to the parameter `DS`. During the dynamic parameter evaluation, individual buffer size parameters are converted internally into the new `DS` parameter format, for example, `SSIZE=55` is converted into `DS=(SSIZE,55)`.
3. However, there are some buffer sizes (for example, `ESIZE`, `VSIZE`, etc.) which cannot be specified by the profile parameter `DS`, due to certain reasons, for example, the size is part of a larger buffer or the size defines the total maximum of a number of buffers.
4. For further information, see *Natural Storage Management* and *General Rules for Parameter Usage* in the *Operations* documentation.

The following topics are covered below:

DS Parameter Syntax

The `DS` parameter is specified as follows:

```
DS=(name,size,name,size,...)
```

Where:

Syntax Element	**Value**	**Explanation**
`name`	1 - 8 characters.	The buffer name (1-8 characters), see *Table of Buffer Sizes*.
`size`	-	The buffer size in kilobytes. For limit values, see *Table of Buffer Sizes*.

Note: Multiple pairs of buffer name and size values can be specified; see *Example of DS Parameter*.

NTDS Macro Syntax

The `NTDS` macro is specified as follows:

```
        NTDS   name,size
        NTDS   name,size
        ...
```

Where:

Syntax Element	**Value**	**Explanation**
`name`	1 - 8 characters.	The buffer name (1-8 characters), see *Table of Buffer Sizes*.
`size`	-	The buffer size in kilobytes. For limit values, see *Table of Buffer Sizes*.

Note: A separate `NTDS` macro must be specified for each pair of name and size values; see *Examples of NTDS Macros*.

Table of Buffer Sizes

Buffer Name	**Description**	**Buffer Size (KB)**	**Default**	**Available as subparameter of `DS` and alternatively as individual profile parameter**
`ASIZE`	Entire System Server auxiliary buffer	`0` or `64 - 512`	`0`	yes **Note:** For details, see description of profile parameter `ASIZE`.
`BSIZE`	Size of EntireX Broker buffer	`0` or `1 - 64`	`0`	yes
`CSIZE`	Size of Con-nect/Con-form buffer area	`0` or `1 - 512`	`0`	yes
`DATSIZE`	Size of buffer for local data	`10 - 2097151`	`32`	yes

Buffer Name	Description	Buffer Size (KB)	Default	Available as subparameter of `DS` and alternatively as individual profile parameter
`DSIZE`	Initial size of `DBLOG` buffer	`0` or `2 - 2097151`	`2`	yes **Note:** The individual profile parameter `DSIZE` allows you to set a maximum size in addition.
`EDPSIZE`	Size of the Software AG Editor auxiliary buffer pool	`0` or `48 - 2097151`	`0`	yes
`EXCSIZE`	Size of buffer for Natural Expert C interface	n/a	n/a	yes **Note:** This subparameter is only retained for compatibility reasons.
`EXRSIZE`	Size of buffer for Natural Expert rule tables	n/a	n/a	yes **Note:** This subparameter is only retained for compatibility reasons.
`FLTUSER`	Size of buffer for fast locate table and subroutine cache. Information contained in the fast locate table improves performance when Natural objects are called repeatedly. Information contained in the subroutine cache improves performance when the name of the module in which the subroutine is defined is being retrieved from the subroutine name specified with the `PERFORM` statement. The `FLTUSER` buffer contains only information related to Natural objects belonging to the user's application.	`0` or `4 - 2097151`	`8`	Available only as subparameter of `DS`. **Note:** The value `0` enforces that neither a fast locate table nor a subroutine cache is allocated and will cause more database calls to be issued in order to retrieve information about Natural objects that are to be called. Therefore, the value `0` should only be used in environments where Natural objects of the same name exist in different steplib libraries (see *Steplib Libraries* in *Using Natural*), and the steplib libraries should always be searched for an object in the defined order. The following formula helps estimate the initial buffer size required to store all entries in the fast locate table: `6 KB + (number of objects * ↩ 112 bytes)` which corresponds to a setting of `DS=(FLTUSER,21)`. This specifies the buffer size for the session start; more space is allocated automatically during the session if required.

Buffer Name	Description	Buffer Size (KB)	Default	Available as subparameter of `DS` and alternatively as individual profile parameter
`MONSIZE`	Size of `SYSTP` monitor buffer	`0` or `5 - 256`	`0`	yes
`MULFETCH`	Size of multi-fetch buffer	`0` or `1 - 1024`	`64`	Available only as subparameter of `DS`. **Note:** A value specified for this buffer does not represent the default initial size but the maximum size which can be allocated for multi-fetch purposes. See *Size of the Multi-Fetch Buffer* in the *Programming Guide*.
`NAFSIZE`	Size of buffer for Natural Advanced Facilities	`0` or `1 - 64`	`0`	yes
`NSFSIZE`	Size of SAF interface buffer.	`0` or `8 - 64`	`0`	Available only as subparameter of `DS`. See information on how to adjust the Natural parameter module in *Installing and Activating Natural SAF Security* in the *Natural SAF Security* documentation.
`RDCSIZE`	Size of buffer for the Natural Data Collector	`0` or `2 - 128`	`0`	yes
`RJESIZE`	Initial size of `NATRJE` buffer	`0` or `1 - 2097151`	`8`	yes
`RUNSIZE`	Size of runtime buffer	`10 - 64`	`16`	yes
`SSIZE`	Size of Software AG Editor buffer	`0` or `40 - 512`	`64`	yes
`TSIZE`	Size of the buffer for Adabas Text Retrieval	`0` or `1 - 2097151`	`0`	yes
`WSISIZE`	Buffer for Natural Workstation Interface	`0` or `10 - 256`	`0`	yes
`ZSIZE`	Size of Entire DB buffer area	`0` or `1 - 64`	`0`	yes

For detailed information, refer to the descriptions of the individual buffer size parameters.

Example of DS Parameter

```
DS=(ASIZE,33,TSIZE,60,EDPSIZE,500)
```

Example of NTDS Macro

```
         NTDS ASIZE,33
         NTDS TSIZE,60
         NTDS EDPSIZE,500
```

77 DSC - Data-Stream Compression (for 3270-Type Terminals)

With this parameter, you can switch off Natural's automatic optimization *and* compression of the screen data stream for 3270-type terminals.

Possible settings	`ON`	Data-stream optimization and compression are used.
	`OFF`	Data-stream optimization and compression are *not* used.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to 3270-type terminals.
2. Screen optimization means that only those fields of the screen are sent to the terminal whose content has changed. Screen compression constitutes a (further) reduction of the amount of data sent by using counters for repeating characters.
3. Natural's screen optimization causes screen data to be sent as compressed as possible. If this should conflict with any TP monitor's screen optimization or hardware limitation, you can use this parameter to switch off Natural's screen optimization; screen data will then be sent in non-compressed form; for example, see *Profile Parameter DSC=OFF Recommended* in the *Natural under CICS* documentation.
4. This parameter has the same function as the terminal command `%RO`.
5. If you use the `BX` session parameter settings `BX=L` or `BX=R`, you should switch off Natural's screen optimization using `DSC=OFF` or `%ROOFF`.

78 DSIZE - Size of DBLOG Buffer

This Natural profile parameter specifies the size of the Natural `DBLOG` buffer.

Possible settings	See *DSIZE Parameter Syntax*.	
Default setting	`2`	*`initial-size`*
	`256`	*`maximum-size`*
Dynamic specification	yes	
Specification within session	no	

Notes:

1. Alternatively, you can use the Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.
2. The Natural `DBLOG` buffer area is used by the DBLOG utility, which is described in the *Utilities* documentation.

DSIZE Parameter Syntax

The parameter syntax is as follows:

DSIZE=*`initial-size`*

Or:

`DSIZE=(initial-size)`

Or:

`DSIZE=(initial-size,maximum-size)`

Where:

Syntax Element	Value	Explanation
initial-size	`2 - 2097151`	`DBLOG` buffer initial size in KB. If the initial size is not sufficient, the DBLOG utility automatically increases the buffer size (repeatedly, if necessary) up to the specified *maximum-size*.
	`0`	Disables the DBLOG utility.
maximum-size	`2 - 2097151`	`DBLOG` buffer maximum size in KB. If the value is not greater than the *initial-size*, the buffer is not increased.
	`0`	No increase limit. Allows the DBLOG utility to increase the buffer up to the maximum of storage available.

Examples:

```
DSIZE=100          * initial-size = 100                maximum-size =  256 (default)
```

```
DSIZE=(,2500)      * initial-size =   2 (default)      maximum-size = 2500
```

```
DSIZE=(50,800)     * initial-size =  50                maximum-size =  800
```

79

DTFORM - Date Format

This Natural profile parameter indicates the default format in which dates are to be provided automatically by Natural as part of the default title on Natural reports, as date constants and date input.

Possible settings	**Value**	**Area**	**Date Format**
	`E`	Europe	`DD/MM/YYYY`
	`G`	Germany	`DD.MM.YYYY`
	`I`	International	`YYYY-MM-DD`
	`U`	USA	`MM/DD/YYYY`
Default setting	`I`		
Dynamic specification	yes		
Specification within session	no		
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.	

Notes:

1. The first day of a week is assumed to be Monday - unless `DTFORM=U` is specified, in which case Sunday is used.
2. For date constants, the year component (`YYYY`) consists of all four digits. Only the last two digits of the year component are used for reports, date input, the Natural system function `VAL`, and when the date is moved to an alphanumeric field.
3. The output format of the date in a default report page title is also specified by the profile parameter `DFTITLE`.
4. See also *Processing of Date Information* and *Default Edit Mask for Date - DTFORM Parameter* in the *Programming Guide*.

80

DU - Dump Generation

This Natural profile and session parameter determines whether a memory dump is to be generated in the case of an abnormal termination during the Natural session.

Possible settings	`ON`	A memory dump is produced in the case of an abnormal termination (TP-monitor dump data set or `SYSUDUMP` in z/OS batch mode or under TSO). Then the Natural session terminates with the error message NAT9967 or NAT9974.
	`OFF`	No memory dump is produced. **Note:** 1. In batch mode, subsequent action taken by Natural is determined by the setting of the `CC` profile parameter. 2. In online mode, Natural responds with the error message NAT0950, NAT0953, NAT0954, NAT0955 or NAT0956. 3. For further information on the abnormal termination, you can use the system command `DUMP`.
	`SNAP`	This setting forces an immediate dump in the case of an abnormal termination during a Natural session. **Note:** The Natural session continues as with `DU=OFF` after the dump has been taken.
	`FORCE`	This setting forces an immediate dump in the case of an abnormal termination during a Natural session and terminates the Natural session immediately. This is useful for testing purposes in some environments. **Note:** If Natural is LE enabled, Natural terminates the Natural session immediately without a dump and passes control to the LE error handling. It is therefore strongly recommended to specify the LE run-time option `TERMTHDACT(UAIMM)` to get all the required diagnostic information.
	`ABEND`	This works as with `DU=ON`, except that the session is terminated with the abend occurred - instead of the error message NAT9974.

		Note: `DU=ABEND` is not available with the Natural session parameter `DU`.	
Default setting	`OFF`		
Dynamic specification	yes		
Specification within session	yes	Applicable statement:	`SET GLOBALS`
		Applicable command:	`GLOBALS`

Notes:

1. Within a Natural session, the profile parameter `DU` can be overridden by the session parameter `DU`.
2. Setting the `DU` profile parameter may impair the system performance considerably, due to I/O processing on the dump data set.
3. Be careful when you use this parameter, because all programs and subroutines currently active for the current user will be retained in the Natural buffer pool.
4. `DU=ON`, `DU=SNAP` or `DU=FORCE` may cause buffer fragmentation which may result in a significant degradation in system performance.
5. Under *open*UTM, this parameter is ignored; under *open*UTM, a dump is always produced in the case of an abnormal program termination.
6. Profile parameter `DUE` can be used to get a storage dump for specific errors.
7. Under Natural Security, the setting of this parameter can be overridden by the *Session Parameters* option of the Library Profile.

81 DUE - Dump Generation, Error-Specific

This Natural profile parameter can be used to specify Natural error numbers for which a storage dump shall be taken.

Possible settings	`1 - 9999`	A single error number or multiple error numbers (separated by a comma and enclosed in parentheses) for which a dump shall be taken. **Note:** If `DUE` is specified multiple times, all error numbers are saved in one table.
	`OFF`	Deletes the table and any error numbers specified previously are removed.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	Terminal command `%DUE`

Notes:

1. This type of storage dump may be helpful to get a dump for the analysis of a specific error situation by Software AG personnel.
2. If an error occurs which has been specified by `DUE`, a program check is forced.
3. If the profile/session parameter `DU=OFF` is set, it will be changed to `DU=ON`. For further processing, the `DU` parameter setting is honored.

Examples:

```
DUE=1302
DUE=(6501,6502,6503,1500)
DUE=OFF
```

82

DY - Dynamic Attributes

This session parameter is used to assign attributes for dynamic attribute field display.

Possible settings	See *DY Parameter Syntax*.	
Default setting	none	

Applicable statements	`DISPLAY`
	`INPUT`
	`PRINT`
	`WRITE`
		Parameter may be specified at statement level and/or at element level.

Applicable command	none	

Special identification characters (escape characters) are used to indicate the beginning and end of attribute definitions.

An alphanumeric field which is processed with an `INPUT`, `DISPLAY`, `WRITE` or `PRINT` statement, and which contains escape characters, is split into subfields at the escape character position. The corresponding attribute is then assigned to the subfield. A blank is substituted for the escape character.

For a part of a field for which a `DY` specification applies, the current field presentation and color is changed to what is newly defined in the `DY` entry. If the `DY` segment *does not contain a new setting* for the

- **field presentation**

  (means *no* `B`, `C`, `D`, `I`, `N`, `U`, `V`), the attribute active for the complete field remains in effect, regardless of whether originally derived from a static setting (for example, `AD=I`) or from a control variable (for example, `CV=#C`).

- **field color**

  (means *no* `BL`, `GR`, `NE`, `PI`, `RE`, `TU`, `YE`), the color is set to what is statically assigned to the field (with `CD=..`), without considering a color which was possibly set via a control variable (`CV=..`). If the field has no static (`CD=..`) assignment, the color information is completely removed from the field segment affected by the `DY` manipulation.

The following topics are covered below:

DY Parameter Syntax

DY={{*escape-character1*} [*color-attribute*] [*i/o-characteristics*] [*field-representation-attribute*] } . . . {*escape-character2*}

The possible settings are explained below.

escape-character1

An escape character which denotes the beginning of the attribute definition. Any special character or a hexadecimal number preceded by an apostrophe (`'xx`) may be used.

color-attribute

The color attribute to be assigned. See also the session parameter `CD` (color definition).

`BL`	blue
`GR`	green
`NE`	neutral
`PI`	pink
`RE`	red
`TU`	turquoise
`YE`	yellow

i/o-characteristics

Value	Meaning
`P`	Subfield is to be write-protected.

A `P` may be specified to make the subfield write-protected. See also the session parameter `AD` (attribute definition).

field-representation-attribute

Additional attributes to be assigned. See also the session parameter `AD` (attribute definition).

Value	Meaning
`B`	blinking (*)
`C`	cursive/italic (*)
`D`	default intensity
`I`	intensified
`N`	non-display
`U`	underlined
`V`	reverse video (*)

* The field representation attributes marked with an asterisk (*) require corresponding hardware features, and will be ignored at runtime if these features are not available.

`escape-character2`

An escape character which denotes the end of the attribute definition. Any special character (*`c`*) or a hexadecimal number preceded by an apostrophe (`'xx`) may be used.

You may specify up to eight escape sequences (escape characters and attributes) before the character indicating the end of the attribute definitions.

Examples

Example 1:

```
DY=<U>
```

The text string:

```
THIS <is> UNDERLINED
```

is printed as:

```
THIS is UNDERLINED
```

Example 2:

```
DY=<BL|RE/GR>
```

Assigns:

Blue to `<`

Red to `|`

Green to `/`

`>` switches back to the initial field color.

Example 3:

```
DY=<P>;
```

The text string:

```
Do not overwrite <this>
```

is printed as:

```
Do not overwrite this
```

(where `this` is protected)

83 DYNPARM - Control Use of Dynamic Parameters

This Natural profile parameter can be used to restrict the use of dynamic profile parameters outside of `PROFILE` and `SYS` profile parameter strings. It corresponds to the `NTDYNP` macro in the Natural parameter module.

Possible settings	`ON` or `OFF`, with or without list of profile parameters. See ***DYNPARM Parameter Syntax***.	
Default setting	`ON`	With `DYNPARM=ON`, all profile parameters can be specified dynamically.
Dynamic specification	yes	Outside of `PROFILE` or `SYS` parameter strings, the `DYNPARM` parameter can be used only once and only if the `NTDYNP` macro is not specified in the Natural parameter module.
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The parameter restrictions defined by `DYNPARM` (or the `NTDYNP` macro) do not apply within `PROFILE` or `SYS` profile parameter strings. If `DYNPARM` is used within `PROFILE` or `SYS` strings, it replaces any previous restrictions defined by `DYNPARM` or macro `NTDYNP`.
2. `DYNPARM` can be used only once within one string and should be placed at the end of it.

The following topics are covered below:

DYNPARM Parameter Syntax

The `DYNPARM` parameter is specified as follows:

```
DYNPARM=ON
```

All profile parameters can be specified dynamically.

```
DYNPARM=(ON,parameter-name,parameter-name,...)
```

Only those parameters whose *`parameter-name`* is specified, can be specified dynamically. Other parameters cause Error Message NAT7008 to be issued.

Or:

```
DYNPARM=OFF
```

No profile parameters can be specified dynamically.

```
DYNPARM=(OFF,parameter-name,parameter-name,...)
```

All profile parameters can be specified dynamically - except those whose `parameter-name` is specified. These parameters cause Error Message NAT7008 to be issued.

NTDYNP Macro Syntax

The `NTDYNP` macro is specified as follows:

```
         NTDYNP ON,parameter-name1,parameter-name2,parameter-name3
         NTDYNP parameter-name4,parameter-name5,...
```

Only those parameters whose `parameter-name` is specified, can be specified dynamically. Other parameters cause Error Message NAT7008 to be issued.

Or:

```
         NTDYNP OFF,parameter-name1,parameter-name2,parameter-name3
         NTDYNP parameter-name4,parameter-name5,...
```

All profile parameters can be specified dynamically - except those whose `parameter-name` is specified. These parameters cause Error Message NAT7008 to be issued.

Examples

The example illustrates restricting of the dynamic parameters `FNAT` and `FSEC`. In the Natural parameter module, the following parameter restriction should be defined:

```
         NTPRM DBID=0,FNR=0
         NTDYNP ON,PROFILE
```

Additionally, almost all parameter profiles could look like the following:

```
...,FNAT=(22,7,PASSW),FSEC=(22,9,PASSW),DYNPARM=(OFF,FNAT,FSEC)
```

If some special users are to be allowed to use all parameters including `FNAT` and `FSEC`, their parameter profiles could look like the following:

```
USER=(ADM1,ADM2),...,FNAT=(22,8),FUSER=(22,12),DYNPARM=(OFF,DUMMY)
```

This forces normal users to enter the `PROFILE` parameter as the first dynamic parameter. Subsequently, all parameters except `FNAT` and `FSEC` are allowed. Of course, the access to the parameter profile application `SYSPARM` must be restricted.

84 ECHO - Control Printing of Batch Input Data

This Natural profile parameter is used to enable or disable the printing of input data from the data set `CMSYNIN` or `CMOBJIN` for `INPUT` statements provided to Natural during batch mode processing.

Possible settings	`ON`	Natural prints the input data provided during batch mode processing to the batch output file `CMPRINT`.
	`OFF`	Natural does *not* print input data provided during batch processing.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies in batch mode.
2. It is also possible to suppress printing of a *single input line* by preceding it with a line containing the terminal command for record suppression `%*`.

85 EDBP - Software AG Editor Buffer Pool Definitions

This Natural profile parameter controls the initialization and operation of the editor buffer pool and its work file. It corresponds to the `NTEDBP` macro in the Natural parameter module.

Possible settings	See *EDBP Parameter Syntax*.	
Default setting	See *Keyword Subparameters*.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTEDBP` is used instead.
Specification within session	yes	Use the SYSEDT utility; see *SYSEDT Utility - Editor Buffer Pool Administration*.

Notes:

1. The editor buffer pool is defined for a session by profile parameter `BPI` with `TYPE=EDIT` or by profile parameter `EDPSIZE` (editor auxiliary buffer pool).
2. Shared Editor Buffer Pool: If the editor buffer pool is shared between multiple Natural sessions, all subparameters (except `DDNAME`, `DSNAME` and `FMODE`) are honored by the very first session only, which initializes the editor buffer pool work file during a buffer pool cold start. During a buffer pool warm start, the editor buffer pool subparameters (except `DDNAME`, `DSNAME` and `FMODE`) are read from the buffer pool work file. With subparameter `COLD=ON`, a buffer pool cold start can be forced during the initialization of the editor buffer pool.
3. Editor Auxiliary Buffer Pool: If an editor auxiliary buffer pool is used (see profile parameter `EDPSIZE`), only the following subparameters apply: `FTOUT` | `LRECL` | `MAXLF`
4. For further information on the editor buffer pool, refer to *Editor Buffer Pool* in the *Operations* documentation.
5. For information on buffer pool performance, refer to *SYSEDT Utility - Editor Buffer Pool Administration* in the *Utilities* documentation.

The following topics are covered below:

EDBP Parameter Syntax

The `EDBP` parameter is specified as follows:

```
EDBP=(keyword-subparameter=value, keyword-subparameter=value,...)
```

See *Keyword Subparameters*.

NTEDBP Macro Syntax

The `NTEDBP` macro is specified as follows:

```
         NTEDBP CHECK=value,                                           *
               COLD=value,                                             *
               CTOUT=value,                                            *
               DDNAME=value,                                           *
               DSNAME=value,                                           *
               DTOUT=value,                                            *
               FMODE=value,                                            *
               FTOUT=value,                                            *
               IMSG=value,                                             *
               ITOUT=value,                                            *
               LRECL=value,                                            *
               LTOUT=value,                                            *
               MAXLF=value,                                            *
               PWORK=value,                                            *
               RECNUM=value,                                           *
               RWORK=value,                                            *
               UTOUT=value
```

See ***Keyword Subparameters***.

Keyword Subparameters

`CHECK` | `COLD` | `CTOUT` | `DDNAME` | `DSNAME` | `DTOUT` | `FMODE` | `FTOUT` | `IMSG` | `ITOUT` | `LRECL` | `LTOUT` | `MAXLF` | `PWORK` | `RECNUM` | `RWORK` | `UTOUT`

CHECK – Check and Initialize the Software AG Editor at Session Start

`CHECK=value` specifies whether the Software AG Editor is checked and initialized already during the start of the Natural session rather than at the first call of the Software AG Editor.

Value	Explanation
`ON`	The Software AG Editor is initialized during the Natural session start if the following conditions are true: 1. The Software AG Editor module `NATEDT` is linked to Natural. 2. The profile parameter `SSIZE` is set to a nonzero value. During the Software AG Editor initialization, the `SSIZE` buffer is allocated, and the Software AG Editor buffer pool and (if `EDPSIZE=0`) its work file are accessed and checked for availability.

Value	Explanation
`OFF`	The check and the initialization are done during the first call of the Software AG Editor. This is the default value.

COLD - Buffer Pool Cold Start

`COLD=`*`value`* specifies whether a buffer pool cold start is performed.

Value	Explanation
`ON`	Cold start is performed.
`OFF`	Cold start is not performed. This is the default value.

Note: A cold start means that the buffer pool work file is cleared and reinitialized during buffer pool initialization. Any editor recovery information and all buffer pool parameters stored in the work file are lost.

CTOUT - Timeout for Changed Buffer Pool Blocks

`CTOUT=`*`value`* specifies the timeout value for changed buffer pool blocks.

Value	Explanation
`1 - 32767`	Timeout value (in seconds).
`120`	This is the default value.

Note: A changed buffer pool block is written to the work file after the specified time interval has been exceeded, and no unchanged or free block is available.

DDNAME - Logical Work File Name of the JCL Definition

`DDNAME=`*`value`* specifies the logical work file name of the JCL definition.

Value	Explanation
1 - 8 bytes.	Name of the logical work file.
`CMEDIT`	This is the default value.

Notes:

1. Under CICS: A corresponding file control table entry must be defined for the editor work file.
2. Under Com-plete: The specified logical work file name is the name of the `SD` file.

DSNAME - Work File Data Set Name

`DSNAME=`*`value`* specifies the work file data set name for batch and TSO.

Possible values:

Value	Explanation
`1 - 44`	Name of the work file data set (in bytes).
	There is no default value.

This subparameter applies under z/OS only.
If no DD JCL statement is supplied and no `ALLOC` statement is issued (under TSO only) for the editor work file, then `DSNAME` will be allocated dynamically.

DTOUT - Logical File Timeout Check Value

`DTOUT=`*`value`* specifies the logical file timeout check value.

Value	Explanation
`1 - 32767`	Timeout check value (in seconds).
`300`	This is the default value.

Note: Logical files are checked for timeout each time the specified time interval has been exceeded.

FMODE - Work File Mode

`FMODE=`*`value`* specifies the file mode for the work file.

Value	Explanation
1 - 2 characters	File mode. Under Com-plete/SMARTS, the value `SM` determines that a SMARTS work file is used. In this case, the SMARTS environment variable `$NAT_WORK_ROOT` determines the path. Under Com-plete/SMARTS, if a value other than `SM` is specified, a Com-plete SD file is used. In a SMARTS environment without Com-plete, `SM` must be specified.
`A1`	This is the default value.

Note: This subparameter applies under Com-plete/SMARTS only.

FTOUT - Timeout Value for Logical Files

`FTOUT=`*`value`* specifies the timeout value for logical files.

Value	Explanation
`60 - 16777215`	Timeout value (in seconds).
`86400`	This is the default value.

Note: A logical file is deleted after the specified time interval has been exceeded and no access has occurred.

IMSG - Buffer Pool Initialization and Termination Message

`IMSG=`*`value`* specifies whether a buffer pool initialization and termination message is issued on the operator console.

Value	Explanation
`ON`	A buffer pool initialization and termination message is issued.
`OFF`	No buffer pool initialization and termination message is issued. This is the default value.

ITOUT - Buffer Pool Initialization Timeout Value

`ITOUT=`*`value`* specifies the buffer pool initialization timeout value for multi-user buffer pools only.

Value	Explanation
`1 - 32767`	Buffer pool initialization timeout value (in seconds).
`300`	This is the default value.

Note: The buffer pool is initialized by the first user by whom it is accessed. Other users have to wait until the first user finishes initialization. If the initialization is not finished after the specified time interval (for example, due to an abnormal termination of the first user), all other users receive an error message.

LRECL - Work File Record Length

`LRECL=value` specifies the buffer pool block size and work file record length.

Value	Explanation
`800 - 16384`	Length (in bytes). Under BS2000, the record length must be a multiple of 2048 bytes.
`4096`	This is the default value.

Notes:

1. This parameter is honored under BS2000, under Com-plete, and for editor auxiliary buffer pools only.
2. For other environments, the work file record length is determined when the editor work file is created.

LTOUT - Timeout Value for Locked Buffer Pool Blocks

`LTOUT=value` specifies the timeout value for locked buffer pool blocks.

Value	Explanation
`1 - 32767`	Timeout value (in seconds).
`20`	This is the default value.

Note: A buffer pool block that was locked during a read from the work file is freed after the specified time interval has been exceeded.

MAXLF - Maximum Number of Logical Files

`MAXLF=value` specifies the maximum number of logical files.

Value	Explanation
`100 - 999999`	Maximum number of logical files.
`1000`	This is the default value.

PWORK - Percentage of Work File Records Used as Work Records

`PWORK=value` specifies the percentage of work file records used as work records during an editor buffer pool cold start.

Value	Explanation
`0 - 100`	Work file records used (in percent).
`50`	This is the default value.

Note: The remaining records are used as recovery records.

RECNUM - Number of Work File Records

`RECNUM=value` specifies the number of work file records (under Com-plete only) during an editor buffer pool cold start.

Value	Explanation
`100 - 65535`	Number of work file records.
`200`	This is the default value.

Notes:

1. This subparameter applies under Com-plete only.
2. This number determines the size of the work file.
3. The number of work file records is determined when the editor work file is created.

RWORK - Percentage of Work Records Used for Regular Logical Files

`RWORK=value` specifies the percentage of work records that is used for regular logical files during an editor buffer pool cold start.

Value	Explanation
`51 - 100`	Work records used (in percent).
`90`	This is the default value.

Note: The remaining records are used internally to release blocks from the buffer pool.

UTOUT - Timeout Value for Unchanged Buffer Pool Blocks

`UTOUT=`*`value`* specifies the timeout value for unchanged buffer pool blocks.

Value	Explanation
`1 - 32767`	Timeout value (in seconds).
`20`	This is the default value.

Note: An unchanged buffer pool block is written to the work file after the specified time interval has been exceeded and no free block is available.

Example of EDBP Parameter

```
EDBP=(DDNAME=EDFILE1,IMSG=ON,CHECK=ON)
```

Example of NTEDBP Macro

```
         NTEDBP DDNAME=EDFILE1,IMSG=ON,CHECK=ON
```

86 EDPSIZE - Size of Software AG Editor Auxiliary Buffer Pool

This Natural profile parameter determines the size of the Software AG Editor auxiliary buffer pool.

Possible settings	`48 -2097151`	Editor auxiliary buffer pool size in KB.
	`0`	No editor auxiliary buffer pool is used.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter must be used when the Software AG Editor runs in a z/OS Parallel Sysplex environment. It allows the Software AG Editor to be run without a Software AG Editor (local or global) buffer pool.
2. Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.
3. No Software AG Editor work file is required for the auxiliary buffer pool.
4. When the auxiliary buffer pool is used, the Software AG Editor's recovery function is not available.
5. If `EDPSIZE` is not zero, an auxiliary buffer pool is allocated and used although a (local or global) Software AG Editor buffer pool is defined with the `BPI` profile parameter or the `NTBPI` macro.
6. For further information on the Software AG Editor, see *Operating the Software AG Editor* in the *Operations* documentation.

87

EJ - Page Eject

This Natural profile and session parameter is used to specify whether a page eject is to be performed as a result of a logical page break, a break between program input and output, and the “normal end” message.

Possible settings	`ON`	A page eject is performed.		
	`OFF`	No page eject is performed. **Note:** This setting may be used to save paper during test runs where page ejects are not needed.		
Default setting	`ON`			
Dynamic specification	yes			
Specification within session	yes	Applicable statement:	`SET GLOBALS`	Parameter is evaluated at runtime.
		Applicable command:	`GLOBALS`	Parameter may be specified dynamically with the `GLOBALS` system command.
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.		

Notes:

1. Within a Natural session, the profile parameter `EJ` can be overridden by the session parameter `EJ`.
2. The `EJ` setting can in turn be overridden by an `EJECT` statement.
3. This parameter only applies to the first report (Report 0). For additional reports, the statement `EJECT` with report specification (*`rep`*) has to be used.

4. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.

Caution: The profile parameter `EJ` has a slightly different meaning when specified for a Natural session under CICS in batch mode (for example, `TTYPE=ASYL` or `TTYPE=BTCH`). For details, see *Asynchronous Natural Processing under CICS* in the *TP Monitor Interfaces* documentation.

88 EM - Edit Mask

With this session parameter, you can specify an edit mask for an input and/or output field that is used in one of the statements listed in the following table under *Applicable statements*.

Possible settings	See *EM Parameter Syntax*.	
Default setting	none	
Applicable statements	`FORMAT`	Parameter may be specified dynamically with the `FORMAT` statement. **Note:** Only the setting `EM=OFF`, but no actual edit mask specification is admissible.

	`DEFINE DATA`
	`DISPLAY`
	`INPUT`
	`PRINT`
	`PROCESS PAGE/PROCESS PAGE`
	`UPDATE`
	`WRITE`
		Parameter may be specified at statement level and/or at element level.

	`MOVE EDITED`	Parameter may be specified at element level.
Applicable command	none	

Notes:

1. For information on Unicode edit masks, see session parameter `EMU`.
2. The parameter `EM` can also be used with U format fields. For information on Unicode format, see *Unicode and Code Page Support in the Natural Programming Language*, *Session Parameters*, `EMU`, `ICU`, `LCU`, `TCU` versus `EM`, `IC`, `LC`, `TC`.
3. See also *Edit Masks - EM Parameter* in the *Programming Guide*.

The following topics are covered below:

EM Parameter Syntax

For input fields, values must be entered exactly matching the edit mask. If you would like to display the edit mask for an input field, the field should be defined as modifiable (`AD=M`).

For a database field, a default edit mask may have been defined in the DDM. If you specify with the `EM` parameter an edit mask for a database field, this edit mask specified will be used instead of any default edit mask which may be defined for the field in the DDM.

If you specify `EM=OFF` for a field, no edit mask will be used for the field, not even one that may be defined in the DDM.

At statement level of a `DISPLAY`, `FORMAT`, `INPUT` or `WRITE` statement, no detail field edit mask may be specified, except `EM=OFF`.

An edit mask overrides any settings for the session parameters `AL`, `NL` and `SG`.

The characters `9`, `H`, `X` and `Z` represent significant print positions in numeric (`9,Z`), hexadecimal (`H`), and alphanumeric (`X`) edit masks. For the difference between `9` and `Z`, see ***Edit Masks for Numeric Fields***, below.

Examples

```
DISPLAY AA(EM=OFF) AB(EM=XX.XX)
WRITE SALARY (EM=ZZZ,ZZ9)
```

You may replace a sequence of the same significant characters with a numeric notation, such as `x(8)` for `xxxxxxxx`. The following examples demonstrate the abbreviated notation which may be used for the significant characters of numeric (`Z,9`), hexadecimal (`H`), alphanumeric (`X`) and date (`N,L`) edit masks:

```
EM=9(4)-9(5)          is equivalent to: EM=9999-99999
EM=H(10)              is equivalent to: EM=HHHHHHHHHH
EM=X(6)..X(3)         is equivalent to: EM=XXXXXX..XXX
EM=YYYY-L(8)-DD-N(8) is equivalent to: EM=YYYY-LLLLLLLL-DD-NNNNNNNN
```

Blanks in Edit Masks

Blanks behind the equal sign (=) of the `EM` parameter are not allowed (for example: `EM=<blank>XXX`).

Blanks within an edit mask are represented by the character on your keyboard that in hexadecimal code corresponds to `H'20'` (ASCII) or `H'5F'` (EBCDIC), that is, the character `^` (or ¬).

Default Edit Masks

If no edit mask is specified for a field, a default edit mask is assigned to the field depending on the field format:

Field Format	Default Edit Mask
A	`X`
B	`H`
N, P, I	`Z9`
F	scientific representation
D	depends on default date format (as set with the profile parameter `DTFORM`)
T	`HH:II:SS`
L	blank / `X`

Edit Masks for Numeric Fields

An edit mask specified for a field of format N, P, I, or F must contain at least one `9` or `Z`.

If more `9`s or `Z`s exist than the number of positions contained in the field value, the number of print positions in the edit mask will be adjusted to the number of digits defined for the field value.

If fewer `9`s or `Z`s exist, the high-order digits before the decimal separator and/or low-order digits after the decimal separator will be truncated.

The following topics are covered below:

- Characters for the Definition of Numeric Edit Masks
- Sign Characters
- Literal Leading Characters
- Literal Insertion and Trailing Characters
- Trailing Sign Characters
- Examples of Numeric Edit Masks

Characters for the Definition of Numeric Edit Masks

Character	Function
`9`	Position to be displayed (one digit of the field value).
. (period)	The first period inserted is used as a decimal separator. Subsequent periods are treated as literal characters. **Note:** At this point, the period represents the sign currently defined as decimal separator character. If another character is chosen (for example, a comma) with the session or profile parameter `DC`, this character is to be used instead.
`Z`	Zero suppression for leading zeros. This is the default for numeric fields. The letter `Z` may be repeatedly specified to represent floating zero suppression. `Z` must not be specified to the right of the decimal separator character. A zero value may be displayed as blanks using all `Z`s in the edit mask (see also the session parameter `ZP`).

The `9`s or `Z`s can be preceded by one or more other characters.

Sign Characters

If the first character before the `9`s or `Z`s is `+`, `-`, `S` or `N`, a sign may be displayed:

Character	Function
`+`	A floating sign is to be displayed preceding (leading sign character) or following (trailing sign character) the number. The sign may be generated as a plus or minus depending on the value of the field.
`-`	A floating minus is to be displayed preceding (leading sign character) or following (trailing sign character) the number if the value of the field is negative.
`S`	A sign is to be displayed to the left of the column. A plus sign is displayed for a positive value and a minus sign is displayed for a negative value.
`N`	A minus sign is to be displayed to the left of the column if the value of the field is negative.

Literal Leading Characters

Any number of literal leading characters can appear before the first displayable position (as indicated by `Z` or `9`). These must follow any sign character. If there is no sign character and the first literal leading character is `+`, `-`, `S` or `N`, it must be enclosed in apostrophes. If a literal leading character is `H`, `X`, `Z` or `9`, it must be enclosed in apostrophes.

The first literal leading character specified will appear in the output only if the value contains leading zeros and the edit mask is defined with Z (leading zero suppression). This character will then be used as a filler character displayed instead of a blank for leading zeros. Subsequent literal leading characters will be displayed as they are input.

Literal Insertion and Trailing Characters

Literal insertion and trailing characters can also be used. The symbol (`^`) can be used to represent a leading, inserted, or trailing blank. By enclosing significant characters (`9`, `H`, `Z`, `X`) in apostrophes, it is possible to use any characters as leading, insertion, or trailing characters. Insignificant edit mask characters need not be enclosed in apostrophes. Within the same edit mask notation, it is possible to have groups of leading, insertion, and/or trailing character strings, some of which are bounded by apostrophes and some of which are not.

Trailing Sign Characters

A trailing sign character can be specified for numeric edit masks by using the + or - character as the last character in the edit mask. A + will produce a trailing + or - sign depending on the value of the field. A - will produce a trailing space or - sign depending on the value of the field. If a leading and trailing sign are specified in the edit mask, both will be produced.

Examples of Numeric Edit Masks

The table below lists the results obtained from the original values shown at the top of each column as they are output without editing mask. All values used as column headings represent format N fields. The lines below the top column represent the formats obtained using the different editing masks:

Value	`0000.03 (N4.2)`	`-0054 (N4)`	`+0087 (N4)`	`0962 (N4)`	`1830 (N4)`
Edit Mask					
`EM=9.9`	0.0	4.	7.	2.	0.
`EM=99`	00	54	87	62	30
`EM=S99`	+00	-54	+87	+62	+30
`EM=+Z9`	+0	-54	+87	+62	+30
`EM=-9.99`	0.03	-4.	7.	2.	0.
`EM=N9`	0	-4	7	2	0
`EM=*9.99`	0.03	4.	7.	2.	0.
`EM=Z99`	00	54	87	962	830
`EM=*EURZZ9.9`	EUR**0.0	EUR*54.	EUR*87.	EUR962.	EUR830.
`EM=999+`	000+	054-	087+	962+	830+
`EM=999-`	000	054-	087	962	830
`IC=$ EM=ZZZ.99`	$.03	$54.	$87.	$962.	$830.
`EM=H(6)`					
- *ASCII:*	303030303033	30303574	30303837	30393632	31383330
- *EBCDIC:*	F0F0F0F0F0F3	F0F0F5D4	F0F0F8F7	F0F9F6F2	F1F8F3F0

By combining edit masks with the parameters `IC` and `TC`, negative numbers can be displayed in varying formats using a `DISPLAY` statement.

Edit Masks for Alphanumeric Fields

An alphanumeric edit mask which is only to be used with A format fields must contain at least one `X` which represents a character to be displayed. An `H` as the first character designates a **hexadecimal edit mask**. A blank is represented by a (^) symbol. All other characters except closing parentheses are permissible including leading, trailing, and insertion characters. It is also possible to specify leading, insertion, or trailing characters enclosed within apostrophes. If the character `X`, a closing parenthesis, or a quotation mark is specified as an insertion character, it must be enclosed within apostrophes.

If leading characters are used before the first displayable position `X` of an alphanumeric edit mask, the first of these leading characters will not be displayed, but is used as filler character and replaces all leading blanks in the alphanumeric output field.

Example:

```
DEFINE DATA LOCAL
1 #X (A4)  INIT <'  34'>
END-DEFINE
WRITE #X (EM=*A:X:)
   6X #X (EM=*A:XX:)
   6X #X (EM=*A:XXX:)
   6X #X (EM=*A:XXXX:)
   6X #X (EM=1234XXXX5678)
END
```

Output Produced:

```
A:*:      A:**:      A:**3:      A:**34:      23411345678
```

Trailing characters which immediately follow the last permissible print position will be displayed.

If the number of positions specified with the mask is smaller than the field length, the overhanging field content is not displayed.

If the number of positions specified with the mask is higher than the field length, the mask is truncated on the first overhanging position.

Example:

```
DEFINE DATA LOCAL
1 #TEXT  (A4) INIT <'BLUE'>
END-DEFINE
WRITE #TEXT (EM=X-X-X)        /* 'B-L-U', 3 bytes of field only.
WRITE #TEXT (EM=X-X-X-X-X)    /* 'B-L-U-E-', with truncated mask.
END
```

Example of Alphanumeric Edit Masks

The following program lists the alphanumeric edit masks for a field that is defined with format/length A4 and contains the value `BLUE`.

```
** Example 'EMMASK1': Edit mask
************************************************************************
DEFINE DATA LOCAL
1 #TEXT  (A4)
END-DEFINE
*
ASSIGN #TEXT = 'BLUE'
WRITE NOTITLE 'MASK 1:' 5X #TEXT (EM=X.X.X.X)
      /       'MASK 2:' 5X #TEXT (EM=X^X^X^X)
      /       'MASK 3:' 5X #TEXT (EM=X--X--X)
      /       'MASK 4:' 5X #TEXT (EM=X-X-X-X-X)
      /       'MASK 5:' 5X #TEXT (EM=X' 'X' 'X' 'X)
      /       'MASK 6:' 5X #TEXT (EM=XX....XXX)
      /       'MASK 7:' 5X #TEXT (EM=1234XXXX)
END
```

Output of Program `EMMASK1`:

```
MASK 1:     B.L.U.E
MASK 2:     B L U E
MASK 3:     B--L--U
MASK 4:     B-L-U-E-
MASK 5:     B L U E
MASK 6:     BL....UE
MASK 7:     234BLUE
```

Edit Masks for Binary Fields - Format B

Edit masks for binary fields may be set using `X` or `H` notation. For binary fields, the `X` notation is supported as if `H` had been specified instead of `X`.

Hexadecimal Edit Masks

If the character `H` is specified as the first character in an edit mask, the content of an alphanumeric or numeric field will be displayed in hexadecimal format. Each `H` represents two print positions that will occur for each byte in the source field. Characters other than `H` serve as insertion or trailing characters in the mask. The number of positions to be displayed will be adjusted to the length of the edit mask if the mask is shorter than the field. The length of the edit mask will be adjusted to the length of the field if the field length is shorter than the edit mask.

Insertion or trailing characters may be optionally specified bounded by apostrophes.

All fields displayed with a hexadecimal edit mask are treated as alphanumeric. Therefore, if the edit mask is shorter than the field to be edited, numeric or alphanumeric positions will be displayed from left to right disregarding any decimal separator positions.

If a hexadecimal edit mask is used as an input edit mask, every `0-9`, `a-f`, `A-F`, blank and hex zero are accepted as a hex digit.

Note: Blank and hex zero are regarded as `0` and a lower-case letter (`a-f`) is regarded as an upper-case letter.

Edit Mask Examples for Hexadecimal Fields:

The tables below list the hexadecimal edit masks with results obtained from the original fields and values shown above each column. All numeric values (`-10`, `+10`, `01`) to which edit masks have been applied originated in fields defined with N2 format. The alphanumeric value `AB` originated from a field defined with format/length A2.

ASCII:

Value =>	AB	-10	+10	01
EM=HH	4142	3170	3130	3031
EM=H^H	41 42	31 70	31 30	30 31
EM=HH^H	4142	3170	3130	3031
EM=H-H	41-42	31-70	31-30	30-31
EM=H	41	31	31	30

EBCDIC:

Value =>	AB	-10	+10	01
EM=HH	C1C2	F1D0	F1F0	F0F1
EM=H:H	C1 C2	F1 D0	F1 F0	F0 F1
EM=HH:H	C1C2	F1D0	F1F0	F0F1
EM=H-H	C1-C2	F1-D0	F1-F0	F0-F1
EM=H	C1	F1	F1	F0

Example Program Using Hexadecimal Edit Masks:

```
** Example 'EMMASK2': Edit mask
************************************************************************
DEFINE DATA LOCAL
1 #TEXT1 (A2)
1 #TEXT2 (N2)
END-DEFINE
*
ASSIGN #TEXT1 = 'AB'
ASSIGN #TEXT2 =  10
*
WRITE NOTITLE
        'MASK (EM=HH)  :' 18T #TEXT1 (EM=HH)    30T #TEXT2 (EM=HH)
      / 'MASK (EM=H^H) :' 18T #TEXT1 (EM=H^H)   30T #TEXT2 (EM=H^H)
      / 'MASK (EM=HH^H):' 18T #TEXT1 (EM=HH^H)  30T #TEXT2 (EM=HH^H)
      / 'MASK (EM=H-H) :' 18T #TEXT1 (EM=H-H)   30T #TEXT2 (EM=H-H)
      / 'MASK (EM=H)   :' 18T #TEXT1 (EM=H)     30T #TEXT2 (EM=H)
END
```

Output of Program `EMMASK2` (ASCII):

```
MASK (EM=HH)  :  4142        3130
MASK (EM=H^H) :  41 42       31 30
MASK (EM=HH^H):  4142        3130
MASK (EM=H-H) :  41-42       31-30
MASK (EM=H)   :  41          31
```

Output of Program `EMMASK2` (EBCDIC):

```
MASK (EM=HH)  :  C1C2        F1F0
MASK (EM=H^H) :  C1 C2       F1 F0
MASK (EM=HH^H):  C1C2        F1F0
MASK (EM=H-H) :  C1-C2       F1-F0
MASK (EM=H)   :  C1          F1
```

Edit Masks for Date and Time Fields - Formats D and T

In edit masks for fields which are defined with format D (date) or T (time), the characters described in the following sections can be specified.

- Date - Format D, and Time - Format T
- Syntactical Restrictions for Date Characters
- Hints for Input Edit Mask
- Hints for Week Display (WW or ZW) in Output Edit Mask
- Time - Format T - only
- Examples of Date and Time Edit Masks

Date - Format D, and Time - Format T

Character	Usage
`DD`	Day.
`ZD`	Day, with zero suppression.
`MM`	Month.
`ZM`	Month, with zero suppression.
`YYYY`	Year, 4 digits (see the section *Hints for Input Edit Mask*).
`YY`	Year, 2 digits (see the section *Hints for Input Edit Mask*).
`Y`	Year, 1 digit. Must not be used for input fields.
`WW`	Number of week (see the sections *Hints for Input Edit Mask* and *Hints for Week Display in Output Edit Mask*).
`ZW`	Number of week, with zero suppression (see the sections *Hints for Input Edit Mask* and *Hints for Week Display in Output Edit Mask*).
`JJJ`	Julian day.
`ZZJ`	Julian day with zero suppression.

Character	Usage
`NN...` or `N(n)`	Name of day (language-dependent). The maximum length is determined by the number of `N`s or by *`n`*. If the name is longer than the maximum length, it will be truncated; if it is shorter, the actual length of the name will be used.
`O`	Number of week day. The profile parameter `DTFORM` determines whether Monday or Sunday is considered the first day of the week. With `DTFORM=U`: (Sunday = 1, Monday = 2, etc.). With `DTFORM=other`: (Monday = 1, Tuesday = 2, etc.).
`LL...` or `L(n)`	Name of month (language-dependent). The maximum length is determined by the number of L characters or by *`n`*. If the name is longer than the maximum length, it will be truncated; if it is shorter, the actual length of the name will be used.
`R`	Year in Roman numerals (maximum 13 digits). Must not be used for input fields. The upper limit for displayable year values is 2887.

Syntactical Restrictions for Date Characters

For *Input* and *Output* edit masks, you *may not* use the following:

text			characters		
month	with	month name	`MM` or `ZM`	with	`LL` or `L(n)`
day name	with	week day number	`NN` or `N(n)`	with	`O`

For *Input* edit masks, you *may not* use the following:

text			characters		
1-digit year	nor	a year in Roman numerals	`Y`	nor	`R`
Day	without	month or month name	`DD` or `ZD`	without	`MM` or `ZM` or `LL` or `L(n)`
Week	without	year	`WW` or `ZW`	without	`YYYY` or `YY`
Month	without	year	`MM` or `ZM`	without	`YYYY` or `YY`
Julian day	without	year	`JJJ` or `ZZJ`	without	`YYYY` or `YY`
Day name	without	week	`NN` or `N(n)`	without	`WW` or `ZW`
Week day number	without	week	`O`	without	`WW` or `ZW`
Julian day	with	month	`JJJ` or `ZZJ`	with	`MM` or `ZM`
Julian day	with	week	`JJJ` or `ZZJ`	with	`WW` or `ZW`
Month	with	week	`MM` or `ZM`	with	`WW` or `ZW`

Hints for Input Edit Mask

The range of valid year values (`YYYY`) is `1582 - 2699`. If the profile parameter `MAXYEAR` is set to `9999`, the range of valid year values is `1582 - 9999`.

If only year (`YY` or `YYYY`) but no month or day is specified within an input edit mask, the values for month and day will both be set to `01`. If only year (`YY` or `YYYY`) and month (`MM`) but no day is specified within an input edit mask, the value for day will be set to `01`.

If a 2-digits year (`YY`) is used, the century used to fill up the year representation is the current century by default. However, this does not apply when a Sliding or Fixed Window is set. For more details, refer to profile parameter `YSLW` in the *Parameter Reference* documentation.

If a week number (`WW` or `ZW`) but no number of week day (`O`) or name of day (`NN...`) is specified, the first day of the week is assumed.

Hints for Week Display (WW or ZW) in Output Edit Mask

When `DTFORM=U` (USA format) is set, the week starts on Sunday; whereas for all other `DTFORM` settings the first weekday is Monday. Whether a week is week 52/53 of the old year or week 01 of the new year depends on which year contains more days of the week. In other words, if Thursday (Wednesday for `DTFORM=U`) of that week is in the previous year, the week belongs to the previous year; if it is in the next year, the week belongs to the next year.

If the number of week (`WW` or `ZW`) and a year representation (`YYYY` or `YY` or `Y`) is in the same edit mask, the display for year always corresponds to the week number, regardless of the year in the underlying date field.

Example:

```
DEFINE DATA LOCAL
1 D (D)
END-DEFINE
MOVE EDITED '31-12-2003' TO D(EM=DD-MM-YYYY)
DISPLAY D(EM=DD-MM-YYYY_N(10)) D(EM=DD-MM-YYYY/WW)
END
```

Although the underlying date is the 31 Dec. 2003, when the week number `WW` is contained in the edit mask, it displays as:

```
D                  D
--------------------  -------------
31-12-2003_Wednesday  31-12-2004/01
```

Time - Format T - only

Character	Usage
`T`	Tenths of a second.
`SS`	Seconds.
`ZS`	Seconds, with zero suppression.
`II`	Minutes.
`ZI`	Minutes, with zero suppression.
`HH`	Hours.
`ZH`	Hours, with zero suppression.
`AP`	AM/PM element.

Examples of Date and Time Edit Masks

```
** Example 'EMDATI': Edit mask for date and time variables
************************************************************************
*
WRITE NOTITLE
  'DATE INTERNAL :' *DATX (DF=L) /
  '              :' *DATX (EM=N(9)' 'ZW.'WEEK 'YYYY)  /
  '              :' *DATX (EM=ZZJ'.DAY 'YYYY)        /
  '     ROMAN    :' *DATX (EM=R)  /
  '     AMERICAN :' *DATX (EM=MM/DD/YYYY)      12X 'OR  ' *DAT4U /
  '     JULIAN   :' *DATX (EM=YYYYJJJ)         15X 'OR  ' *DAT4J /
  '     GREGORIAN:' *DATX (EM=ZD.''L(10)''YYYY) 5X 'OR  ' *DATG ///
*
  'TIME INTERNAL :' *TIMX                      14X 'OR  ' *TIME /
  '              :' *TIMX (EM=HH.II.SS.T) /
  '              :' *TIMX (EM=HH.II.SS' 'AP) /
  '              :' *TIMX (EM=HH)
END
```

Output of Program `EMDATI`:

```
DATE INTERNAL : 2005-01-12
              : Wednesday  2.WEEK 2005
              :  12.DAY 2005
     ROMAN    : MMV
     AMERICAN : 01/12/2005            OR   01/12/2005
     JULIAN   : 2005012               OR   2005012
     GREGORIAN: 12.January2005        OR   12January  2005


TIME INTERNAL : 16:04:14              OR   16:04:14.8
              : 16.04.14.8
              : 04.04.14 PM
              : 16
```

Edit Masks for Logical Fields - Format L

For fields of format L (logical fields), edit masks can be defined as follows:

(EM=[*false-string*/]*true-string*)

The `false-string` must not be longer than 31 characters.

Example of Edit Masks for Logical Field

```
** Example 'EMLOGV': Edit mask for logical variables
************************************************************************
DEFINE DATA LOCAL
1 #SWITCH (L)  INIT <true>
1 #INDEX  (I1)
END-DEFINE
*
FOR #INDEX 1 5
  WRITE NOTITLE #SWITCH (EM=FALSE/TRUE) 5X 'INDEX =' #INDEX
  WRITE NOTITLE #SWITCH (EM=OFF/ON)     7X 'INDEX =' #INDEX
  IF #SWITCH
    MOVE FALSE TO #SWITCH
  ELSE
    MOVE TRUE TO #SWITCH
  END-IF
  /*
  SKIP 1
END-FOR
END
```

Output of Program `EMLOGV`:

```
TRUE      INDEX =    1
ON        INDEX =    1

FALSE     INDEX =    2
OFF       INDEX =    2

TRUE      INDEX =    3
ON        INDEX =    3

FALSE     INDEX =    4
OFF       INDEX =    4

TRUE      INDEX =    5
ON        INDEX =    5
```

89 EMFM - Edit Mask Free Mode

This Natural profile parameter is used to activate/deactivate the Edit Mask Free mode at session startup.

Possible settings	`ON`	Edit Mask Free Mode is activated.
	`OFF`	Edit Mask Free Mode is deactivated.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	Within a running Natural session, you may override this setting with the terminal control command `%FM+` or `%FM-`.

Notes:

1. The Edit Mask Free mode allows you to omit literals during input into a field with a numeric edit mask.
2. For additional information, see *Numeric Edit Mask Free Mode* in the `INPUT` statement description in the *Statements* documentation.

90

EMU - Unicode Edit Mask

With this session parameter, you can specify a Unicode edit mask for an input and/or output field that is used in one of the statements listed in the following table under *Applicable statements*.

Possible settings	The syntax of the session parameter `EMU` is identical to that of the session parameter `EM` (see ***EM Parameter Syntax***). **Note:** See also *Unicode Edit Masks - EMU Parameter* in the *Programming Guide*.	
Default setting	none	

Applicable statements	`DEFINE DATA`
	`DISPLAY`
	`INPUT`
	`PRINT`
	`WRITE`
	`MOVE EDITED`
	`PROCESS PAGE`
		Parameter may be specified at statement level and/or at element level.

Applicable command	none	

Notes:

1. Edit masks which are defined with `EMU` are kept in Unicode format so that the content is independent of the installed system code page.
2. For further information and an example, see also *Unicode and Code Page Support in the Natural Programming Language*, *Session Parameters*, section *EMU, ICU, LCU, TCU versus EM, IC, LC, TC*.

91 ENDBT - BACKOUT TRANSACTION at Session End

This Natural profile parameter determines whether or not an implicit `BACKOUT TRANSACTION` statement is to be issued at the end of the Natural session.

Possible settings	`ON`	Natural will issue an implicit `BACKOUT TRANSACTION` statement at session end.
	`OFF`	Natural will not issue an implicit `BACKOUT TRANSACTION` statement at session end.
	`ETDB`	Natural will issue an implicit `BACKOUT TRANSACTION` statement at session end only for the database specified with the profile parameter `ETDB`.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Other transaction processing related parameters: `ADAMODE` | `DBCLOSE` | `DBOPEN` | `ET` | `ETDB` | `ETEOP` | `ETIO` | `ETSYNC`

92 ENDMSG - Display Session-End Message

This Natural profile parameter is used to suppress the display the default message NAT9995 that is displayed at the end of the Natural session to indicate that the Natural session has been ended normally.

Possible settings	`ON`	Message NAT9995 will be displayed at the end of the session.
	`OFF`	Message NAT9995 will not be displayed at the end of the session.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Note: If a session back-end program is defined with the profile parameter `PROGRAM`, the `ENDMSG` profile parameter has no effect; the message text will then be passed to the back-end program in the parameter area and will not be displayed by Natural.

93

ES - Empty Line Suppression

With this session parameter, you can suppress the printing of empty lines generated by a `DISPLAY` or `WRITE` statement.

Possible settings	`ON`	A line resulting from a `DISPLAY` or `WRITE` statement which contains all blank values will not be printed. **Note:** This setting is particularly useful when displaying arrays (for example, multiple-value fields or fields contained within a periodic group) to avoid printing a large number of empty lines.
	`OFF`	Empty line suppression is disabled.
Default setting	`OFF`	
Specification within session	yes	
Applicable statements	`FORMAT`	
	`DISPLAY` `WRITE`	Parameter may be specified at statement level and/or at element level.
Applicable command	none	

Notes:

1. To achieve empty suppression for numeric values, the field must be specified with `ZP=OFF` and `ES=ON` in order to have null values printed as blanks. See also the session parameters `IS` and `ZP`.
2. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

Example:

```
DISPLAY (ES=ON) NAME CITY
```

94 ESCAPE - Ignore Terminal Commands %% and %.

This Natural profile parameter can be used to disable the terminal commands `%%` and `%.`.

Possible settings	`ON`	Enables the use of terminal commands `%%` and `%.`.
	`OFF`	The terminal commands `%%` and `%.` will be ignored; that is, it will not be possible to leave the currently active Natural program or the Natural session respectively by entering `%%` or `%.`.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

95 ESIZE - Size of User-Buffer Extension Area

This Natural profile parameter sets the size of the user-buffer extension area. It determines the size of the Natural source area which is used by the Natural editors.

Possible settings	`2 - 32767`	Size of buffer extension area in KB. **Note:** In a runtime environment (where the editors are not used), you can only set a value smaller than the default setting.
Default setting	`28`	
Dynamic specification	yes	
Specification within session	no	

The user-buffer extension area contains:

- the source code of the Natural object to be compiled,
- the table of currently active PA/PF keys,
- other tables and work areas internally used by Natural.

Notes:

1. In a production environment, Natural sources are not needed and the `ESIZE` value can therefore be reduced accordingly.
2. If this area is not large enough to contain the necessary information, the Natural error message NAT0886 is issued.

96 ET - Execution of END/BACKOUT TRANSACTION Statements

This Natural profile parameter specifies for which databases `END TRANSACTION` and `BACKOUT TRANSACTION` statements are to be executed.

Possible settings	`ON`	`END TRANSACTION` and `BACKOUT TRANSACTION` statements are executed for all databases which have been referenced since the beginning of the Natural session or since the last execution of an `END TRANSACTION` and `BACKOUT TRANSACTION` statement.
	`OFF`	`END TRANSACTION` and `BACKOUT TRANSACTION` statements are executed only for the databases affected by the transaction (and - if applicable - for the database to which transaction data are written).
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Note: Any updates to a database which are not executed under the control of Natural (that is, by native invocation of the database link routines) do not affect the Natural transaction logic.

Other transaction processing related parameters: `ADAMODE` | `DBCLOSE` | `DBOPEN` | `ENDBT` | `ETDB` | `ETEOP` | `ETIO` | `ETSYNC`

97 ETA - Error Transaction Program

This Natural profile parameter provides the name of the program which receives control if an error condition is detected during Natural program execution.

Possible settings	1 to 8 characters	Program name for error transaction.
	`' '` (blank)	With `ETA=' '`, no error transaction program is called.
Default setting	`' '` (blank)	
Dynamic specification	yes	
Specification within session	yes	
Application programming interface	`USR1041N`	`USR1041N`is a sample error transaction program delivered in source form. See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The setting of this parameter can be modified by a user program by way of assignment to the system variable `*ERROR-TA` or, if Natural Security is installed, within the Natural Security library profile; see *Components of a Library Profile* in the *Natural Security* documentation.
2. For further information, see *Using an Error Transaction Program* in the *Programming Guide*.

98 ETDB - Database for Transaction Data

This Natural profile parameter specifies the database in which transaction data, as supplied with an `END TRANSACTION` statement is to be stored.

Possible settings	`1 - 65535`, except `255`	Database ID. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see profile parameter `LFILE`.
	`0`	The transaction data is written to the database where the Natural Security system file (`FSEC`) is located. If `FSEC` is not specified, it is considered to be identical to the Natural system file `FNAT` (if Natural Security is not installed, the transaction data are written to the database where `FNAT` is located.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Other transaction processing related parameters: `ADAMODE` | `DBCLOSE` | `DBOPEN` | `ENDBT` | `ET` | `ETEOP` | `ETIO` | `ETSYNC`

99 ETEOP - Issue END TRANSACTION at End of Program

This Natural profile parameter determines whether or not an implicit `END TRANSACTION` statement is to be issued at the end of a Natural program (that is, before `NEXT` mode is reached).

Possible settings	`ON`	Natural will issue an implicit `END TRANSACTION` statement at the end of a Natural program.
	`OFF`	Natural will not issue any implicit `END TRANSACTION` statement at the end of a Natural program.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Other transaction processing related parameters: `ADAMODE` | `DBCLOSE` | `DBOPEN` | `ENDBT` | `ET` | `ETDB` | `ETIO` | `ETSYNC`

100 ETID - Adabas User Identification

This Natural profile parameter is used as an identifier for Adabas-related information; for example, for identification of data stored as a result of an `END TRANSACTION` statement.

Possible settings	1 - 8 characters	The setting is used as the user ID setting in an Adabas open call.
	`OFF`	Natural does not issue any Adabas open and close commands at the beginning of the Natural session. If, however, any `ETID` and/or `OPRB` specifications are present in Natural Security, these specifications are used in the subsequent open issued by Natural Security. This parameter setting is provided for use in conjunction with Natural Security to prevent Natural batch jobs that are sent at the same time from causing duplicate user ID settings in an Adabas open call during the initialization phase.
	`'  '` (blank)	If the `ETID` parameter is set to blanks, Natural does not issue any Adabas open and close commands; the `OPRB` parameter (if specified) and any `ETID` and `OPRB` specifications in Natural Security are ignored. In this case, you are recommended to set the Natural profile parameter `DBCLOSE` to `ON` to enforce a close command at session end. Otherwise, the user is not logged off from Adabas and the Adabas user queue element is not deleted. This may cause an overflow situation in the Adabas user queue.
Default setting	`*INIT-USER`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If the `ETID` setting is *not* the same as the setting of the Natural system variable `*INIT-USER`, Natural issues an Adabas open with the specified `ETID` setting (and `OPRB` setting, if specified) at the beginning of the Natural session; this open remains in effect until the end of the Natural session; any `ETID` and `OPRB` specifications in Natural Security are ignored.

2. If the `ETID` setting is the same as the setting of `*INIT-USER`, or if the `ETID` parameter is not specified, Natural issues an Adabas open with the `*INIT-USER` setting as `ETID` (and the `OPRB` setting, if specified) at the beginning of the Natural session. If any Natural Security logon (initial logon or any subsequent logon) would change the currently valid `ETID` or `OPRB` setting (due to the library-/user-specific `ETID` and `OPRB` specifications in Natural Security), Natural Security issues a new open with the new `ETID` and `OPRB` settings. If the settings would remain the same after a logon, Natural Security does not issue a new open.
3. `ETID` and `*INIT-USER` can be modified by user exit `NATUEX1` during session startup. See *NATUEX1 - User Exit for Authorization Control* in the *Operations* documentation.
4. For further `ETID` options available with `ETID=OFF` which can be set in Natural Security, see *Library and User Preset Values* in the *Natural Security* documentation.

101 ETIO - Issue END TRANSACTION upon Terminal I/O

This Natural profile parameter determines whether or not implicit `END TRANSACTION` statements are to be issued upon terminal I/O operations.

Possible settings	`ON`	Natural will issue an implicit `END TRANSACTION` statement whenever a terminal I/O occurs. **Note:** 1. Whenever a transaction monitor commits the associated databases because of a terminal I/O, all related databases are also committed. This is useful for the synchronization of database transactions. 2. Natural add-on products (except for Natural Security) may not function correctly with `ETIO=ON`.
	`OFF`	Natural will issue no implicit `END TRANSACTION` statements upon terminal I/O operations.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Other transaction processing related parameters: `ADAMODE` | `DBCLOSE` | `DBOPEN` | `ENDBT` | `ET` | `ETDB` | `ETEOP` | `ETSYNC`

102 ETRACE - External Trace Function

This Natural profile parameter is used to activate/deactivate the (normal) external trace function or the Generalized Trace Facility (GTF) offered under z/OS and TSO.

Caution: Do not use this parameter without prior consultation of Software AG Support.

Possible settings	`ON`	Activates the (normal) external trace function.
	`OFF`	Deactivates the (normal) external trace function.

	`(ON,GTF)`
	`(OFF,GTF)`
		Activates/deactivates the Generalized Trace Facility (GTF). The trace records are written to the GTF.

	`(ON,NOGTF)`
	`(OFF,NOGTF)`
		Activates/deactivates the (normal) external trace function.

	`(,GTF)`	Equivalent to `ETRACE=GTF`. Trace data is written to the GTF. `ON` or `OFF` is not altered.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	Within a Natural session, the terminal command `%TRE` can be used to activate/deactivate the external trace function, except GTF.

Notes:

1. The trace function is intended primarily for Software AG internal use for debugging purposes. It writes trace data to an external trace data set depending on the TP environment in which Natural is running.
2. In batch and TSO environments, a data set is required for the external trace (see also *CMTRACE - Optional Report Output for Natural Tracing* in the *Operations* documentation).

103 ETSYNC - Issue Syncpoint upon End of Transaction/Backout Transaction

This Natural profile parameter determines whether or not an implicit syncpoint is issued whenever an `END TRANSACTION` or `BACKOUT TRANSACTION` statement is to be issued.

Possible settings	`ON`	Natural issues an implicit syncpoint `COMMIT` whenever an `END TRANSACTION` statement is to be issued. Natural issues an implicit syncpoint `ROLLBACK` whenever a `BACKOUT TRANSACTION` statement is to be issued. This is useful for the synchronization of database transactions that are performed from within 3GL programs.
	`OFF`	Natural does not issue an implicit syncpoint when an `END TRANSACTION` or `BACKOUT TRANSACTION` statement is to be issued.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Notes:

To issue syncpoints, Natural uses

- Resource Recovery Services (RRS) under TSO and in batch mode to commit or rollback the unit of recovery,
- CICS commands `SYNCPOINT` and `SYNCPOINT ROLLBACK` under CICS,
- system service calls `CHECKPOINT` (`CHKP`) and `ROLLBACK` (`ROLB`) under IMS TM.

The processing sequence is as follows:

- an `END TRANSACTION` / `BACKOUT TRANSACTION` statement is issued to the database specified with the profile parameter `ETDB`,
- the syncpoint `COMMIT` / `ROLLBACK` is issued,
- `END TRANSACTION` or `BACKOUT TRANSACTION` statements are issued to the remaining databases.

Restrictions and Limitations:

- This functionality is available under the z/OS operating system
  - in batch mode,
  - under the TP monitor CICS,
  - under the TP monitor TSO,
  - under the TP monitor IMS TM in a non-message driven BMP (in all other environments under IMS TM, only a `ROLLBACK` is executed, but no `CHECKPOINT`).
- To synchronize Adabas transactions, the Adabas Transaction Manager (ATM) must be installed.
- For transactions in batch mode or under TSO that update data stored in a DB2 database, you must configure Natural for DB2 and/or your 3GL application to use the RRSAF interface.
- For transactions in batch mode that update data stored in a DL/I database, Resource Recovery Services are not supported due to a DL/I restriction. If, additionally, data stored in a DB2 database is updated in the same transaction, synchronization is performed by means of the DL/I synchronization mechanism.

  As a consequence, if data stored in an Adabas database is updated in addition to data stored in DB2 and DL/I databases, no synchronization is possible, not even if the Adabas Transaction Manager is installed.

Other transaction processing related parameters: `ADAMODE` | `DBCLOSE` | `DBOPEN` | `ENDBT` | `ET` | `ETDB` | `ETEOP` | `ETIO`

104 EXCSIZE - Size of Buffer for Natural Expert C Interface

This Natural profile parameter is obsolete and only accepted for compatibility reasons.

105 EXRSIZE - Size of Buffer for Natural Expert Rule Tables

This Natural profile parameter is obsolete and only accepted for compatibility reasons.

106 FAMSTD - Overwriting of Print and Work File Access Method Assignments

This Natural profile parameter controls the automatic overwriting of print and work file access method assignments during session initialization according to the data set definition in the job control.

Note: See also the `AM` subparameter of the macros `NTPRINT` and `NTWORK`.

Possible settings	`ON`	All print and work file data sets are automatically assigned to the batch access method `AM=STD` if the logical data set name (defined by the `DEST` subparameter) is defined by job control.
	`OFF`	Automatic print and work file assignment to `AM=STD` is done only if the file is not assigned to another access method; for example, `AM=NAF`. **Note:** If `AM=OFF` is specified, no automatic assignment is done. Specify `AM=0` if you want to reset the access method type and to allow automatic assignment.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

107 FC - Filler Character for INPUT Statement

This Natural profile parameter is used to specify the default filler character to be used for fields displayed by an `INPUT` statement.

Possible settings	any character	Default filler character.
Default setting	`X'00'`	For TTY, web I/O (with the Natural Web I/O Interface) or batch mode, the default setting is `X'40'`, i.e. blank in hexadecimal format.
Dynamic specification	yes	
Specification within session	no	

Notes:

1. The default filler character is used to pre-fill non-protected input fields (field attribute specification `AD=A`) when fields are written to a terminal by an `INPUT` statement.
2. For modifiable input fields (field attribute specification `AD=M`), it is used to fill the rest of the field.

108 FC - Filler Character for DISPLAY Statement

With this session parameter, you specify the filler character which will appear on either side of a heading produced by a `DISPLAY` statement across the full column width.

Possible settings	any character	Filler character for individual headings.
Default setting	blank	
Specification within session	yes	
Applicable statements	`DISPLAY` `FORMAT`	
Applicable command	none	

Notes:

1. `FC` only applies if the column width is determined by the field length and not by the header (see also the session parameter `HW`); otherwise the `FC` setting will be ignored.
2. Unlike the `GC` parameter, which applies to headings across a group of columns, the `FC` parameter applies to individual columns.

Example:

```
DISPLAY (FC=*)
```

109

FCDP - Filler Character for Dynamically Protected Input Fields

This Natural profile and session parameter can be used to suppress the display of filler characters for input fields that have been made write-protected dynamically (that is, to which the attribute `AD=P` has been assigned via an attribute control variable).

Possible settings	`ON`	Dynamically protected input fields are displayed filled with filler characters. This may suggest to the users that they could enter something in the fields.
	`OFF`	Dynamically protected input fields are displayed filled with blanks.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Depending on the setting of the `FCDP` parameter, dynamically protected input fields are displayed filled either with blanks or with the defined filler characters.
2. Within a Natural session, the profile parameter `FCDP` can be overridden by the session parameter `FCDP`.

Example:

```
DEFINE DATA LOCAL
   1 #FIELD1 (A5)
   1 #FIELD2 (A5)
   1 #CVAR1  (C) INIT <(AD=P)>
   1 #CVAR2  (C)
   END-DEFINE
   *
   INPUT #FIELD1 (AD=Y'_' CV=#CVAR1)  /* field is protected
         #FIELD2 (AD=Y'_' CV=#CVAR2)  /* field is not protected
   ...
   END
```

Execution of the above program will display the following:

`FCDP=ON`:

```
#FIELD1 _____ #FIELD2 _____
```

`FCDP=OFF`:

```
#FIELD1       #FIELD2 _____
```

110 FDIC - Predict System File

This Natural profile parameter defines the database ID, file number, password, cipher key and read-only flag for the Predict system file (FDIC), which Predict uses to retrieve and/or store data.

Possible settings	See *FDIC Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	If you specify the `FDIC` parameter dynamically in conjunction with any of the parameters `DBID`, `FNR`, `SYSPSW` and `SYSCIP`, you must specify the `FDIC` parameter *after* any of these other parameters.
Specification within session	no	

Notes:

1. In a remote development environment, a Development Server File is used instead; see the *SPoD - Natural's Single Point of Development* and the *Natural Development Server* documentation.
2. For information on system files, refer to *Natural System Files* in the *Natural System Architecture* documentation.

FDIC Parameter Syntax

The parameter syntax is as follows:

```
FDIC=(database-ID,file-number,password,cipher-key,RO)
```

Where:

Syntax Element	Value	Explanation
`database-ID`	`1 - 65535`, except `255`	Database identification of the database in which the Predict system file is located. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see Natural profile parameter `LFILE`.
`file-number`	`1 - 65535`	File number of the database file in which the Predict system file is located.
`password`	1 - 8 characters	Password for the Predict system file. **Note:** 1. A password is only required if the Predict system file has been password-protected using the Adabas security feature. 2. When Natural is used with VSAM system files, the password is used to specify the logical name (DD or DLBL) of the system file as defined to VSAM. Example: `FDIC=(10,5,SYSVSAM)`
`cipher-key`	1 - 8 numeric characters	Cipher key for the Predict system file. **Note:** A cipher key is only required if the Predict system file has been ciphered using the Adabas security feature.
`RO`	-	Read-only option. **Note:** `RO` indicates that the Predict system file is “read-only” and is only specified if modifications on the file are to be disabled.

Note: **Default values:** If any subparameter of the `FDIC` setting is not specified, the corresponding setting of the profile parameter `DBID`, `FNR`, `SYSPSW` or `SYSCIP` applies for the Predict system file.

Examples:

```
FDIC=(10,5,PASSW1,12345678)
FDIC=(1,200,,12345678)
FDIC=(1,5)
FDIC=(,5)
```

111 FL - Floating Point Mantissa Length

With this session parameter, you specify the mantissa length of a floating point variable during input or output.

Possible settings	`1 - 16`	Mantissa length. **Note:** The total length is `FL + 6` for sign, exponent, and decimal character.
Default setting	none	
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`INPUT`
	`PRINT`
	`WRITE`	

Applicable command	none	

Example:

```
DISPLAY FL=5     ->     +1.2345E+03
```

112

FNAT - Natural System File for System Programs

This Natural profile parameter defines the database ID, file number, password, cipher key and read-only flag for the Natural system file for Natural system programs (FNAT).

Possible settings	See *FNAT Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	**Note:** If you specify the `FNAT` parameter dynamically in conjunction with any of the profile parameters `DBID`, `FNR`, `SYSPSW`, `SYSCIP` or `ROSY`, you must specify the `FNAT` parameter *after* any of these parameters.
Specification within session	no	

Notes:

1. The Natural system file is the database file from which all Natural system programs are retrieved and upon which all system commands operate. Error texts and Natural help information are also contained in this system file.
2. For information on system files, refer to *Natural System Files* in the *Natural System Architecture* documentation.

FNAT Parameter Syntax

The parameter syntax is as follows:

FNAT=(*database-ID*,*file-number*,*password*,*cipher-key*,RO)

Where:

Syntax Element	Value	Explanation
database-ID	`1 - 65535`, except `255`	Database identification of the database in which the Natural system file is located. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see Natural profile parameter `LFILE`.
file-number	`1 -65535`	File number of the database file in which the Natural system file is located.
password	1 - 8 characters	Password for the Natural system file. **Note:** 1. A password is only required if the Natural system file has been password-protected using the Adabas security feature. 2. When Natural is used with VSAM system files, the password is used to specify the logical name (DD or DLBL) of the system file as defined to VSAM. Example: `FNAT=(22,5,SYSVSAM)`
cipher-key	8 numeric characters	Cipher key for the Natural system file. **Note:** A cipher key is only required if the Natural system file has been ciphered using the Adabas security feature.
`RO`	-	Read-only option. **Note:** 1. `RO` indicates that the Natural system file is “read-only”. 2. `RO` is only specified if modifications on the file are to be disabled.

Notes:

1. **Default values:** If any subparameter of the `FNAT` setting is not specified, the corresponding setting of the profile parameter `DBID`, `FNR`, `SYSPSW`, `SYSCIP` or `ROSY` applies for the Natural system file for system programs.
2. If you reorganize an `FNAT` file or if you unload/load data from/to the `FNAT` file (for example, by using `ADAULD/ADALOD`), you must specify `USERISN=YES` for the ADALOD utility.

Examples:

```
FNAT=(,8)
FNAT=(22,5,PASSW2)
```

113 FNR - Default File Number of Natural System Files

This Natural profile parameter identifies the default number of the database file in which the Natural system files (`FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF`, `FREG`) are located.

Possible settings	`1 - 65535`	Default file number.
Default setting	none	
Dynamic specification	yes	**Note:** If you specify the profile parameter `FNR` dynamically in conjunction with any of the individual profile parameters which define the system files (`FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF`, `FREG`), you must specify the `FNR` parameter *before* any individual system file parameter.
Specification within session	yes	

Notes:

1. The default file number applies to all Natural system files for which no individual database file numbers are specified.
2. File numbers for individual system files can be specified with the *`file-number`* subparameter of the profile parameters `FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF` and `FREG`.

Examples:

Example 1:

```
FNR=5,DBID=10,FUSER=(,8)
```

This example assigns the user-program system file to File 8 on Database 10. All other system files are assigned to File 5 on Database 10.

Example 2:

```
FUSER=(,8),FNR=5,DBID=10
```

This example assigns all system files to File 5 on Database 10.

114 FPROF - Natural System File for Parameter Profiles

This Natural profile parameter can be used to specify a system file for parameter profiles (FPROF).

Possible settings	See *FPROF Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	**Note:** If you specify the profile parameter `FPROF` dynamically in conjunction with any of the profile parameters `DBID`, `FNR`, `SYSPSW` or `SYSCIP`, you must specify the `FPROF` parameter *after* any of these parameters.
Specification within session	no	

Notes:

1. The system file `FPROF` is used to read parameter profiles specified by the profile parameter `PROFILE`, provided no database information is supplied as subparameter of `PROFILE`.
2. If `FPROF` is not defined, the system file `FNAT` is used as the system file for parameter profiles.
3. Parameter profiles can be maintained with the SYSPARM utility.
4. For information on system files, refer to *Natural System Files* in the *Natural System Architecture* documentation.

FPROF Parameter Syntax

The parameter syntax is as follows:

```
FPROF=(database-ID,file-number,password,cipher-key,RO)
```

Where:

Syntax Element	Value	Explanation
`database-ID`	`1 - 65535`, except `255`	Database identification of the database in which the Natural system file for parameter profiles is located. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see Natural profile parameter `LFILE`.
`file-number`	`1 -65535`	File number of the database file in which the Natural system file for parameter profiles is located.
`password`	1 - 8 characters	Password for the Natural system file `FPROF`. **Note:** 1. A password is only required if the Natural system file for parameter profiles has been password-protected using the Adabas security feature. 2. When Natural is used with VSAM system files, the password is used to specify the logical name (DD or DLBL) of the system file as defined to VSAM. Example: `FPROF=(10,33,SYSVSAM)`
`cipher-key`	8 characters	Cipher key for the Natural system file `FPROF`. **Note:** A cipher key is only required if the Natural system file for parameter profiles has been ciphered using the Adabas security feature.
`RO`	-	Read-only option. **Note:** 1. `RO` indicates that the Natural system file for parameter profiles is “read-only”. 2. `RO` is only specified if modifications on the system file for parameter profiles are to be disabled. This would mean, for example, that no profiles could be stored on this file by means of the SYSPARM utility.

Note: **Default values:** If any subparameter of profile parameter `FPROF` is not specified, the corresponding setting of the profile parameter `DBID`, `FNR`, `SYSPSW` or `SYSCIP` applies to the system file for parameter profiles.

Example:

```
FPROF=(10,56)
```

In this example, the system file `FPROF` is located on database file 56 in database 10.

115 FREEGDA - Release GDA in Utility Mode

This Natural profile parameter controls whether current user global data area (GDA) and application-independent variables (AIV) are to be reset or not when a utility is invoked in utility mode (see *Utility Activation* in the *Utilities* documentation), that is, by using the direct command that corresponds to the utility's name.

Possible settings	`ON`	The current user GDA and AIV variables are reset before a utility is started. **Note:** This behavior corresponds to the previous situation when the utility was invoked using the system command `LOGON` *`library-name`*.
	`OFF`	The current user GDA and AIV variables are preserved when a utility is started. **Note:** This will increase the data size correspondingly and may lead to thread problems under certain operating systems.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

116 FREG - Natural Registry System File

This Natural profile parameter is used to specify the database ID, file number, password and cipher key for the Natural Registry system file.

The Natural Registry system file is the database file which is used to store the registry information for Natural sessions which are started with a value > 0 for the Natural profile parameter `UCONMAX`.

Possible settings	See *FREG Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	**Note:** If you specify the `FREG` parameter dynamically in conjunction with any of the parameters `DBID`, `FNR`, `SYSPSW` or `SYSCIP`, you must specify the `FREG` parameter *after* any of these other parameters.
Specification within session	no	

FREG Parameter Syntax

The `FREG` parameter syntax is as follows:

```
FREG=(database-ID,file-number,password,cipher-key)
```

Where:

Syntax Element	Value	Explanation
database-ID	`1-65535`, except `255`	Database identification of the database in which the Natural Registry system file is located. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see Natural profile parameter `LFILE`.
file-number	`1-65535`	File number of the database file in which the Natural Registry system file is located.

Syntax Element	Value	Explanation
password	1 - 8 characters	Password for the Natural Registry system file. **Note:** 1. A password is only required if the Natural Registry system file has been password-protected using the Adabas security feature. 2. For Natural with VSAM system files, the password is used to specify the logical name (DD or DLBL) of the system file as defined to VSAM. Example: `FNAT=(22,5,SYSVSAM)`.
cipher-key	8 numeric characters	Cipher key for the Natural Registry system file. **Note:** A cipher key is only required if the Natural Registry system file has been ciphered using the Adabas security feature.

Notes:

1. If any subparameter of the `FREG` setting is not specified, the corresponding setting of the parameter `DBID`, `FNR`, `SYSPSW` or `SYSCIP` applies for the Natural Registry system file.
2. The `FREG` system file must not reside on a read-only database.
3. If the `FREG` system file is not declared, Natural will use the `FSEC` system file for session registration. If in addition the `FSEC` system file is also not declared, Natural will use the `FNAT` system file for session registration.

Examples:

```
FREG=(,8)
FREG=(22,5,PASSW2)
```

117 FS - Default Format/Length Setting for User-Defined Variables

This Natural profile and session parameter determines whether a default format/length setting is to be in effect for the definition of user-defined variables in reporting mode.

Note: See also *Format and Length of User-Defined Variables* in the *Programming Guide*.

Possible settings	`ON`	No default format/length is assigned by Natural for a newly introduced variable in reporting mode. **Note:** The format/length of all user-defined variables must be explicitly specified.
	`OFF`	A user-defined variable in a Natural program for which no format/length is specified is assigned the default format/length N7.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. This Natural profile and session parameter only applies to reporting mode; it has no effect in structured mode.
2. Within a Natural session, the profile parameter `FS` can be overridden by the session parameter `FS`.

3. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.

118 FSEC - Natural Security System File

This Natural profile parameter defines the database ID, file number, password, cipher key and read-only flag for the Natural Security system file (FSEC), which is used by Natural Security to retrieve/store its security information.

Possible settings	See *FSEC Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	**Note:** If you specify the `FSEC` parameter dynamically in conjunction with any of the parameters `DBID`, `FNR`, `SYSPSW`, `SYSCIP` or `ROSY`, you must specify the `FSEC` parameter *after* any of these other parameters.
Specification within session	no	

Notes:

1. This Natural profile parameter only applies if Natural Security is used.
2. For information on system files, refer to *Natural System Files* in the *Natural System Architecture* documentation.

FSEC Parameter Syntax

The `FSEC` parameter syntax is as follows:

```
FSEC=(database-ID,file-number,password,cipher-key,RO)
```

Where:

Syntax Element	Value	Explanation
`database-ID`	`1-65535`, except `255`	Database identification of the database in which the Natural Security system file is located. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see Natural profile parameter `LFILE`.
`file-number`	`1-65535`	File number of the database file in which the Natural Security system file is located.
`password`	1 - 8 characters	Password for the Natural Security system file. **Note:** A password is only required if the Natural Security system file has been password-protected using the Adabas security feature.
`cipher-key`	8 numeric characters	Cipher key for the Natural Security system file. **Note:** A cipher key is only required if the Natural Security system file has been ciphered using the Adabas security feature.
`RO`	-	Read-only option. **Note:** 1. `RO` indicates that the Natural Security system file is "read-only". 2. `RO` is only specified if modifications on the Natural Security system file are to be disabled.

Note: **Default values:** If any subparameter of profile parameter `FSEC` is not specified, the corresponding setting of profile parameter `DBID`, `FNR`, `SYSPSW`, `SYSCIP` or `ROSY` applies to the Natural Security system file.

Example:

```
FSEC=(10,8)
```

119 FSIZE (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

120

FSPOOL - Natural Advanced Facilities Spool File

This Natural profile parameter defines the database ID, file number, password, cipher key and read-only flag for the Natural Advanced Facilities spool file.

Possible settings	See *FSPOOL Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	**Note:** If you specify the `FSPOOL` parameter dynamically in conjunction with any of the parameters `DBID`, `FNR`, `SYSPSW`, or `SYSCIP`, you must specify the `FSEC` parameter *after* any of these other parameters.
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Natural Advanced Facilities.
2. The spool file is the database file that is used by Natural Advanced Facilities.
3. The spool file must be different from the `FNAT`, `FUSER`, `FDIC`, `FSEC` and `FPROF` system files.
4. For information on system files, refer to *Natural System Files* in the *Natural System Architecture* documentation.

FSPOOL Parameter Syntax

The `FSPOOL` parameter syntax is as follows:

`FSPOOL=(`*`database-ID`*`,`*`file-number`*`,`*`password`*`,`*`cipher-key`*`,RO)`

Where:

Syntax Element	**Value**	**Explanation**
`database-ID`	`1-65535`, except `255`	Database identification of the database in which the spool file is located. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see Natural profile parameter `LFILE`.
`file-number`	`1-65535`	File number of the database file in which the spool file is located.
`password`	1 - 8 characters	Password for the spool system file. **Note:** 1. A password is only required if the spool file has been password-protected using the Adabas security feature. 2. When Natural is used with VSAM system files, the password is used to specify the logical name (DD or DLBL) of the system file as defined to VSAM. Example: `FSPOOL=(10,8,SYSVSAM)`
`cipher-key`	8 numeric characters	Cipher key for the spool file. **Note:** A cipher key is only required if the Natural Security system file has been ciphered using the Adabas security feature.
`RO`	-	Read-only option. **Note:** 1. `RO` indicates that the spool file is “read-only”. 2. `RO` is only specified if modifications on the spool file are to be disabled. This would mean, for example, that no reports could be stored on the spool file.

Example:

```
FSPOOL=(10,8)
```

121

FUSER - Natural System File for User Programs

This Natural profile parameter defines the database ID, file number, password, cipher key and read-only flag for the Natural user-program system file (`FUSER`).

Possible settings	See *FUSER Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	**Note:** If you specify the `FUSER` parameter dynamically in conjunction with any of the parameters `DBID`, `FNR`, `SYSPSW`, `SYSCIP` or `ROSY`, you must specify the `FUSER` parameter *after* any of these parameters.
Specification within session	no	

Notes:

1. The Natural user-program system file (`FUSER`) is the database file from which all user-written Natural programs are retrieved.
2. For information on system files, refer to *System Files* in the *Natural System Architecture* documentation.

FUSER Parameter Syntax

The `FUSER` parameter syntax is as follows:

FUSER=(*database-ID*,*file-number*,*password*,*cipher-key*,RO)

Where:

Syntax Element	Value	Explanation
`database-ID`	`1- 65535`, except `255`	Database identification of the database in which the Natural user-program system file is located. **Note:** Database ID `255` is reserved for logical system files for Software AG products, see Natural profile parameter `LFILE`.
`file-number`	`1-65535`	File number of the database file in which the Natural user-program system file is located.
`password`	1 to 8 characters	Password for the Natural user-program system file. **Note:** 1. A password is only required if the Natural user-program system file has been password-protected using the Adabas security feature. 2. When Natural is used with VSAM system files, the password is used to specify the logical name (DD or DLBL) of the system file as defined to VSAM. Example: `FUSER=(22,5,SYSVSAM)`
`cipher-key`	8 numeric characters	Cipher key for the Natural user-program system file. **Note:** A cipher key is only required if the Natural user-program system file has been ciphered using the Adabas security feature.
`RO`	-	Read-only option. **Note:** 1. `RO` indicates that the Natural user-program system file is "read-only". 2. `RO` is only specified if modifications on the Natural user-program system file are to be disabled.

Note: If any subparameter of the `FUSER` setting is not specified, the corresponding setting of the parameter `DBID`, `FNR`, `SYSPSW`, `SYSCIP` or `ROSY` applies for the Natural user-program system file.

Examples:

```
FUSER=(,8) FUSER=(22,5,PASSW2)
```

122 GC - Filler Character for Group Headers

With this session parameter, you specify the filler character which will appear on either side of a group heading produced by a `DISPLAY` statement across all field columns that belong to that group.

Possible settings	any character	Filler character for group headers.
Default setting	blank	
Specification within session	yes	
Applicable statements	`DISPLAY` `FORMAT`	
Applicable command	none	

Note: Unlike the `FC` parameter, which applies to individual columns, the `GC` parameter applies to headings across a group of columns.

Example:

```
DISPLAY (GC=*)
```

123 HC - Header Centering

This session parameter determines the placement of column headers.

Possible settings	`C`	Headers will be centered.
	`L`	Headers will be left-justified.
	`R`	Headers will be right-justified.
Default setting	`C`	
Specification within session	yes	
Applicable statements	`DISPLAY` `FORMAT`	
Applicable command	none	

Example:

```
DISPLAY (HC=L)
```

124 HCAM - Hardcopy Access Method

This Natural profile parameter determines which access method is to be used for hardcopy output processing.

Possible settings	With `HCAM=`*`value`*, you can specify one of the following access-method names:	
	Value:	**Access Method:**
	`STD`	Standard sequential file (batch, TSO, TIAM).
	`COMP`	Com-plete print file.
	`CICS`	CICS transient data or temporary storage.
	`IMS`	IMS TM printer.
	`NAF`	Natural Advanced Facilities.
	`USER`	Third-party vendor print interface.
	`SMARTS`	SMARTS print file.
	`ESS`	Entire System Server.
	`ANY`	Hardcopy output processing will be handled by the first access method available (the search sequence for available access methods is the sequence in which the access methods are listed here).
	`OFF`	Hardcopy output processing will not be handled by any access method.
Default setting	`ANY`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. `HCAM=`*`value`* is equivalent to the `AM` subparameter of the profile parameter `PRINT` for Print File 0, that is, `PRINT=((0),AM=`*`xxx`*`)`.
2. The hardcopy output destination is specified using the profile parameter `HCDEST`. More specifications for the hardcopy output file can be made using the `PRINT` profile parameter or the `NTPRINT` macro for Printer 0.
3. Under BS2000, `HCAM=STD` is a necessary setting for routing hardcopy output to standard print files.

125 HCDEST - Hardcopy Output Destination

This Natural profile parameter presets the hardcopy output destination for the terminal command `%H` (without the *`destination`* operand).

Possible settings	1 - 8 characters	Valid hardcopy output destination.
	blank	
Default setting	blank	**Note:** In some environments, a default destination may be supplied by the operating system or TP monitor. If `HCAM=STD` is assigned for hardcopy, the default hardcopy output destination is the data set `CMHCOPY`.
Dynamic specification	yes	
Specification within session	yes	**Note:** The hardcopy output destination can be overwritten during the session by specifying `%H`*`destination`*; see also the terminal command `%H`.

Notes:

1. `HCDEST=`*`value`* is equivalent to the `DEST` subparameter of the profile parameter `PRINT` for Print File 0, that is, `PRINT=((0),DEST=`*`value`*`)`.
2. If you are running Natural under TSO or in batch mode, the data set must be defined in the JCL or by dynamic allocation.
3. Under TSO, the hardcopy data set specified by `HCDEST` is closed after `%H` at the next terminal I/O. The default `CMHCOPY` data set is closed not at terminal I/O, but at session termination.
4. The hardcopy output access method can be specified by profile parameter `HCAM` or by the `DEST` subparameter of profile parameter `PRINT` for Printer 0. More specifications for the hardcopy output file can be made using the profile parameter `PRINT` or the macro `NTPRINT` for Printer 0.

126 HD - Header Definition

With this session parameter, you define which default text is to be used when

- the field is output with a `DISPLAY` statement;
- an equal sign (=) is placed immediately before the field in a `WRITE` or `INPUT` statement.

Possible settings	`'text'`	120 alphanumeric or Unicode characters at maximum.
Default setting	none	
Applicable statements	`DEFINE DATA`	Parameter may be specified at field level and/or element level.
Applicable command	none	

This parameter can be specified

- at field/element level in a `DEFINE DATA` statement; see the sections *View Definition* and *EM, HD, PM Parameters for Field/Variable*;
- in the `Miscellaneous` field of the Data Area Editor (see *Columns in the Editing Area*);
- in the SYSDDM utility (see *Specifying Extended Field Attributes*).

127 HE - Helproutine

With this session parameter, you assign a helproutine or a help map to a field.

Possible settings		See *HE Parameter Syntax* below.
Default setting	none	
Specification within session	yes	
Applicable statements	`INPUT`	
Applicable command	none	

Helproutines can be created with the Natural program editor, help maps with the Natural map editor.

The helproutine or help map may then be invoked during processing of an `INPUT` statement or a map by choosing either of the following methods:

- In the field for which to invoke the help request, enter the help character in the leftmost position of the field and press ENTER. The default help character is a question mark (?).

  If you enter the help character at a different position of the field or if you enter more than one character, the string is taken as user input and no help is invoked. If the field contains hexadecimal zeroes, it depends on the terminal emulation whether Natural can interpret the values as a help request.
- Or:

  Place the cursor in the field for which to invoke the help request and press the PF key defined as help function key with the `SET KEY` statement.

The following topics are covered below:

HE Parameter Syntax

The syntax of this parameter is:

$$\texttt{HE=}\textbf{operand1}\left[\;,\left\{\begin{array}{l}\textbf{operand2}\\ \textbf{=}\\ \textbf{nX}\end{array}\right\}\right]\ldots 20$$

Operand Definition Table:

Operand	Possible Structure						Possible Formats													Referencing Permitted	Dynamic Definition
operand1	C	S					A													no	no
operand2	C	S	A				A	U	N	P	I	F	B	D	T	L	C		O	no	no

Syntax Element Description:

Syntax Element	Description
operand1	*operand1* is the name of the helproutine or help map to be invoked. The name may be a 1 to 8 character alphanumeric constant or user-defined variable. If a variable is used, it must have been previously defined. The case of the specified name is not translated. The name may contain an ampersand (&); at execution time, this character will be replaced by the one-character code corresponding to the current value of the Natural system variable `*LANGUAGE`. This feature allows the use of multi-lingual helproutines or help maps. For additional information on using *operand1* within a map, see the `HE` helproutine option described in *Extended Field Editing* in *Map Editor* in the *Editors* documentation.
operand2	You may specify 1 to 20 parameters (*operand2*) which are passed to the helproutine or help map. They may be specified as constants or as user-defined variables which contain the values of the parameters. For additional information on using *operand2* within a map, see the HE helproutine option described in *Extended Field Editing* in *Map Editor* in the *Editors* documentation.
=	The equals sign (=) is used to pass an object or a field name to a helproutine or help map: - If the equals sign is entered in the `HE=` specification at statement level, the name of the object (as contained in the system variable `*PROGRAM`) being executed is passed to the helproutine or help map. In **Example 3**, the object name passed is `PROGRAM1`. - If the equals sign is entered in the `HE=` specification at field level, the name of the field is passed to the helproutine or help map. In **Example 3**, the field name passed is `#PARM1`. If the equals sign is used as a parameter, the corresponding parameter in the helproutine or help map must be specified with format/length A65.
*n*X	The notation *n*`X` can be used to specify parameters to be omitted, that is, for which no values are to be passed. The corresponding receiving parameters in the called helproutine's `DEFINE DATA PARAMETER` statement must be defined as `OPTIONAL`.

Notes:

1. The operands must be separated either by the input delimiter character (as specified with the session parameter `ID`) or by a comma. However, a comma must not be used for this purpose if the comma is defined as decimal character (with the session parameter `DC`).
2. If parameters are specified, the helproutine must begin with a `DEFINE DATA PARAMETER` statement which defines fields that correspond with the parameters in format and length.

3. The value of the field for which a helproutine is specified may be referenced within the helproutine. This is done by specifying a field in the `DEFINE DATA PARAMETER` statement which corresponds in format and length with the original field. In the block of fields defined within the `DEFINE DATA PARAMETER` statement, this field must always be defined behind the parameters, if present.
4. If the field for which a helproutine is specified is an array element, its indices may be referenced by the helproutine. To do so, you specify index parameters with format I (integer), N (numeric unpacked), or P (packed numeric) at the end of the `DEFINE DATA PARAMETER` statement. You may specify up to three index parameters according to array dimensions.

Execution of Helproutines

If a helproutine or help map is requested - by entering a question mark (?) in the field, or by pressing the help key (as defined with a `SET KEY` statement), or via a `REINPUT USING HELP` statement - all other data that may have been entered into fields are not assigned to the program variables until all help requests have been processed.

Note: Only one help request per `INPUT` statement is possible; that is, if help is requested for more than one field (for example, by entering question marks in multiple fields), only the first help request will be executed.

Examples

Example 1:

```
/* MAIN PROGRAM
DEFINE DATA
1 #A(A20/1:3)
END-DEFINE
...
SET KEY PF1=HELP
...
INPUT #A (2) (HE='HELPA',=)
...
END
```

Example 2:

```
/* HELP-ROUTINE 'HELPA'
DEFINE DATA PARAMETER
1 #VARNAME  (A65)
1 #PARM1    (A20)
1 #VARINDEX (I2)
END-DEFINE
...
```

Example 3:

```
* Program 'PROGRAM1'
*
DEFINE DATA LOCAL
1 #PARM1 (A65) INIT <'valueparm1'>
END-DEFINE
SET KEY PF1 = HELP
FORMAT KD=ON
*
INPUT (AD=M HE='HELP1',=)
  'Enter ? for name of executed object:'
  / #PARM1
*
INPUT (AD=M)
  'Enter ? for field name:'
  / #PARM1 (HE='HELP1',=)
*
END
```

Parameter Data Area in Example Helproutine `HELP1`:

```
* Helproutine 'HELP1'
*
DEFINE DATA PARAMETER
1 #FLD1 (A65)
END-DEFINE
...
```

128 HI - Help Character

This Natural profile parameter defines the character which is to be used to invoke a field-specific helproutine or a map helproutine (if defined for a given map).

Possible settings	any special character	The character which is to be used to invoke a field-specific helproutine or a map helproutine. **Note:** The character specified with the `HI` parameter must not be the same as the one specified with the `CF` (control character for mainframe terminal commands) parameter. In addition, we recommend that this character is not the same as the one specified with the `DC` (decimal character), `IA` (input assign character) or `ID` (input delimiter character) parameter.
	blank	**Note:** 1. Numeric fields which have a helproutine assigned are internally translated to alphanumeric format so as to make it possible for the user to enter a question mark into the field to invoke the helproutine. 2. To prevent this internal translation (that is, if you wish to make sure that alphabetical characters cannot be entered into a numeric field) you can set the profile parameter `HI` to blank. 3. When `HI=' '` is set, a help key must be defined in the Natural application, using the `SETKEY` statement correspondingly; otherwise it is not possible to invoke a helproutine for any field.
Default setting	?	Question mark.
Dynamic specification	yes	
Specification within session	no	

Application programming interface	`USR0350N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

129 HW - Heading Width

With this session parameter you determine the width of a column output with a `DISPLAY` statement.

Possible settings	`ON`	The width of a `DISPLAY` column is determined by either the length of the heading text or the length of the field, whichever is longer. **Note:** This is true even if no heading text is output, either because the `DISPLAY` statement contains the keyword `NOHDR` or the `DISPLAY` statement is a subsequent `DISPLAY` (see also the `DISPLAY` statement).
	`OFF`	The width of a `DISPLAY` column is determined by the length of the field. **Note:** `HW=OFF` only applies to `DISPLAY` statements which do not create headers (that is, either a first `DISPLAY` statement with `NOHDR` option or a subsequent `DISPLAY` statement).
Default setting	`ON`	
Specification within session	yes	
Applicable statements	`DISPLAY` `FORMAT`	
Applicable command	none	

Example:

```
DISPLAY (HW=OFF)
```

130 IA - Input Assign Character

This Natural profile and session parameter defines the character to be used as the assignment character for the input parameter processing in `INPUT` statements, either in keyword/delimiter mode or when processing data from the Natural stack.

Possible settings	any special character	Assignment character for the input parameter processing in `INPUT` statements.
Default setting	=	Equals sign.
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR0350N`, `USR1005N` *	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation. * Recommended.

Notes:

1. The character specified with the `IA` parameter must not be the same as the character specified with the `DC` (decimal character) or `ID` (input delimiter character) parameter. In addition, we recommend that this character is not the same as the one specified with the `CF` (control character for mainframe terminal commands) or `HI` (help character) parameter.
2. Within a Natural session, the profile parameter `IA` can be overridden by the session parameter `IA`.
3. Under Natural Security, the setting of this parameter can be overridden by the *Session Parameters* option of the Library Profile.

Example:

In the following example, it is assumed that, for the beginning, the default input assign character (=) applies.

```
** Example 'IACHAR': Input Assign character
************************************************************************
DEFINE DATA LOCAL
1 #A (A1)
1 #B (A1)
END-DEFINE
*
INPUT #A #B
*
WRITE 'Field #A:' #A  / 'Field #B:' #B
*
END
```

1. Enter the command

   ```
   IACHAR #A=Y,#B=X
   ```

   The program produces the following output:

   ```
   Page      1                                                     05-01-19  11:05:51

   Field #A: Y
   Field #B: X
   ```

2. Enter the command

   ```
   GLOBALS IA=:
   ```

   This sets the input assign character to colon (:).

3. Then enter the command

   ```
   IACHAR #B:X,#A:Y
   ```

   The program produces the following output:

   ```
   Page      1                                                     06-11-13  12:12:24

   Field #A: Y
   Field #B: X
   ```

131

IC - Insertion Character

With this session parameter, you specify the character string to be inserted in the column immediately preceding the value of a field output with a `DISPLAY` statement. The width of the output column is increased accordingly.

Possible settings	any character	Character string to be inserted. You can specify a string of one to ten characters. **Note:** Insertion characters may optionally be specified within apostrophes, in which case any characters can be specified. Any character string specified which contains a closing parenthesis or a quotation mark must be enclosed within apostrophes. A blank in a character string not enclosed within apostrophes is represented by a circumflex accent (^).
Default setting	none	
Specification within session	yes	
Applicable statements	`FORMAT`	Parameter may be specified dynamically with the `FORMAT` statement.
	`DISPLAY`	Parameter may be specified at statement level and/or at element level.
Applicable command	none	

Notes:

1. The insertion character is inserted between leading spaces and the field value whereas the leading character is output in front of the leading space.
2. For numeric values, the insertion characters will be placed before the first significant digit printed.
3. The `IC` and `LC` parameters are mutually exclusive.
4. The parameter `IC` can also be used with U format fields.

5. For information on Unicode format, see also *Unicode and Code Page Support in the Natural Programming Language, Session Parameters, EMU, ICU, LCU, TCU versus EM, IC, LC, TC* in the *Unicode and Code Page Support* documentation.
6. The difference between the session parameters `LC`, `LCU` and `IC`, `ICU` will be evident, if the corresponding field is output right justified (session parameter `AD=R`).
7. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

Examples:

```
DISPLAY AA(IC=*)
DISPLAY SALARY(IC='$')
```

132 ICU - Unicode Insertion Character

With this session parameter, you specify the character string to be inserted in the column immediately preceding the value of a field output with a `DISPLAY` statement. The width of the output column is enlarged accordingly.

Possible settings	any character	Character string to be inserted. You can specify a string of one to ten characters. **Note:** Insertion characters may optionally be specified within apostrophes, in which case any characters can be specified. Any character string specified which contains a closing parenthesis or a quotation mark must be enclosed within apostrophes. A blank in a character string not enclosed within apostrophes is represented by a circumflex (^).
Default setting	none	
Specification within session	yes	
Applicable statements	`FORMAT`	Parameter may be specified dynamically with the `FORMAT` statement.
	`DISPLAY`	Parameter may be specified at statement level and/or at element level.
Applicable command	none	

Notes:

1. The session parameter `ICU` is identical to the session parameter `IC`. The difference is that the insertion characters are always stored in Unicode format. This allows you to specify insertion characters with mixed characters from different code pages, and assures that always the correct character is displayed independent of the installed system code page.
2. For numeric values, the insertion characters will be placed before the first significant digit printed.
3. The parameters `ICU` and `LCU` are mutually exclusive.

See also:

- *Parameters to Influence the Output of Fields* in the *Programming Guide*
- *Unicode and Code Page Support in the Natural Programming Language, Session Parameters, EMU, ICU, LCU, TCU versus EM, IC, LC, TC* in the *Unicode and Code Page Support* documentation.

133 ID - Input Delimiter Character

This Natural profile and session parameter defines the character to be used as a delimiter character for `INPUT` statements in keyword/delimiter mode.

Possible settings	any special character or blank	Input delimiter character. **Note:** If "blank" is specified, no input delimiter character is defined.
Default setting	,	Comma (,). **Note:** If the input delimiter character is to be a comma (,), it must be specified as `ID=','` when using the dynamic parameter facility, because the comma character separates individual parameters.
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The character specified with this parameter must not be the same as the one specified with the `DC` (decimal character) or `IA` (input assign character) parameter. In addition, we recommend that this character is not the same as the one specified with the `CF` (control character for mainframe terminal commands) or `HI` (help character) parameter.

2. The period (.) should not be used as input delimiter, because this might lead to situations in which a program termination period would be misinterpreted as input delimiter. An asterisk (*) should not be used either.
3. Within a Natural session, the profile parameter `ID` can be overridden by the session parameter `ID`.
4. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.

134 IKEY - Processing of PA and PF Keys

This Natural profile parameter specifies the action to be taken when a video-terminal program-attention key (PA key) or program-function key (PF key) is used to enter data, and the key has not been defined to the Natural program with the `SET KEY` statement.

Possible settings	`ON`	The value `ENTR` is placed in the Natural system variable `*PF-KEY`; that is, Natural reacts as if ENTER had been pressed.
	`OFF`	A `REINPUT` message is generated, prompting the user to press a valid key.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

135 IM - Input Mode

This Natural profile and session parameter determines the default mode for video-terminal input.

Possible settings	`F`	Forms mode.
	`D`	Delimiter mode.
Default setting	`F`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Within a Natural session, the setting of the profile parameter `IM` can be overridden by the session parameter `IM`.
2. The `IM` parameter setting may also be changed with the Natural terminal commands `%D` and `%F`.
3. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.
4. For information on delimiter mode and forms mode, see the `INPUT` statement.

136 IMSG - Session Initialization Error Messages

This Natural profile parameter is used to suppress the initialization error-messages screen. It can be useful to avoid undesired output, for example, for printer sessions.

Caution: As error diagnosis may become difficult, use this parameter with caution.

Possible settings	`ON`	The initialization error messages screen is displayed in the case of an error.
	`OFF`	The initialization error messages screen is not displayed.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

137

IMSP - General Parameters for Natural IMS TM Interface

This Natural profile parameter can only be specified with the `NTIMSP` macro, dynamic parameter specification is not possible yet.

The `NTIMSP` macro is used to set general parameters for the Natural IMS TM Interface.

The `NTIMSP` macro can be specified only once in the Natural parameter module.

Possible settings	See below.
Default setting	See below.
Dynamic specification	no
Specification within session	no

This section covers the following topics:

NTIMSP Macro Syntax

The `NTIMSP` macro is specified as follows:

```
         NTIMSP NIINAME=value,                                           *
               SUBPOOL=value,                                            *
               THRELO=value,                                             *
               TRNCODE=value
```

See *Keyword Subparameters*.

Keyword Subparameters

`NIINAME` | `SUBPOOL` | `THRELO` | `TRNCODE`

NIINAME - Natural IMS TM Interface Module Name

`NIINAME=value` specifies the name of the Natural IMS TM Interface module to be used by the current driver.

Value	Explanation
1- 8 characters	A valid module name.
`NIIINTFM`	This is the default value.

SUBPOOL - Storage Subpool

`SUBPOOL=`*`value`* specifies the z/OS subpool to be used for storage requests.

Value	Explanation
`0 - 127`	The number of a z/OS task-related subpool used by a problem state program.
`125`	This is the default value.

THRELO - Thread Relocation

`THRELO=`*`value`* determines whether the Natural storage thread can be allocated at a different virtual address after a terminal I/O.

Value	Explanation
`ON`	Thread relocation is enabled: the Natural thread can be allocated at a different virtual address after a terminal I/O. This is the default value.
`OFF`	Thread relocation is disabled and the Natural thread remains at the same virtual address. The size of the Natural thread is determined by the first Natural session that allocates the thread. As a consequence: ■ the `THSIZE` keyword subparameter (`NTIMSPE` macro) is ignored except for the first Natural session in an MPP environment, ■ the `RELO` profile parameter is ignored, and ■ the storage for the Natural thread remains allocated until the MPP environment is stopped. **Note:** We recommend that you only set `THRELO=OFF` if you use Con-form.

TRNCODE - Identify Transaction Code

`TRNCODE=`*`value`* can be used to identify the transaction code.

Value	Explanation
`ON`	The `TRNCODE` keyword subparameter is activated to evaluate the `TRNCODE` parameter: ■ on the first card of the BPM `CONTROL` file, ■ in the `NIMBOOT` macro, ■ used as the startup parameter of the server environment.
`OFF`	The `TRNCODE` keyword subparameter is deactivated. This is the default value.

Example of NTIMSP Macro

```
         NTIMSP NIINAME=NATIMS1,SUBPOOL=0
```

138 IMSPE - Environment Parameters for Natural IMS TM Interface

This Natural profile parameter can only be specified with the `NTIMSPE` macro, dynamic parameter specification is not possible yet.

The `NTIMSPE` macro is used to define environment-specific parameter sets, which can be assigned to transaction definitions (`ENVPID` keyword subparameter of `NTIMSPT` macro).

Note: You can only define an `NTIMSPE` macro to the Natural parameter module if the module already contains at least one `NTIMSPT` macro. Otherwise, an assembly error occurs when trying to assemble the module.

Possible settings	See below.
Default setting	See below.
Dynamic specification	no
Specification within session	no

This section covers the following topics:

NTIMSPE Macro Syntax

The `NTIMSPE` macro is specified as follows:

```
         NTIMSPE ACTACTV=value,                                          *
               ACTAHDR=value,                                            *
               ACTARID=value,                                            *
               ACTLOG=value,                                             *
               BMPABER=value,                                            *
               BROACTV=value,                                            *
               CMBSIZE=value,                                            *
               COLPSCR=value,                                            *
               ENDMODN=value,                                            *
               ENTRYNM=value,                                            *
               ERRLHDR=value,                                            *
               HCBSIZE=value,                                            *
               HDENSDU=value,                                            *
               LINPSCR=value,                                            *
               MISIZE=value,                                             *
               MONACTV=value,                                            *
               MOSIZE=value,                                             *
               NSBNAME=value,                                            *
               PRTDRIV=value,                                            *
               ROLLSRV=value,                                            *
               ROLLFN=value,                                             *
               SPASIZE=value,                                            *
               SPATID=value,                                             *
               SUPNONC=value,                                            *
               TERMDB=value,                                             *
               TERMIPL=value,                                             *
               THBELOW=value,                                             *
               THSIZE=value,                                              *
               USERID=value
```

See ***Keyword Subparameters***.

Keyword Subparameters

`ACTACTV` | `ACTAHDR` | `ACTARID` | `ACTLOG` | `BMPABER` | `BROACTV` | `CMBSIZE` | `COLPSCR` | `ENDMODN` | `ENTRYNM` | `ERRLHDR` | `HCBSIZE` | `HDENSDU` | `LINPSCR` | `MISIZE` | `MONACTV` | `MOSIZE` | `NSBNAME` | `PRTDRIV` | `ROLLSRV` | `ROLLFN` | `SPASIZE` | `SPATID` | `SUPNONC` | `TERMDB` | `TERMIPL` | `THBELOW` | `THSIZE` | `USERID`

ACTACTV - Activate Accounting

This keyword subparameter only applies in dialog-oriented environments.

`ACTACTV=value` specifies whether the accounting function is activated.

Value	Explanation
`ON`	An accounting record is written with each terminal I/O.
`OFF`	No accounting record is written. This is the default value.

ACTAHDR - Header for Accounting Records

This keyword subparameter only applies to the accounting function.

`ACTAHDR=value` defines the header for the accounting records if written to the IMS log file. This keyword subparameter is only evaluated if the `ACTLOG` keyword subparameter is set to `CMD`.

Value	Explanation
1 - 8 characters	A header name.
`SAG$$$$$`	This is the default value.

ACTARID - Accounting Record ID

This keyword subparameter only applies to the accounting function.

`ACTARID=`*`value`* specifies the accounting record ID if the accounting record is written using the `LOG` or `SMF` settings of the `ACTLOG` keyword subparameter.

Value	Explanation
`A0 - FF`	The log code if `ACTLOG` is set to `LOG`.
`128 - 255`	The number of the SMF record type if `ACTLOG` is set to `SMF`.

Note: There is no default value.

ACTLOG - Write Accounting Records to Log File

This keyword subparameter only applies to the accounting function.

`ACTLOG=`*`value`* specifies how accounting records are written.

Value	Explanation
`CMD`	Accounting records are written to the IMS log file using the `CMD` call. This is the default value.
`LOG`	Accounting records are written to the IMS log file using the `LOG` call.
`SMF`	Accounting records are written to SMF using the *Authorized Services Manager* (see the *TP Monitor Interfaces* documentation).

BMPABER - Specify Termination Error for BMP Run

`BMPABER=`*`value`* specifies how a BMP run should be terminated if either a Natural runtime error or a Natural IMS TM Interface non-recoverable error occurs.

Value	Explanation
`ON`	The BMP run is terminated with user abend code U3521.
`OFF`	The BMP run is terminated normally with the Natural termination error as the condition code. If the BMP run is terminated with a non-recoverable Natural IMS TM Interface error, condition code 1024 is set. This is the default value.

BROACTV - Enable/Disable Broadcasting

This keyword subparameter only applies in dialog-oriented environments.

`BROACTV=value` enables or disables the broadcasting function.

Value	Explanation
`ON`	The broadcasting function is enabled
`OFF`	The broadcasting function is disabled. This is the default value.

CMBSIZE - Command Buffer Size

`CMBSIZE=value` specifies the size of the command buffer.

The command buffer is used by the application programming interfaces `NIICMD` and `NIIGCMD`, the service module `CMCMMND` and the accounting function. For details, see the *TP Monitor Interfaces* documentation.

The size of the command buffer must accommodate the maximum length of the IMS commands to be processed and the maximum length of the accounting record including the user extension.

Value	Explanation
`100 - 16777216`	A buffer size in KB.
`1024`	This is the default value.

COLPSCR - Number of Screen Columns

`COLPSCR=value` specifies the number of columns per screen.

Value	Explanation
`35 - 250`	A screen width.
`80`	This is the default value.

ENDMODN - Format of Termination Screen

`ENDMODN=value` specifies the `MOD name` to be used for formatting the screen which appears after a Natural session is terminated successfully. `ENDMODN` enables Natural to be included in a customer-specific menu.

The value of `ENDMODN` can be overridden by the application programming interface `NIIEMOD` and the service module `CMEMOD`. For details, see the *TP Monitor Interfaces* documentation.

If a Natural session terminates with an error, `DFSMO2` is always used to issue the appropriate Natural error message.

Value	Explanation
1 - 8 characters	A valid `MOD name`.
`DFSMO2`	This is the default value.

ENTRYNM - Identify Current Environment Parameter Set

`ENTRYNM=value` identifies the current environment-specific parameter set.

Value	Explanation
1 - 8 characters	The name of the environment parameter set currently used.
`ENV00000`	This is the default value.

ERRLHDR - Header for IMS TM Error Logs

`ERRLHDR=value` specifies the header of the IMS TM log records which are written when errors occur in the Natural IMS TM Interface.

If you do not wish a message to be written to the IMS TM log in the case of a non-recoverable Natural IMS TM Interface error, explicitly set `ERRLHDR` to null, that is, specify `ERRLHDR=,`.

For further information, see *Recovery Handling* in the *TP Monitor Interfaces* documentation.

Value	Explanation
1 - 8 characters	A header name.
`NIIERR$$`	This is the default value.

HCBSIZE - Size of Hardcopy Print Buffer

`HCBSIZE=`*`value`* specifies the size of the hardcopy print buffer.

Records which are sent to a printer destination using the Natural hardcopy function are buffered.

Value	Explanation
`0 - 16777216`	A buffer size in KB.
`1024`	This is the default value.

HDENSDU - High-Density Dump

`HDENSDU=`*`value`* specifies whether a snap dump provoked by a Natural IMS TM Interface error is to be written as a high-density dump to a 3800 printing subsystem.

Value	Explanation
`ON`	A high-density dump is written.
`OFF`	No high-density dump is written. This is the default value.

LINPSCR - Number of Screen Lines

`LINPSCR=`*`value`* defines the number of lines per screen.

Value	Explanation
`1 - 250`	A screen size.
`24`	This is the default value.

MISIZE - Buffer Size for Input Message

`MISIZE=`*`value`* specifies the size of the buffer which is to contain the input message.

This area must be as large as the largest input message to be received from IMS TM.

Value	Explanation
`100 - 16777216`	A buffer size in KB.
`4096`	This is the default value.

MONACTV - Enable/Disable Monitoring

This keyword subparameter only applies in dialog-oriented environments.

`MONACTV=`*`value`* enabled or disables the monitoring function. If enabled, the session status is written to the SIP server at each terminal I/O. If disabled, no session status is maintained.

Value	Explanation
`ON`	The monitoring function is enabled.
`OFF`	The monitoring function is disabled. This is the default value.

MOSIZE - Buffer Size for Output Message

`MOSIZE=`*`value`* specifies the size of the buffer which is to contain the output message. This area must be as large as the largest output message to be sent to IMS TM.

Value	Explanation
`2048 - 16777216`	A buffer size in KB.
`4096`	This is the default value.

NSBNAME - Use NSB Name of Natural DL/I Interface

`NSBNAME=`*`value`* sets the name of the NSB to the NSB used by the Natural DL/I Interface.

Value	Explanation
`ON`	Sets the NSB name to the PSB name defined by the `PSB` keyword subparameter of the `NTIMSPT` macro for the transaction code in use.
`OFF`	Sets the NSB name to the PSB name used by IMS TM. This is the default value.

PRTDRIV - Driver for IMS TM Printer

`PRTDRIV=`*`value`* specifies the print driver to be used for reports which are directly written to an IMS TM printer.

For further information, see *Support of the Natural WRITE (n) Statement* in the *TP Monitor Interfaces* documentation.

Value	Explanation
driver-name	A driver name as listed in the following tables: *Drivers for SCS Printers* *Drivers for Non-SCS Printers* *Drivers for JES API*
SCS_S2	This is the default value.

Drivers for SCS Printers

Driver	Purpose
SCS_B1	Form feed at start and end of report, starts page on line 1.
SCS_B2	Form feed at start and end of report, starts page on line 2.
SCS_E1	Form feed at end of report, starts page on line 1.
SCS_E2	Form feed at end of report, starts page on line 2.
SCS_N1	No form feed at start or end of report, starts page on line 1.
SCS_N2	No form feed at start or end of report, starts page on line 2.
SCS_S1	Form feed at start report, starts page on line 1.
SCS_S2	Form feed at start of report, starts page on line 2.

Drivers for Non-SCS Printers

Driver	Purpose
NSCS_B1	Form feed at start and end of report, starts page on line 1.
NSCS_B2	Form feed at start and end of report, starts page on line 2.
NSCS_E1	Form feed at end of report, starts page on line 1.
NSCS_E2	Form feed at end of report, starts page on line 2.
NSCS_N1	No form feed at start or end of report, starts page on line 1.
NSCS_N2	No form feed at start or end of report, starts page on line 2.
NSCS_S1	Form feed at start report, starts page on line 1.
NSCS_S2	Form feed at start of report, starts page on line 2.

Drivers for JES API

<table>
<tr><th>Driver</th><th colspan="3">Purpose</th></tr>
<tr><td rowspan="4">JES</td><td colspan="3">The following data set processing options for JES are taken from the corresponding NTPRINT macro parameters or DEFINE PRINTER statement (see the <i>Statements</i> documentation) options:</td></tr>
<tr><td>JES</td><td>NTPRINT</td><td>DEFINE PRINTER</td></tr>
<tr><td>CLASS</td><td>CLASS</td><td>CLASS</td></tr>
<tr><td>COPIES</td><td>COPIES</td><td>COPIES</td></tr>
</table>

Driver	Purpose		
	DEST	DEST	OUTPUT
	FORMS	FORMS	FORMS
	NAME	NAME	NAME
	OUTDISP	DISP	DISP
	PRTY	PRTY	PRTY

The generated JES API parameter string is:

```
IAFP=A0M,PRTO=..OUTDI(disp),DES(dest),
CLA(class,COP(copies),
FORMS(forms),NAME(name),
PRTY(prty)
```

Note: Unspecified `NTPRINT/DEFINE PRINTER` parameters/options are ignored.

JES *xxxxx*

The data set processing options for JES are taken from the `OUTPUT JCL` statement with the name `JES`*xxxxx* where *xxxxx* can be up to 5 characters.

The generated JES API parameter string is:

```
IAFP=A0M,OUTN=JESxxxxx
```

The `OUTPUT JCL` statement may look like:

```
JESxxxxx OUTPUT OUTDISP=WRITE,DEST=dest,
CLASS=A,COPIES=1,FORMS=form,...
```

Note: If the `OUTPUT JCL` statement is missing in the job stream, an error is reported.

ROLLSRV - Natural Thread Storage

This keyword subparameter only applies in dialog-oriented environments.

`ROLLSRV=`*value* specifies the medium for saving a Natural thread between terminal output and input.

See also *Roll File and Roll Server* in the *TP Monitor Interfaces* documentation.

Value	Explanation
ON	The Natural roll server is used. This is the default value.
OFF	Roll files are used: see ROLLFN.

ROLLFN - Number of Roll Files

This keyword subparameter only applies in dialog-oriented environments.

`ROLLFN=value` specifies the number of roll files to be used if `ROLLSRV` is set to `OFF`.

Value	Explanation
`1 - 5`	The number of roll files to be used.
`1`	This is the default value.

SPASIZE - Buffer Size for Scratch-Pad Area

This keyword subparameter only applies in dialog-oriented environments.

`SPASIZE=value` specifies the size of the buffer which is to contain the scratch-pad area.

In a non-conversational environment, this is also the size of the simulated SPA which is written to the SIP server.

Value	Explanation
`256 - 16777216`	A buffer size in KB.
`1024`	This is the default value.

SPATID - Subsystem ID for ASM

`SPATID=value` specifies the Natural subsystem ID for the Authorized Services Manager (ASM) which is used to save the SPA for a non-conversational driver. The ASM is described in the *Operations* documentation.

Value	Explanation
1 - 5 characters	A subsystem ID. This value must be the same for all parameter tables and must be the same as the value specified for `SPATID` in the `NIMPIXT` macro (see the *TP Monitor Interfaces* documentation).
`NAT2`	This is the default value.

SUPNONC - Enable/Disable Environment Switching

This keyword subparameter only applies in dialog-oriented conversational environments.

`SUPNONC=value` specifies whether switching from a terminal-oriented non-conversational environment to a conversational environment is possible.

Value	Explanation
`ON`	Switching is allowed.
`OFF`	Switching is not allowed. This is the default value.

TERMDB - Session Termination for Missing DL/I Database

This keyword subparameter only applies in dialog-oriented environments.

`TERMDB=value` specifies whether the Natural session is to be terminated if one of the DL/I databases specified in the PSB is not available.

Value	Explanation
`ON`	The Natural session terminates.
`OFF`	The Natural session does not terminate. If one of the databases is not available when it is accessed, the Natural transaction code is suspended by IMS TM. This is the default value.

TERMIPL - Session Termination for IPL

This keyword subparameter only applies in dialog-oriented environments.

`TERMIPL=value` specifies whether a Natural session is terminated with an error message when an IPL has occurred between the current transaction step and the start of the session.

Value	Explanation
`ON`	The Natural session terminates.
`OFF`	The Natural session does not terminate. This is the default value.

THBELOW - Natural Thread Allocation

`THBELOW=value` specifies whether the Natural thread is allocated below or above the 16 MB line.

Value	Explanation
`ON`	The Natural thread is allocated below the 16-MB line. This is the default value.
`OFF`	The Natural thread is allocated above the 16 MB line.

THSIZE - Natural Thread Size

`THSIZE=value` specifies the size of the Natural thread. This is the area which contains all Natural buffers that relate to user sessions.

Value	Explanation
`100000 - 99999999`	A thread size in bytes in multiples of eight greater than or equal to 100000.
`1048576`	This is the default value.

USERID - Determine *INIT-USER Value

This keyword subparameter only applies to the BMP driver.

`USERID=value` specifies how the value of the system variable `*INIT-USER` (see the *System Variables* documentation) is determined.

Value	Explanation
`ON`	The Natural user ID specified in `*INIT-USER` is either taken from the security access control block if a security package is active or from the `USER` parameter of the job card.
`OFF`	The Natural user ID specified in `*INIT-USER` is taken from the job name. This is the default value.

Example of NTIMSPE Macro

```
         NTIMSPE ENTRYNM=ENV0021,                                      *
               ACTACTV=ON,                                             *
               ACTLOG=LOG,                                             *
               ACTARID=A1,                                             *
               PRTDRIV=SCS_E,                                          *
               THBELOW=OFF,                                            *
               THSIZE=1000000,                                         *
               SPATID=ASM1
```

139 IMSPT - Transaction Definitions for Natural IMS TM Interface

This Natural profile parameter can only be specified with the `NTIMSPT` macro, dynamic parameter specification is not possible yet.

The `NTIMSPT` macro is required to define the transaction code for each Natural transaction, along with individual transaction parameters. The keyword subparameters `TRAN` and `PSB` must be explicitly specified; there are no default value settings.

Possible settings	See below.
Default setting	See below.
Dynamic specification	no
Specification within session	no

This section covers the following topics:

NTIMSPT Macro Syntax

The `NTIMSPT` macro is specified as follows:

```
         NTIMSPT ALTPCB=value,                                        *
               ENVPID=value,                                          *
               HCPCB=value,                                           *
               MSGPCB=value,                                          *
               NRAST=value,                                           *
               PCBS=value,                                            *
               PSB=value,                                             *
               TRAN=value,                                            *
               TYPE=value,                                            *
               WRKPCBS=value
```

See *Keyword Subparameters*.

Keyword Subparameters

`ALTPCB` | `ENVPID` | `HCPCB` | `MSGPCB` | `NRAST` | `PCBS` | `PSB` | `TRAN` | `TYPE` | `WRKPCBS`

ALTPCB - Alternate TP PCB

`ALTPCB=value` determines the alternate TP PCB to be used for the service modules `CMQUEUE`, `CMQUEUEX`, `NIIDQUMS` and `NIIDPURG` (see the *TP Monitor Interfaces* documentation).

The specified number can be overwritten by the service modules.

Value	Explanation
`1 - 255`	The number of the alternate TP PCB to be used.
`1`	This is the default value.

ENVPID - Specify Environment Parameter Set

`ENVPID=value` specifies the environment parameter set to be used in the transaction code table.

Value	Explanation
1 - 8 characters	The name of the environment parameter set as defined by the `ENTRYNM` keyword subparameter in the `NTIMSPE` macro.
`ENV00000`	This is the default value.

HCPCB - PCB for Hardcopy

`HCPCB=value` specifies the PCB number to be used for the hardcopy print function.

Value	Explanation
`SYSPCB`	The first alternate TP PCB is used. This is the default value.
`WRKPCB`	One of the additional alternate TP PCBs is used. This enables you to use an express TP PCB for the hardcopy function.

MSGPCB - Message Print PCB

This parameter only applies to message-oriented environments and the server driver.

`MSGPCB=value` specifies the PCB to be used for printing error messages and standard output in a message-oriented environment and the server driver.

Value	Explanation
`SYSPCB`	The first alternate TP PCB is used. This is the default value.
`OWNPCB`	The second alternate TP PCB is reserved and used. This enables you to use an express TP PCB for sending messages.

NRAST - Natural Offset within Scratch Pad Area

`NRAST=`*`value`* defines the offset of the Natural Reserved Area (NRA) within the IMS TM scratch-pad area.

The current length of the NRA is 157 bytes. The length of the NRA can change with the next version of the Natural IMS TM Interface.

Note: If you want to save your own information in the SPA in order to pass it to a non-Natural transaction, we recommended that you save your data in front of the NRA in order to be version compatible.

Value	Explanation
`16 - 32600`	The offset of the NRA within the scratch-pad area.
`16`	This is the default value.

PCBS - PCB Name Assignment

`PCBS=(`*`pcb-1`*`,`*`num-1`*`,`*`pcb-2`*`,`*`num-2`*`,...)` is used to assign a logical name to a PCB.

Value	Explanation
`pcb-n` (1 - 8 characters)	*`pcb-n`* specifies the logical name of the PCB.
`1-255`	This parameter is optional. *`num-n`* specifies the positional number of the PCB in the PSB. If not specified, the sequence number of the PCB in the parameter list is used.

Note: There is no default value.

PSB - PSB Name of IMS TM Transaction

`PSB=value` defines the name of the PSB used by the IMS TM transaction. The PSB name is used to identify the entry in the IMS TM transaction code table for non-message-driven batch message processing and for batch environment.

Value	Explanation
1 - 8 characters	The PSB name that corresponds to the current transaction code. The PSB name must comply with the naming conventions IMS TM uses in the APPLCTN macro. There is no default value.

TRAN - IMS TM Transaction Code

`TRAN=value` specifies the IMS TM transaction code.

The transaction code is ignored in non-message-driven BMP and batch processing environments.

Value	Explanation
1 - 8 characters	A transaction code. The code must comply with the naming conventions IMS TM uses in the `TRANSACT` macro. There is no default value.

TYPE - Natural Transaction Type

`TYPE=value` defines the type of Natural transaction.

Value	Explanation
`CONV`	Conversational Natural session. This is the default value.
`NONC`	Non-conversational Natural session.
`SFE`	Natural Development Server and/or Natural Web I/O Interface server session.

WRKPCBS - Number of Alternate PCBs for Printing

`WRKPCBS=`*`value`* specifies the number of alternate TP PCBs used for printing in addition to the first TP PCB and, if appropriate, to the `MSGPCB`.

Value	Explanation
`0`	No IMS TM printer is available. This is the default value.

`1 - 32`	The number of alternate TP PCBs used for printing.
	Example 1:
	`MSGPCB=SYSPCB`
	`WRKPCBS=2`
	The PSB must contain 3 alternate TP PCBs.
	Example 2:
	`MSGPCB=OWNPCB`
	`WRKPCBS=2`
	The PSB must contain 4 alternate TP PCBs. The second alternate TP PCB is reserved for the error messages and standard output of the message-oriented environment.

Example of NTIMSPT Macro

```
         NTIMSPT TRAN=MYNAT,PSB=MYPSB
```

140 INTENS - Printing of Intensified Fields

This Natural profile parameter specifies how many times an intensified field or the underline character is to be overprinted when it is printed on a print device.

Possible settings	`1 - 10`	Number of times an intensified field or the underline character is overprinted. **Note:** The underline character is printed only if the parameter is set greater than 1. With `INTENS=1`, underlined fields are printed without underlining.
Default setting	`3`	
Dynamic specification	yes	
Specification within session	no	

141 IP - INPUT Prompting Text

This session parameter is used to control prompting text in `INPUT` statements.

Possible settings	`ON`	Even if no text is specified preceding the input/output in an `INPUT` statement, the name of the field will be generated by default as a text element preceding the field as prompting text.
	`OFF`	No automatic prompting text will be generated for input/output fields in an `INPUT` statement. Only fields explicitly preceded with a text element will receive the text as prompting text.
Default setting	`ON`	
Specification within session	yes	
Applicable statements	`FORMAT` `INPUT`	
Applicable command	none	

Example:

```
FORMAT IP=OFF
```

142

IS - Identical Suppress

With this session parameter, you can suppress the printing of identical information in successive lines created by a `WRITE` or `DISPLAY` statement.

Possible settings	`ON`	A value which is identical to the previous value for the field will not be displayed. **Note:** If a `DISPLAY` or `WRITE` statement is used to create multiple output lines using the `VERT` or slash (/) notation, `IS=ON` applies only to the first line.
	`OFF`	No automatic suppression will be used.
Default setting	`OFF`	
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`WRITE`	

Applicable command	none	

Notes:

1. The `IS` parameter setting can be suspended for one record by issuing the `SUSPEND IDENTICAL SUPPRESS` statement.
2. The `IS` parameter may be used in combination with the parameters `ES` and `ZP` to cause empty line suppression.
3. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

Example:

```
FORMAT IS=ON
```

143 ISIZE - Size of Initialization Buffer

This Natural profile parameter specifies the size of the Natural initialization buffer (`ISIZE`).

Possible settings	`8-32767`	Buffer size in KB.
Default setting	`16`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. The `ISIZE` buffer is used to hold the parameters Natural is initialized with, as well as the work areas and tables used by Natural during the initialization.
2. The profile parameter `ISIZE` is ignored if it is specified in a parameter string activated by a `SYS` or `PROFILE` profile parameter or in an alternative Natural parameter module (as specified with the `PARM` profile parameter).

144 ITERM - Session Termination in Case of Initialization Error

This Natural profile parameter specifies whether or not the Natural session is to continue in the case of a session initialization error.

Possible settings	`ON`	If a session initialization error occurs, the session is terminated immediately with message NAT9970 after the initialization error messages.
	`OFF`	If session initialization errors occur, the following happens: ■ In online mode, the initialization errors are displayed, and you can choose to either continue or terminate the session. ■ In batch mode, the initialization errors are displayed, and the session is continued - possibly leading to errors or undesired results later in the session. In case no other errors occur during session execution, the message NAT9964 (with condition code 4) will be issued at session termination instead of NAT9995. **Note:** The setting `ITERM=OFF` is not honored when an `INPL` command is placed on the Natural command stack at the beginning of the Natural session, that is, with `STACK=INPL`. The setting `ITERM=ON` is forced in this case.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Note: The setting of `ITERM` is irrelevant if profile parameter `IMSG` is set to `OFF`, because then any initialization errors are suppressed, and the session continues.

145 ITRACE - Internal Trace Function

This Natural profile parameter is used to activate/deactivate the internal trace function.

Important: Do not use this parameter without prior consultation of Software AG Support.

Possible settings	`ON`	Trace data is passed to the SYSRDC utility.
	`OFF`	No trace data is passed to the SYSRDC utility.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	Within a Natural session, the terminal command `%TRI` can be used to activate/ deactivate the internal trace function.

Note: The internal trace function is intended primarily for Software AG internal use for debugging purposes.

146 KD - Key Definition

This session parameter is used to display the names assigned to the PF keys (see the `SET KEY` statement).

Possible settings	`ON`	The names assigned to the PF keys are displayed.
	`OFF`	The names assigned to the PF keys are not displayed.
Default setting	`OFF`	
Specification within session	yes	
Applicable statements	`FORMAT`	
Applicable command	none	

Notes:

1. The PF key assignment information will always be displayed automatically in the two bottom lines of the physical screen with any output created by the `INPUT`, `WRITE`, `DISPLAY`, and `PRINT` statement.
2. As the key assignment display requires two lines, the logical page size (see the session parameter `PS`) must be reduced by two.

Example:

```
FORMAT KD=ON
```

147 KEY - Setting Assignments to PA, PF and CLEAR Keys

This Natural profile parameter is used to assign settings to the CLEAR key, program attention keys (PA keys) and program function keys (PF keys) on video terminals.

Possible settings	any character string	Settings can be assigned to the keys PA1 to PA3, PF1 to PF24 and to the CLEAR key. **Note:** The setting assigned to each key can be any character string. The character string must represent a Natural system command or a user command (user program). If the setting contains embedded blanks, it must be enclosed in apostrophes.
Default setting	none	
Dynamic specification	no	
Specification within session	yes	
Application programming interface	`USR4005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Assignments made with the profile parameter `KEY` are only valid when specified from the Natural `NEXT` prompt.
2. The entire string specified with the profile parameter `KEY` must be enclosed in parentheses (except `KEY=OFF`). `KEY=OFF` un-assigns all keys.

Examples:

```
KEY=(PF4=OFF,PF1=HELP,PF3='EDIT MAP',PF2=USERPGM1,CLR=LOGOFF)
KEY=OFF
KEY PF4=OFF
KEY PF3="EDIT MAP"
KEY CLR=LOGOFF
KEY OFF
```

148 LC - Lower to Upper Case Translation

This Natural profile parameter controls lower-case to upper-case translation of input characters.

Possible settings	`ON`	No translation of lower-case characters to upper case is performed.
	`OFF`	Natural translates all lower-case characters, except input from the Natural stack which was placed there by the `STACK` statement, to upper case.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	To disable or enable lower-case to upper-case translation dynamically within the active Natural session, you should use the terminal commands `%L` or `%U`
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. This parameter does not apply to Natural stack data which was placed on the Natural stack by the `STACK` statement.
2. Lower/upper-case translation can also be performed by a TP monitor before control is given to Natural. The corresponding TP-monitor parameters for lower/upper-case translation also have to be reviewed to ensure correct translation.
3. A user-supplied translation table can be used to perform translation from lower case to upper case; see `NTUTAB1` macro (contained in the `UTAB1` profile parameter description).

149

LC - Leading Characters

With this session parameter, you can specify leading characters that are displayed immediately before a field output by a `DISPLAY` statement. The width of the output column is increased accordingly.

Possible settings	any character	Up to 10 characters may be specified. **Note:** 1. Leading characters may optionally be specified enclosed within apostrophes, in which case, any characters can be specified. 2. Any character string specified which contains a closing parenthesis or a quotation mark must be enclosed within apostrophes. 3. A circumflex (`^`) is used to represent a blank in a character string not enclosed within apostrophes.
Default setting	none	
Specification within session	yes	
Applicable statements	`FORMAT`	Parameter may be specified dynamically with the `FORMAT` statement.
	`DISPLAY`	Parameter may be specified at statement level and/or at element level.
Applicable command	none	

Notes:

1. The session parameters `LC` and `IC` are mutually exclusive.
2. The parameter LC can also be used with U format fields.
3. For information on Unicode format, see also *Unicode and Code Page Support in the Natural Programming Language, Session Parameters, EMU, ICU, LCU, TCU versus EM, IC, LC, TC*.

4. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

Example:

```
DISPLAY (LC=*)
```

150 LCU - Unicode Leading Characters

With this session parameter, you can specify leading characters that are displayed immediately before a field output by a `DISPLAY` statement. The width of the output column is enlarged accordingly.

Possible settings	any character	Up to 10 characters may be specified. **Note:** 1. Leading characters may optionally be specified enclosed within apostrophes, in which case, any characters can be specified. 2. Any character string specified which contains a closing parenthesis or a quotation mark must be enclosed within apostrophes. 3. A circumflex (`^`) is used to represent a blank in a character string not enclosed within apostrophes.
Default setting	none	
Specification within session	yes	
Applicable statements	`FORMAT`	Parameter may be specified dynamically with the `FORMAT` statement.
	`DISPLAY`	Parameter may be specified at statement level and/or at element level.
Applicable command	none	

Notes:

1. The session parameter `LCU` is identical to the session parameter `LC`. The difference is that the leading characters are always stored in Unicode format. This allows you to specify leading characters with mixed characters from different code pages, and assures that always the correct character is displayed independent of the installed system code page.

2. The session parameters `LCU` and `ICU` are mutually exclusive.

See also:

- *Parameters to Influence the Output of Fields* in the *Programming Guide*
- *Unicode and Code Page Support in the Natural Programming Language, Session Parameters, EMU, ICU, LCU, TCU versus EM, IC, LC, TC* in the *Unicode and Code Page Support* documentation.

151 LE - Reaction when Limit for Processing Loop Exceeded

This Natural profile and session parameter controls the action to be taken if the limit of retrieved records was exceeded in a `READ`, `FIND` or `HISTOGRAM` processing loop.

Possible settings	`ON`	The database loop will be terminated when the limit is reached. The program flow will continue normally with the statement following the terminated database loop. When the execution of the Natural object is complete, error NAT0957 (`Database loop limit reached with 'LE=ON'.`) is raised. **Note:** `LE=ON` applies only to programs which are loaded from a library located in the system file `FUSER`, that is, library `SYSTEM`, or with a (library) name that does not start with the prefix `SYS`.
	`OFF`	The database loop will be terminated when the limit is reached. The program flow will continue normally with the statement following the terminated database loop. When the execution of the Natural object is complete, no error message appears.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The `LE` parameter applies to `READ`, `FIND` and `HISTOGRAM` statements with a limit specified (see *Example*).
2. The limit may be specified either globally for a Natural object by using the `LIMIT` statement or by specifying an explicit limit value supplied in the database processing loop.
3. Within a Natural session, the profile parameter `LE` can be overridden by using the session parameter `LE`.

Example:

```
DEFINE DATA LOCAL
1 EMPL-VIEW VIEW OF EMPLOYEES
  2 NAME
END-DEFINE
READ (10) EMPL-VIEW BY NAME
  WRITE NAME
END-READ
END
```

`LE=OFF`: after 10 records the loop ends without a message.

`LE=ON`: after 10 records the loop ends with an error message NAT0957 (`Database loop limit reached with 'LE=ON'`).

152 LFILE - Logical System File Definition

This Natural profile parameter specifies information concerning the physical database file to be associated with a logical system file for Software AG products. It corresponds to the macro `NTLFILE` in the Natural parameter module.

Possible settings	See *LFILE Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTLFILE` is used instead.
Specification within session	no	
Application programming interface	`USR0011N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.
	`USR2004N` (recommended)	

Notes:

- `LFILE` and `NTLFILE` can be used for Software AG products which have their own system files (for example, Con-nect, Natural Review) to specify where such a system file is to be located. Such products use the database ID 255 and a **logical file number** in their data definition modules (DDMs). With the `LFILE` parameter or the macro `NTLFILE`, you specify which **physical file number** and **database ID** (and, if applicable, password and cipher key) are associated with that logical file number. Natural maps the logical file number to the physical file number and database ID and uses it for any database calls.
- `LFILE` is especially useful for defining a scratch-pad file with the logical file number 212; see also the profile parameter `ROSY`, *Natural Scratch-Pad File* in the *Operations* documentation, and *Defining a Scratch-Pad File* in the *Installation for z/OS*, *Installation for BS2000* and *Installation for z/VSE* documentation.

LFILE Parameter Syntax

The `LFILE` parameter is specified as follows:

LFILE=(*logical-fnr*,*physical-dbid*,*physical-fnr*,*password*,*cipher-key*,[**RO**])

Where:

Syntax Element	Value	Explanation
logical-fnr	`1 - 251`	Logical file number. This parameter is mandatory.
physical-dbid	`0 - 65535`, except `255`	Physical database ID. Database ID `255` is reserved for logical system files for Software AG products.
physical-fnr	`1 - 65535`	Physical file number.
password	1 - 8 characters.	Password and cipher key are only required if the database file has been password-protected and/or ciphered using the Adabas security feature.
cipher-key	8 numerical digits.	
`RO`	n/a	Flag for read-only access.

Note: To define different logical files, the `LFILE` parameter must be specified multiple times (separated by a comma or a blank); see *Example of LFILE Parameter*.

NTLFILE Macro Syntax

The `NTLFILE` macro is specified as follows:

```
        NTLFILE logical-fnr,physical-dbid,physical-fnr,password,cipher-key,RO
```

Notes:

1. For an explanation of the syntax elements and for possible values, see *LFILE Parameter Syntax*.
2. To define different logical files, the `LFILE` parameter or the `NTLFILE` macro must be specified several times; see *Examples of NTLFILE Macro*.

Example of LFILE Parameter

```
LFILE=(180,73,10),LFILE=(251,40,9,TEST99)
```

Examples of NTLFILE Macro

```
         NTLFILE 180,73,10
         NTLFILE 251,40,9,TEST99
```

153 LIBNAM - Name of External Program Load Library

This Natural profile parameter specifies the name of the load library from which programs are to be loaded dynamically when Natural is used under BS2000, z/OS batch mode, or TSO.

Possible settings	character string	Any valid BS2000 file name, or 8-byte `DDNAME` of load library
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies under BS2000, z/OS batch mode, and TSO.
2. Under z/OS, a JCL statement with a `DDNAME` that equals the `LIBNAM` setting also needs to be specified. By default, programs are loaded from the job steplib.

154 LOG (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

155 LS - Line Size

This Natural profile and session parameter specifies the maximum number of characters permitted per line for `DISPLAY`, `INPUT` and `WRITE` statements.

The following topics are covered below:

Profile Parameter LS

When used as a profile parameter, `LS` is honored in batch mode only and defines the physical line size. In online mode, the line size is always set to the physical screen width.

Possible settings	`35 - 250`	Maximum number of characters permitted per line.
	`0`	Use physical line size (mostly 132).
Default setting	`0`	
Dynamic specification	yes	

Session Parameter LS

Possible settings	`2 - 250`	Maximum number of characters permitted per line.
	`0`	Only permitted with the statement `SET GLOBALS` or with the system command `GLOBALS`. The value `0` will be replaced by the physical line size.
Default setting	Physical line size.	
Applicable command	`GLOBALS`	

Applicable statements	`FORMAT`
	`SET GLOBALS`	

Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. At logon to a library, `LS` is reset to the physical line size.
2. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.

Specification with Statements

When specified with a statement, the `LS` parameter is evaluated at compilation time.

Applicable statements	`DISPLAY`
	`INPUT`
	`WRITE`
		Parameter may be specified at statement level.

156

LT - Limit for Processing Loops

This Natural profile and session parameter is used to limit the number of database records which can be retrieved within Natural applications.

Possible settings	`1 - 2147483647`
	`0`
		Maximum number of records that can be retrieved. All retrieved records (including records rejected by means of a `WHERE` clause) are counted and compared with this limit.
		`LT=0` defines that no limit is in effect for the number of retrieved records.
		Note: Within a session, you can specify a value in the range of `0` to *`n`*, where *`n`* is the value of profile parameter `LT` at session start.

Default setting	`99999999`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	**Note:** When the `LT` parameter is used in conjunction with the statement `SET GLOBALS` or the system command `GLOBALS`, the limit value that can be set may not exceed the `LT` value defined in the Natural parameter module.
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The limit set with the `LT` parameter applies to all statements retrieving records from the database; that is, statements that initiate processing loops, such as `READ`, `FIND`, `HISTOGRAM` or `SELECT`, and statements that retrieve only a single record, such as `FIND UNIQUE`, `FIND NUMBER`, `FIND FIRST`, `GET` (`SAME`) and `SELECT SINGLE`.
2. All retrieved records are counted and the result of the count is compared with the `LT` limit. The count also includes those records which were rejected by a `WHERE` clause of a `FIND`, `READ` or

`HISTOGRAM` statement. The `LT` limit does not affect the statements `STORE`, `UPDATE`, `DELETE`, `END TRANSACTION` and `BACKOUT TRANSACTION`.

3. When a record is retrieved from the database, the count of retrieved records is incremented before it is compared with the current value of the `LT` parameter. If the incremented count exceeds the current `LT` value, Natural error NAT1003 (`Global limit for database calls reached`) is raised. The count of retrieved records is reset to zero whenever a Natural program is started on Level 1. The count is not reset if the program on Level 1 invokes another Natural object (for further information, see *Multiple Levels of Invoked Objects* in the *Programming Guide*). Therefore, the `LT` parameter limits the number of records retrieved from the database by a Level 1 program and objects invoked by that program on a level other than 1.
4. If the value of the `LT` parameter is dynamically changed within a program by using a `SET GLOBALS LT=`*n* statement, the new limit value becomes effective for the next statement that retrieves a record from the database.
5. Within a Natural session, the profile parameter `LT` can be overridden by using the session parameter `LT`.

157 MADIO - Maximum DBMS Calls between Screen I/O Operations

This Natural profile parameter is used to specify the maximum number of DBMS calls permitted between two screen I/O operations (also in batch mode).

Possible settings	`30 - 65535`	Maximum number of DBMS calls.
	`0`	`MADIO=0` indicates that no limit is to be in effect.
Default setting	`512`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation. * Recommended.
	`USR1068N` *	

Note: If the specified limit is exceeded, the Natural program is interrupted and the user is notified with Natural error message NAT1009.

158 MAINPR - Override Default Output Report Number

This Natural profile parameter can be used to separate program output from Natural system output, which may be useful particularly in batch mode.

Possible settings	`0 - 31`	Valid printer number.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR6002N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. This applies to program output for Report 0, as produced by `DISPLAY`, `PRINT`, `WRITE` or `INPUT` statements, except `INPUT` statements which contain non-protected input fields (field attribute specification `AD=A`) or modifiable input fields (`AD=M`).
2. If the `MAINPR` parameter is specified, program output for Report 0, which would normally be output on the printer assigned to Report 0, is output on the printer specified with `MAINPR` instead; while system output (`NEXT` prompt, `DATA` prompt, etc.) is always output on the primary output device (Report 0); the `MAINPR` setting must be a valid printer number (0 - 31).
3. A logical printer corresponding to the report number specified must be defined to Natural. A printer is defined with the profile parameter `PRINT`, with the macro `NTPRINT` or automatically by JCL (in batch mode or under TSO).
4. The `MAINPR` parameter does not apply to output from system programs in the Natural system library `SYSLIB`, which is always output on the primary output device (Report 0). However, you can use the `USEMAINPR` option of the `LIST` system command to route the output to the printer specified with `MAINPR`. `USEMAINPR` is described in *Settings* in the *System Commands* documentation.

159 MAXCL - Maximum Number of Program Calls

This Natural profile parameter is used to specify the maximum number of program calls permitted between two screen I/O operations.

Possible settings	`10 - 65535`	Maximum number of program calls.
	`0`	`MAXCL=0` indicates that no limit is to be in effect.
Default setting	`50`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation. * Recommended.
	`USR1068N` *	

Note: If the specified limit is exceeded, the Natural program is interrupted and the user is notified with an appropriate Natural error message (NAT1029).

160 MAXROLL - Number of CMROLL Calls before Session Suspension

It specifies the number of `CMROLL` calls after which a Natural session is suspended, that is, a potential roll-out of the Natural thread is to be performed.

Possible settings	`1 - 32767`	Number of `CMROLL` calls.
	`0`	`MAXROLL=0` indicates that no conditional `CMROLL` requests are issued.
Default setting	`128`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter is applicable only under Com-plete and CICS.
2. The `MAXROLL` parameter can be used to control the frequency of conditional `CMROLL` requests. For example, `MAXROLL=128` means that a conditional `CMROLL` request is issued after every 128th statement at compilation.
3. In certain cases, the Natural nucleus issues a conditional `CMROLL` request (wait time = 0), particularly at compilation after each statement. This is done to reset the CPU time window (under Com-plete) in order to avoid an automatic cancel due to the CPU time limit being exceeded; however, this has a negative impact on performance.
4. Note concerning CMROLL: Calling `CMROLL` is the Natural interface for `WAIT` or `DELAY` functionality (see also sample Natural program `SUSPEND` in library `SYSEXTP`); when calling `CMROLL`, you may pass a delay interval/wait time as parameter. When a session has to wait in `CMROLL`, shared resources as a thread in Com-plete or a shared thread in CICS (`THREADS=`*`nonzero`*) are released, and as a consequence a potential roll-out of the Natural thread is performed. Calling `CMROLL` with a delay interval of 0 is called conditional, as the session actually needs not wait for a certain time; however, when other sessions are waiting for a thread, the session is suspended, which

may result in a roll-out of the Natural thread. In CICS if no other session is waiting, just an `EXEC CICS SUSPEND` is executed to prevent AICA abends.

161 MAXYEAR - Maximum Year for Date/Time Values

This Natural profile parameter sets the maximum value for the year part of date and time values that can be entered as constants or as terminal input.

Possible settings	`2699`	The maximum year that can be entered is `2699`; that is, the maximum date value that can be entered is `2699-12-31`.
	`9999`	The maximum year that can be entered is `9999`; that is, the maximum date value that can be entered is `9999-12-31`.
Default setting	`2699`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. `MAXYEAR=9999` changes the maximum date value that can be entered from `2699-12-31` to `9999-12-31`.
2. Before setting the value for `MAXYEAR` to `9999`, you should carefully check your application for arithmetic operations or assignments of date or time values to fields that have data formats other than date or time, and perform the necessary changes. Otherwise, unexpected overflows leading to Natural errors at execution time may occur.

For example, you should check for

- redefinitions of date/time fields with P6/P12 fields
- assignments of date/time values to non-date/time fields such as `P6 := D`
- arithmetic operations with date/time values where the result is assigned to a non-date/time field, for example: `P6 := D + 7`
- input of date/time fields that is used in arithmetic operations with non-date/time fields later on, for example:

```
INPUT D(D)
P6 := D + 1
```

The use of the Natural Engineer is recommended to check your application.

The setting of `MAXYEAR` affects

- checking of date/time constants by the compiler, for example: `P6 := D'2699-12-31'`
- `INPUT` statements with input or modifiable date/time fields
- `MOVE EDITED` statements with source or target date/time fields
- `IS (D)` option in logical condition criteria
- `MASK` option in logical condition criteria with four-digit year check `(YYYY)`
- `VAL` system function with date field as target operand

You should ensure that the `MAXYEAR` settings are the same for

- cataloging and executing a Natural application
- Natural RPC servers and Natural RPC clients

See also:

- *Formats D - Date, and T - Time* in the *Programming Guide*
- *Date and Time Constants* in the *Programming Guide*
- Session parameter `EM` in the *Parameter Reference* documentation
- Profile parameter `YD` in the *Parameter Reference* documentation

162 MC - Multiple-Value Field Count

With this session parameter, you determine the number of values of a multiple-value field to be output by default when the field is specified without an index in a `DISPLAY` or `WRITE` statement.

Possible settings	`0 - 191`	Number of values. **Note:** If `MC=0` is specified, then there is no default index range for the output of an `MU` field. Therefore, when an `MU` field is output, it is necessary to specify an explicit index or index range. Otherwise, a syntax error (NAT0281) will be raised.
Default setting	`1`	
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`INPUT`
	`PRINT`
	`WRITE`	

Applicable command	none	

Note: This parameter may be used in reporting mode only.

Example:

```
FORMAT MC=5
```

163 MENU - Menu Mode

This Natural profile parameter is used to enable or disable Natural menu mode.

Possible settings	`ON`	Menu mode is enabled.
	`OFF`	Menu mode is disabled.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	Within a Natural session, the `MENU` parameter can be overridden by the Natural system command `MAINMENU` (described in the System Command documentation).

164 ML - Position of Message Line

This profile parameter specifies the line to be used for the display of applications which do not set the message line position explicitly by using the `SET CONTROL 'M'` statement.

Possible settings	`B`	Natural messages are displayed at the bottom of the screen.
	`T`	Natural messages are displayed at the top of the screen.
Default setting	`T`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET CONTROL 'M'`	
Applicable command		
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. For information on the operand `'M'`, see the Natural terminal command `%M` (Control of Message Line).
2. Within a Natural session, the profile parameter `ML` can be overridden by the session parameter `ML`.

165 MONSIZE - Size of SYSTP Monitor Buffer

This Natural profile parameter specifies the size of the buffer used by the Monitor function of the SYSTP utility (described in the *Utilities* documentation).

Possible settings	`5-256`	Buffer size in KB.
	`0`	If `MONSIZE=0` or if the requested space is not available, the Monitor function of the SYSTP utility cannot be used, except there is a monitor buffer pool defined by means of profile parameter `BPI` or parameter macro `NTBPI`.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Note: Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.

166 MP - Maximum Number of Pages of a Report

This Natural profile and session parameter specifies the maximum number of pages to be produced for a report.

Possible settings	`1 - 99999`	The value specified is the number of physical pages and has no effect on the starting page number used. The program will be terminated with an error message if the `MP` value is exceeded.
	`0`	No page limit is defined.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Applicable statements	`DISPLAY`
	`FORMAT`
	`PRINT`
	`WRITE`	

Applicable command	none	

Note: Within a Natural session, the setting of profile parameter `MP` can be reduced, but not increased by the `FORMAT` statement. The value specified with the session parameter `MP` applies only to the specified report.

167 MS - Manual Skip

With this session parameter, you control the cursor positioning during the processing of an `INPUT` statement.

Possible settings	`ON`	See example below. **Note:** The setting `MS=ON` is not supported under BS2000.
	`OFF`	The cursor will be positioned to the next input field as soon as the value for the current field is entered with all positions.
Default setting	`OFF`	
Specification within session	yes	
Applicable statements	`FORMAT` `INPUT`	
Applicable command	none	

Example:

```
INPUT (MS=ON) #A #B
```

168 MSGSF - Display System Error Messages in Short/Full Format

This Natural profile parameter can be used to avoid truncation of Natural system error messages.

Possible settings	`ON`	System error messages will be displayed in full; that is, program name, line number and actual message text.
	`OFF`	System error messages will be displayed in short form; that is, only the actual message text will be displayed (but not the program name and line number).
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	Within a Natural session, the profile parameter `MSGSF` can be overridden by the Natural terminal command `%MSGSF`.

By default, a Natural system error message consists of the following:

- the name of the program,
- the number of the line that caused the error,
- the actual text of the message.

Depending on the size of the window in which the message is displayed, the text may be truncated. With this parameter, you can avoid such truncation.

169 MT - Maximum CPU Time

This Natural profile and session parameter determines the maximum amount of CPU time which can be used by a Natural program.

⚠ **Important:** In server environments where the server itself runs without any operating system controlled CPU time limit, it is strongly recommended to set the profile parameter `MT` to a non-zero value to prevent the formation of endless loops caused e.g. by application errors. This recommendation applies to Natural RPC and Natural Development Server servers.

Possible settings	`1 - 9999999`	Maximum amount of CPU time in seconds. **Note:** 1. If Natural Security is installed, the profile parameter `MT` can be overridden within Natural Security. 2. With Natural Security, the maximum value for the profile parameter `MT` is `32767`. 3. To use a higher value as specified with the `MT` profile or session parameter, specify `MT=0` within Natural Security.
	`0`	`MT=0` defines that no Natural CPU time limit is to be in effect.
Default setting	`60`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. This Natural profile and session parameter only applies to programs executed in batch mode, under Natural Development Server (SPoD) or under Natural for TSO.
2. CPU time measurement starts when a Natural program is started from `NEXT` mode or by means of a `FETCH` statement, that is, on program level 1. In non-batch mode (Natural Development Server, Natural for TSO), CPU time measurement is restarted at every terminal I/O.
3. The limit for programs operating in interactive mode is controlled by the TP monitor in use.
4. The maximum value that can be used is determined by the operating system environment. Any setting in excess of the maximum is reduced to the maximum supported by the operating system.
5. In system environments which do not support CPU time measurement, the limit is interpreted as elapsed time. The CPU time limit is ignored for systems without timer support.
6. Within a Natural session, the `MT` profile parameter can be overridden by the session parameter `MT`.
7. When running zIIP-enabled under z/OS, the `MT` profile parameter applies separately to both TCB (except under CICS) and SRB (zIIP) processing modes. If `MT=0` is set in SRB mode, Natural uses the existing z/OS TCB time limit to avoid endless loops because there is no z/OS CPU time limit (for example, the JCL TIME parameter) for SRBs.
8. The `MT` parameter is supported under CICS SRBs (zIIP), but not under CICS TCBs. If `MT=0` is set, Natural uses the CICS runaway time as CPU time limit in SRB mode. A CPU timeout abend that occurs when a CICS open TCB is used cannot be recovered. It causes an immediate AICA abend of the CICS task and the Natural session is aborted.

170 NAFSIZE - Size of Buffer for Natural Advanced Facilities

This Natural profile parameter specifies the size of the work buffer used by Natural Advanced Facilities.

Possible settings	`1 - 64`	Buffer size in KB.
	`0`	`NAFSIZE=0` disables Natural Advanced Facilities.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies if Natural Advanced Facilities is installed.
2. Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.
3. If Natural Advanced Facilities is to be used, a setting has to be specified for this parameter; see *NATSPOOL Initialization* and further information on the `NAFSIZE` parameter in *Natural Profile Parameters for NATSPOOL* in the *Natural Advanced Facilities* documentation.
4. If the requested space is not available, Natural Advanced Facilities cannot be used.

171 NAFUPF - Natural Advanced Facilities User Profile

This Natural profile parameter is used to specify the user-profile name for Natural Advanced Facilities.

Possible settings	1 - 8 characters	Name of the user profile.
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies if Natural Advanced Facilities is installed.
2. See *NATSPOOL Initialization* in the *Natural Advanced Facilities* documentation.

172 NC - Use of Natural System Commands

This Natural profile and session parameter controls whether Natural system commands can be used during the Natural session or not.

Possible settings	`ON`	System commands cannot be used. **Exceptions:** `FIN`, `LAST`, `LOGOFF`, `LOGON`, `MAINMENU`, `RENUMBER`, `RETURN`, `SETUP` and `TECH`. **Note:** 1. If you have Natural Security installed, any system command restrictions you set with Natural Security are valid, regardless of the setting of the `NC` profile parameter. 2. In a Natural Development Server environment on mainframe computers, the value `OFF` will be assumed for the Natural Development Server, even if `NC=ON` has been specified. 3. If `NC=ON` has been specified on the client side, subsequent system commands issued on the client side will be rejected as described above.
	`OFF`	All system commands can be used.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET` `GLOBALS`	
Applicable command		

Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Within a Natural session, the profile parameter `NC` can be overridden by the session parameter `NC`.
2. Natural terminal commands and user-created commands (object module names) are not affected by the `NC` parameter.

173 NISN (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

174 NL - Numeric Length for Output

This session parameter determines the default input/output length for a numeric field used in a `DISPLAY`, `INPUT`, `PRINT` or `WRITE` statement.

Possible settings	*`n.m`*	The length is specified as *`n.m`*, where *`n`* represents the number of positions before the decimal separator, and *`m`* represents the number of positions after the decimal separator. The *`m`* notation is optional. The total of *`n`*+*`m`* must not exceed `29`. **Note:** 1. If `NL` is set less than the field length, values are truncated. No error is produced when relevant digits are truncated. 2. If `NL` is set greater than the field length, values are expanded with blanks. No error is produced when an input field is truncated.
Default setting	none	
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`INPUT`
	`PRINT`
	`WRITE`	

Applicable command	none	

Notes:

1. The `NL` parameter must not be specified for groups.
2. Any edit mask specified for a field will override the `NL` parameter for this field.

3. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

Example:

```
DISPLAY #AA(NL=20) #AB(NL=3.2)
```

175 NUCNAME - Name of Environment-Independent Nucleus

This Natural profile parameter specifies the name of the environment-independent Natural nucleus if it is to be loaded dynamically and not linked to the environment-dependent Natural nucleus.

Possible settings	1 - 8 characters	Valid load module name.
Default setting	none	
Dynamic specification	yes	**Note:** By specifying this parameter dynamically, you are able to use different environment-independent Natural nuclei (for example, for production and for testing) together with the same environment-dependent Natural nucleus without having to relink the nucleus.
Specification within session	no	

Notes:

1. The profile parameter `NUCNAME` does not apply under BS2000.
2. The profile parameter `NUCNAME` is ignored if it is specified in a parameter string activated by a `SYS` or `PROFILE` profile parameter or in an alternative Natural parameter module (as specified with the `PARM` profile parameter).

For further information, see:

- *Environment-Independent Nucleus* in the *Installation for z/OS* documentation.
- *Environment-Independent Nucleus* in the *Installation for z/VSE* documentation.

176 O4I - Collect Data for Optimize for Infrastructure

This Natural profile parameter is used to control the performance data collection in the Optimize Monitor Buffer Pool for Optimize for Infrastructure.

Possible settings	`ON`	Performance data is collected and passed to Optimize for Infrastructure.
	`OFF`	No performance data is collected for Optimize for Infrastructure.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Note: For information on how to start the Optimize Monitor Buffer Pool, see *Optimize Monitor Buffer Pool* in the Natural *Operations* documentation.

177 OBJIN - Use of CMOBJIN as Natural Input File

This Natural profile parameter indicates whether the `CMOBJIN` file, see *Natural in Batch Mode* in the *Operations* documentation is to be used for input data provided with the `INPUT` statement in batch mode.

Possible settings	`Y`	Data for a Natural `INPUT` statement are read from the `CMOBJIN` file.
	`N`	The `CMOBJIN` file is not used and any data for an `INPUT` statement are read from the `CMSYNIN` file.
	`R`	Natural determines which option has been selected for a particular session by the presence or absence of the `CMOBJIN DD/FILE` statement in the Natural execution JCL/JCS.
Default setting	`R`	
Dynamic specification	yes	
Specification within session	yes	

Note: This Natural profile parameter only applies to batch mode.

178 OPF - Overwriting of Protected Fields by Helproutines

This Natural profile and session parameter specifies whether the content of a write-protected field (attribute definition `AD`=`P`) can be overwritten by a helproutine assigned to the field.

Possible settings	`ON`	A helproutine assigned to a field can overwrite the field's content, even if the field is write-protected.
	`OFF`	Helproutines cannot overwrite the contents of write-protected fields.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The `OPF` profile parameter only applies to the field for which a helproutine is invoked; it does not affect parameters explicitly passed to the helproutine. This means that the `OPF` profile parameter takes no effect if the field for which help is invoked is also explicitly specified as a parameter to be passed to the helproutine.
2. In addition, in reporting mode you can change the `OPF` setting using the statement `SET GLOBALS`.
3. Within a Natural session, the profile parameter `OPF` can be overridden by the session parameter `OPF`.

179

OPRB - Database Open/Close Processing

This Natural profile parameter controls the use of database open/close commands during a Natural session that accesses an Adabas or a VSAM database.

The `NTOPRB` macro can be used as an alternative to the profile parameter `OPRB` in the `NTPRM` macro. The maximum length of an `OPRB` parameter specification is 256 bytes. If you require a longer specification, use the `NTOPRB` macro instead of the `OPRB` parameter.

Possible settings	See ***OPRB Parameter Syntax***.	
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. The Natural profile parameter `OPRB` and the corresponding macro `NTOPRB` only apply to Adabas and VSAM databases.
2. Generally, the `OPRB` parameter uses one of the above syntaxes (the possible contents of the *`strings`* depend on the database system).
3. Instead of using the `OPRB` parameter, you can also use the macro `NTOPRB` in the Natural parameter module.
4. If you wish to make `OPRB` specifications that are to apply to all databases, it is strongly recommended that you use the `OPRB` parameter in the `NTPRM` macro (and not an `NTOPRB` macro).

OPRB Parameter Syntax

The `OPRB` parameter is specified as follows:

Variant 1 - Open Request for All Databases

OPRB=(*`string`*)

With this syntax you specify an open request for *all* databases.

Variant 2 - Open Request for Specific Databases

`OPRB=(DBID=nn1,string,DBID=nn2,string,...)`

With this syntax you specify an open request for specific individual databases. As defined in the macro `NTDB`, the specified DBID identifies the type of database.

Variant 3 - Open Request for Specific Databases and Default Open Request for All Others Not Specified Explicitly

`OPRB=(string1,DBID=nn1,string2,DBID=nn2,string3,...)`

With this syntax you specify an open request for specific individual databases (*`string2`* and *`string3`*) and also a default open request - the initial *`string1`*- which applies to all databases for which you do not specify an individual string.

Variant 4 - Open Request Using a (Non-)Restricted Call

`OPRB=(DBID=nn1,NR=value,string,...)`

With the subparameter `NR=value` you specify whether the Adabas open command `OP` is to be executed as a restricted or non-restricted call. This controls the value set in the Command Option 1 (`COPT1`) of the open command `OP`.

Where:

Value	Explanation
`NR=OFF`	Causes a restricted open, with `COPT1=R`. This is the default value.
`NR=ON`	Causes a non-restricted open, when `COPT1` is left empty.

Dynamic OPRB with Natural Security

A dynamically specified `OPRB` parameter applies for all logons to libraries in whose security profiles no `OPRB` parameter is specified. For a logon to a library in whose security profile the `OPRB` parameter is specified, any dynamically specified `OPRB` parameter is ignored, and the one from the security profile applies.

OPRB for VSAM

The `strings` which can be specified for VSAM databases are described under *OPRB Parameter for VSAM Databases* in the *Natural for VSAM* documentation.

OPRB for Adabas

For Adabas databases, the `OPRB` parameter is required if either of the following conditions are true for the Natural session:

- An explicit list of Adabas files to be accessed/updated is to be provided. This is necessary, for example, if Adabas cluster updating or exclusive file control is to be requested.
- A single logical transaction is to span two or more Natural programs and, therefore, it is not desired to have Natural issue an `END TRANSACTION` and `CL` (close) command at the termination of any given Natural program.

Possible Content of Parameter String	**Explanation**
`ACC=(`*`file-list`*`)`	Specifies access permission (read) for the files in the file list.
`UPD=(`*`file-list`*`)`	Specifies access/update permission (read/write) for the files in the file list.
`EXF=(`*`file-list`*`)`	Specifies exclusive file control: no other users may access/update the file.
`EXU=(`*`file-list`*`)`	Specifies exclusive update permission (exclusive read/write) for the files in the file list.
`ACODE`	Specifies the option to enforce a user encoding for A fields. **Note:** The required encoding code for `ACODE` is derived from the current `CP` parameter setting of the Natural session.
`WCODE`	Specifies the option to enforce a user encoding for W fields. **Note:** The required encoding code for `WCODE` is always `4095`.
`ARC`	Defines a special data architecture for fields in the record and value buffers. This definition overrides the architecture key defined for remote calls in Entire Net-work.

Notes:

1. For further information on these settings, refer to the description of the Adabas `OP` command in the Adabas *Command Reference* documentation.
2. If the `OPRB` parameter is omitted in the Natural parameter module or `OPRB=OFF` is specified as a dynamic parameter, a Natural session commences with an Adabas open command requesting

`UPD` (access/update) to the Natural system file. Natural also issues `RELEASE CID` (Adabas `RC`) commands to release all ISN lists (ISN lists specified in a `RETAIN` clause of a Natural `FIND` statement are not released).

The Adabas record buffer to be used with the initial Adabas `OP` command can be explicitly provided. The format is similar to that used in an Adabas record buffer for the `OP` command with the exception that no blanks can be embedded, and the complete setting must be enclosed in parentheses (not apostrophes).

Example 1:

```
OPRB=(ACC=2,4,6,UPD=8.)
```

This specifies that Adabas Files 2,4 and 6 are to be made available for access only and that Adabas File 8 is to be made available for update (which also implies access).

Example 2:

```
OPRB=(EXU=1,2,3.)
```

This specifies that Adabas Files 1,2 and 3 are to be placed under exclusive control for this Natural session.

Combinations of the keywords `ACC`, `UPD` and `EXU` must follow the rules as defined in the relevant Adabas documentation. When these keywords are coded, Natural issues an `OP` command at the start of a Natural session and a `CL` at the end of the Natural session. At the end of a Natural program, only the required `RC` commands are issued to release held ISN lists.

In all of the above situations, the `OP` command, which is always issued at the start of a Natural session, contains the user ID for the Natural session in the Additions 1 field of the Adabas control block. In batch mode, this is the job name. In TP mode, this is the setting supplied at system initialization by the Natural interface module. In both cases, the setting used is available in the Natural system variable `*INIT-USER`.

NTOPRB Macro Syntax

The syntax of the `NTOPRB` macro is as follows:

```
         NTOPRB dbid,'string'
```

Notes:

1. For possible values, see the `OPRB` parameter; if you use Natural with VSAM, see also the *Natural for VSAM* documentation.
2. If `string` is very long, it can be divided in up to five strings separated by commas (see examples below), as the Assembler allows single strings up to 256 bytes only.

Examples of NTOPRB Macros

```
         NTOPRB 12,'ACC=40,UPD=20'
         NTOPRB 15,'EXU=1,','2,3'
```

180

OPT - Control of Natural Optimizer Compiler

This parameter activates/deactivates the Natural Optimizer Compiler and controls the various options related to it. It corresponds to the macro `NTOPT` in the Natural parameter module.

Possible settings	See *Dynamic Profile Parameter OPT* in the Natural *Optimizer Compiler* documentation.	
Default setting	`OFF`	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTOPT` is used instead.
Specification within session	yes	

Note: This Natural profile parameter only applies if the Natural Optimizer Compiler is to be used.

The following topics are covered below:

OPT Parameter Syntax

The parameter syntax of `OPT` is, for example, as follows:

```
OPT=(INDX,OVFLW,ZD=OFF)
```

For more syntax examples, refer to *Dynamic Profile Parameter OPT* in the Natural *Optimizer Compiler* documentation.

NTOPT Macro Syntax

The syntax of the `NTOPT` macro is, for example, as follows:

```
         NTOPT 'INDX,OVFLW,ZD=OFF'
```

For more syntax examples, refer to *Macro NTOPT* in the Natural *Optimizer Compiler* documentation.

181 OSP - Parameters for z/OS Batch

The parameters for z/OS batch can be specified as subparameters of profile parameter `OSP` or macro `NTOSP`.

Possible settings	See *OSP Parameter Syntax*.	
Default setting	See *Keyword Subparameters*.	
Dynamic specification	yes	The parameter `OSP` can only be specified dynamically. In the Natural parameter module, use the macro `NTOSP`.
Specification within session	no	

OSP Parameter Syntax

The `OSP` parameter is specified as follows:

OSP=(*keyword-subparameter=value,keyword-subparameter=value*,...)

For information on subparameter names and values, see ***Keyword Subparameters***.

NTOSP Macro Syntax

The `NTOSP` macro is specified as follows:

```
         NTOSP ABEXIT=value,                                           *
               DUMPDSN=value,                                          *
               LBPNAME=value,                                          *
               LEHDLR=value,                                           *
               SUBPOOL=value,                                          *
               TIOBSZ=(value1,value2),                                 *
               USERID=value
```

See ***Keyword Subparameters***.

Keyword Subparameters

`ABEXIT` | `DUMPDSN` | `LBPNAME` | `LEHDLR` | `SUBPOOL` | `TIOBSZ` | `USERID`

ABEXIT - Abend Processing

`ABEXIT=`*`value`* specifies the mode of abend processing within Natural.

Value	Explanation
`ESTAE`	Natural intercepts all abends and issues the appropriate error messages. This is the default value.
`SPIE`	Only program checks (`S0C`*`x`* abends) are intercepted.
`OFF`	Natural does not intercept any abends or program checks at all. This value corresponds to profile parameter `DU=FORCE`.

Note: The setting `ABEXIT=OFF` is not recommended because some functions, which require the abend interception, will not work any longer. The usage of profile parameter `MT` will cause an abend `U0322` instead of error NAT0953 when the CPU time limit is reached.

DUMPDSN - Dump Data Set Name Prefix

`DUMPDSN=`*`value`* can be used to define the prefix for a dynamically allocated dump data set. Then each dump will be written by the z/OS service `IEATDUMP` to a separate data set instead of the standard data set (`SYSUDUMP` or `SYSMDUMP`). This can be very helpful in batch server environments when multiple dumps from different Natural sessions must be written.

The complete dump data set name will be as follows:

```
value.D&YYMMDD.T&HHMMSS.&SYSNAME.&JOBNAME
```

Value	Explanation
1 - 8 characters	High level qualifier for the dump data set name.
' ' (blank)	No dumps are written by means of `IEATDUMP`. Instead, any dumps are written to the standard dump data set (`SYSUDUMP` or `SYSMDUMP`) if allocated. This is the default value.

LBPNAME - Sharing of Local Buffer Pools

`LBPNAME=`*`value`* controls the sharing of the local buffer pools when running multiple Natural sessions within the same region.

Value	Explanation
1 - 8 characters	Name of shared local buffer pool environment.
' ' (blank)	The local buffer pools are not shared. This is the default value.

Notes:

1. `LBPNAME` defines the name of the shared local buffer pool environment, and is used to locate the shared local buffer pool.
2. When running multiple Natural sessions in a z/OS batch or TSO region concurrently, each session allocates storage for a separate local buffer pool. Except for the Natural z/OS batch mode server, the local buffer pools are not shared by default, that is, if the different sessions use the same Natural objects, these have to be loaded once for each session separately. If a name is specified, all Natural sessions will share the same local buffer pool.

LEHDLR - Use of an LE Error Handler for Calling LE Subprograms

`LEHDLR=`*`value`* specifies whether Natural uses an LE error handler for the call of LE subprograms.

Value	Explanation
`ON`	An LE error handler is set up by Natural during the call of LE subprograms. This means, if an unhandled LE error occurs during the execution of an LE subprogram, Natural will get control and can handle it (by issuing error NAT0954). This is the default value.
`OFF`	No set-up of an LE error handler is done by Natural during the call of LE subprograms. This means, if an unhandled error occurs during the execution of a LE subprogram, the LE enclave is terminated and so the Natural session is lost.

Notes:

1. For information on LE runtime options, see the description of source module `NATLEOPT` in the *Installation for z/OS* documentation.
2. For information on Natural running with the IBM Language Environment, refer to *Natural Execution - Miscellaneous Topics*, *LE Subprograms* in the *Operations* documentation.

SUBPOOL - Storage Subpool for GETMAIN Requests

`SUBPOOL=value` specifies the storage subpool for `GETMAIN` requests.

Value	Explanation
`1 - 127`	Subpool number.
`0`	This is the default value.

Notes:

1. The subparameter `SUBPOOL` is honored only in the Natural parameter module which is linked to the batch driver, but not in an alternative parameter module which is activated using a `PARM=` specification.
2. As the subparameter `SUBPOOL` is evaluated during session initialization only, it cannot be specified as a dynamic subparameter.

TIOBSZ – Size of the Primary I/O Buffer for Batch Processing

`TIOBSZ=(value1,value2)` specifies the size of the primary I/O buffer for batch and/or server processing.

Value	Explanation
	Size of the primary I/O buffer in KB.
value1	Batch size: `4 - 32` KB. It is allocated below the 16 MB line.
value2	Server size: `32 - 999999` KB. It is allocated above the 16 MB line.
`(8,64)`	This is the default setting.

Alternative Specifications:

- `TIOBSZ=10` or `TIOBSZ=(10)` defines only the batch size.
- `TIOBSZ=(,33)` defines only the server size. The value which is not specified will remain unchanged.

USERID - Content of System Variable *INIT-USER

`USERID=value` specifies the content of the system variable `*INIT-USER`.

Value	Explanation
`ON`	The variable is set to either the user ID from the security access control block (ACEE) if a security package (as RACF or ACF2) is involved or the `USER` parameter from the job card.
`OFF`	The user ID is the job name. This is the default value.

Notes:

1. The content of `*INIT-USER` can be changed by the user ID exit `NATUEX1` during session initialization.
2. For further information, see *Configuring Natural*, *Natural User Exits*, *NATUEX1 - User Exit for Authorization Control* in the *Operations* documentation.

Example of OSP Parameter

```
OSP=(LBPNAME=NATTEST1,USERID=ON)
```

Example of NTOSP Macro

```
         NTOSP LBPNAME=NATTEST1,USERID=ON
```

182 OUTDEST - Output Destination for Asynchronous Processing

This Natural profile parameter specifies the destination to which any Natural error message produced by an asynchronous application is to be sent.

Possible settings	1 - 8 characters.	Destination to which a Natural error message is sent.
Default setting	Setting of profile parameter `SENDER`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Natural under CICS, Com-plete and *open*UTM.
2. After an error message has been sent, Natural terminates the asynchronous session.
3. Under *open*UTM, this parameter is used to specify the ID of the terminal where output from an asynchronous application is to be displayed.
4. When and how error messages/output from an asynchronous application are output depends on the respective TP monitor.

For further information, see:

- *Asynchronous Natural Processing under CICS*
- *Asynchronous Natural Processing under Com-plete/SMARTS*
- *Asynchronous Transaction Processing under openUTM*

183 OVSIZE - Storage Thread Overflow Size

This Natural profile parameter specifies the maximum total amount of variable storage that may be allocated by one Natural session outside its storage thread.

Possible settings	`0-2097151`	Maximum total storage outside the thread in KB.
Default setting	`2097151`	That is, the storage outside the thread is limited by the region size only.
Dynamic specification	yes	
Specification within session	no	

If the storage within the thread is exhausted during a Natural session, additional storage can be allocated outside of the thread. `OVSIZE` can be used to limit the total amount of variable storage. This does not affect physical storage (see profile parameter `WPSIZE`), which is allocated outside the thread always.

For non-thread environments (e.g. in batch mode or under TSO), this parameter is not honored.

184 PARM - Alternative Parameter Module

This Natural profile parameter specifies an object module containing profile parameter definitions.

Possible settings	1 - 8 characters	Module name.
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. These definitions are coded using the various macros as described under *Building a Natural Parameter Module* in the *Operations* documentation. The macros are then assembled, resulting in an object module whose name is specified by the user.
2. When the `PARM` parameter is specified (either in the linked Natural parameter module or as a dynamic parameter at Natural session start), the appropriate object module is loaded and the profile parameter definitions contained therein take effect. The parameter module is loaded dynamically from the steplib.
3. Under CICS, a PPT entry is required for this parameter module.
4. Under BS2000, z/OS batch mode and TSO, the current steplib can be defined by profile parameter `LIBNAM`.
5. Any profile parameter definitions in effect before the `PARM` parameter is processed (for example, definitions contained in the linked parameter module or prior dynamic parameters), except the profile parameters `ISIZE` and `NUCNAME`, are overridden when the specified parameter module is loaded. Therefore, any dynamic parameters should be specified after the `PARM` specification.
6. The profile parameters `ISIZE` and `NUCNAME` are ignored if specified in an alternative Natural parameter module.
7. To restrict the use of an alternative parameter module, you can use the macro `NTUSER` (described in the `USER` profile parameter description).

185 PC - Control of Personal-Computer Access Method

This Natural profile parameter determines whether support of the personal-computer access method is to be provided using Natural Connection.

Possible settings	`ON`	Personal-computer support is enabled. The Natural statements `READ PC FILE` or `WRITE PC FILE` can be used (for uploading or downloading); see `UPLOAD PC FILE` and `DOWNLOAD PC FILE`. **Note:** With `PC=ON`, the system variable `*DEVICE` will always contain the value `PC`.
	`OFF`	No personal-computer support is to be provided.
	`NAM`	Field names are sent when data are uploaded/downloaded. **Note:** This value is for mainframe environments only.
	`NONAM`	No field names are sent when data are uploaded/downloaded. **Note:** This value is for mainframe environments only.
Default setting	`(OFF,NAM)`	
Dynamic specification	yes	
Specification within session	yes	The terminal commands `%+` and `%-` can be used to control the PC support.

Notes:

1. This Natural profile parameter only applies if Natural Connection is installed.
2. Multiple values are specified in a value list; see *Example*.
3. See the Natural Connection documentation for further information.
4. The files used for the PC access method have to be defined with the macros `NTPRINT` and `NTWORK` or the profile parameters `PRINT`, `WORK` and `HCAM`.

Example:

```
PC=(ON,NONAM)
```

186 PC - Periodic Group Count

This session parameter determines the number of periodic group occurrences to be output by default if a periodic group (or a field contained within a periodic group) is specified without an index in a `DISPLAY` or `WRITE` statement.

Possible settings	`0` or `1 - 191`	Number of values. **Note:** If `PC=0` is specified, then there is no default index range for the output of a PE field. Therefore, when a PE field is output, it is necessary to specify an explicit index or index range. Otherwise, a syntax error (NAT0281) will be raised
Default setting	`1`	
Specification within session	yes	
Applicable statements	`FORMAT`	

	`INPUT`
	`DISPLAY`
	`WRITE`
	`PRINT`
		Parameter may be specified at statement level and/or at element level.

Applicable command	none	

Note: This session parameter may be used in reporting mode only.

Example:

```
FORMAT PC=5
```

187 PCNTRL - Print-Control Characters

This Natural profile parameter specifies the line-advance characters for printing which are inserted in Column 0 of each print line.

Possible settings	Any character string	This parameter can be specified in character or hexadecimal format.
Default setting	`X'404142434445464748494A4B4C4D4E4F'` (in BS2000 environments)	
	`' 0-'` (z/OS, z/VSE environments) **Note:** This is the default setting according to ASA standard settings, which means that a blank causes a line advance of 1 line, `0` of 2 lines and `-` of 3 lines. **Caution:** Do not change the default setting of this parameter in an IBM environment.	
Dynamic specification	yes	
Specification within session	no	

188 PD - Limit of Pages for NATPAGE

This Natural profile and session parameter specifies the maximum number of pages (screens) which can be stored at the same time in the Natural system file (`FUSER`) with the NATPAGE screen-paging utility.

Within a Natural session, the profile parameter `PD` can be overridden by the session parameter `PD`.

Possible settings	`0` or `1 - 255`	Maximum number of pages (screens).
Default setting	`50`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	

Notes:

1. If the number of stored screens exceeds the setting of `PD`, wrap-around technique is used for the system file, which means that the oldest page is overwritten.
2. For further information on the NATPAGE screen page utility, see the terminal commands `%E`, `%I`, `%O`, %P and `%S`.

189 PDPSIZE - Size of the Profiler Data Pool

This Natural profile parameter determines the size of the Profiler data pool used by the NaturalONE Profiler and the Profiler utility (see the *Utilities* documentation) in batch mode. The Profiler trace session uses this data pool as shared intermediate storage to transfer traced RDC session records to the monitor session. Storage is allocated by the monitor session during the initialization of the NaturalONE Profiler or the Profiler utility.

Possible settings	`100 - 2097151`	Size of the Profiler data pool in KB. Depending on the number of trace records to be transferred, you may have to increase the size value for `PDPSIZE` to avoid a data pool overflow. An overflow pauses trace recording which can degrade performance of large applications. You have to restart the monitor session after increasing the size value.
	`0`	If `PDPSIZE=0` is set or if the specified size exceeds the storage space available for the data pool, you can neither use the NaturalONE Profiler nor run the Profiler utility in batch mode.
Default setting	`500`	
Dynamic specification	yes	
Specification within session	no	

190

PECK - PCHECK/ECHECK Error Processing

This Natural profile parameter controls whether a compilation check with the `ECHECK` or `PCHECK` option of the `COMPOPT` system command (see the *System Commands* documentation) terminates after a syntax error is detected in the object source. In addition, `PECK` determines how the syntax errors are reported.

Possible settings	`S`	Stops when the first syntax error is detected. The cursor is placed in the line that contains the error and the respective error (for example, NAT0935) is issued.
	`WS`	Same as `S` above, but additionally clears the message buffer when the compilation starts.
	`F`	Scans the entire object and places all errors on a stack. The cursor is placed in the line where the first error is detected. If several errors occur in the same line, a Natural error message appears in this line indicating that inconsistencies were found during the `PCHECK/ECHECK` validation. If several errors occur in different lines, the above Natural error message appears in the first line. All errors accumulated on the stack are listed after the scan is complete.
	`WF`	Same as `F` above, but additionally clears the message buffer when the compilation starts.
	`WL`	Same as `L` below, but additionally clears the message buffer when the compilation starts.
Default setting	`L`	Scans the entire object and places all errors on a stack. The cursor is placed in the line where the last error is detected. If several errors occur in the same line, a Natural error message appears in this line indicating that inconsistencies were found during the `PCHECK/ECHECK` validation. If several errors occur in different lines, the above Natural error message appears in the last line.

		All errors accumulated on the stack are listed after the scan is complete.
Dynamic specification	yes	
Specification within session	no	

191

PGP - Properties for External Programs

This Natural profile parameter can be used to predefine the properties for external programs. It corresponds to the `NTPGP` macro in the Natural parameter module.

Possible settings	See ***PGP Parameter Syntax***.	
Default setting	`OFF`	No properties are defined for external programs.
Dynamic specification	yes	The profile parameter `PGP` can only be specified dynamically. In the Natural parameter module, the macro `NTPGP` must be used.
Specification within session	yes	Temporary `SET CONTROL 'P=`*`value`*`'` for each call of the external program.

Notes:

1. When calling an external program with the Natural `CALL` statement, there are already several options available by means of a preceding `SET CONTROL` statement (for example, `SET CONTROL 'P=S'`) to request certain programming interface properties for the subsequent `CALL`. With `NTPGP` or `PGP` it is possible to predefine these program properties in the Natural parameter module or dynamically rather than in the Natural application program, which could be error prone
2. To provide `PGP` definitions for different programs, `PGP` or `NTPGP` must be specified multiple times.
3. The terminal command `%P=` applies only to the next call of an external program, and the call options are reset unconditionally on return from the call, whereas the profile parameter `PGP` sets the call option for an external program permanently.
4. If different properties are defined for the same program, by means of the profile parameter `PGP` or by a `SET CONTROL 'P=`*`value`*`'` statement, these properties will be merged for this program.

PGP Parameter Syntax

With the dynamic `PGP` parameter, you first specify the program name, and then one or more properties for this program.

```
PGP=(program-name,property-1,property-2,...)
```

Or:

PGP=OFF

Notes:

1. The value `OFF` resets all properties for the program previously defined.
2. To provide `PGP` definitions for different programs, the profile parameter `PGP` must be specified multiple times (separated by a comma or a blank).

Syntax Element	Explanation
`program-name`	The name of the external program which shall have the subsequent properties. Generic names are supported by an ending asterisk. Example: `'TESTP*'`. **Note:** The apostrophes must be used for dynamic specification.

`property-1`
`property-2`
`...`
	The various property values are described in the table below. There is a long and a short form for each property.
	Note: As dynamic parameter specification, a property can be reset by a preceding `NO`, for example, `NOS` to reset property `S`.

`OFF`	`PGP=OFF` resets all program property definitions.

Property Values

Property	Short	Explanation	SET CONTROL
`STDL`	`S`	Standard linkage under CICS.	`P=S`
`STDLC`	`SC`	Standard linkage simulates `EXEC CICS LINK`.	`P=SC`
`STDLQ`	`SQ`	Standard linkage on QR TCB under CICS. This property only applies if Natural CICS Interface is used. This property does not apply if an external subprogram is defined to Natural with the `CSTATIC` or `RCA` profile parameter.	`P=SQ`
`ROLL`	`V`	Rollout for call in thread environments.	`P=V`
`COMA`	`C`	Pass values in CICS COMMAREA.	`P=C`
`CONT`	`CC`	Pass parameter values in CICS container.	`P=CC`
`LEMAIN`	`L`	The called program is an IBM LE main program.	`P=L`
`LESUB`	`LS`	The called program is an IBM LE subprogram (IBM only).	`P=LS`
`IMSPCB`	`I`	Pass the IBM IMS TM PCB address.	`P=I`
`DIGR`	`D`	The called program is a DIGNUS remote subprogram.	`P=D`
`DIGL`	`DL`	The called program is a DIGNUS local subprogram.	`P=DL`
`SUOW`	`U`	SUOW, separate CICS unit of work.	`P=U`

Property	Short	Explanation	SET CONTROL
`SUOWB`	`UT`	Separate CICS unit of work tolerate backout.	`P=UT`
`OFF`		Clear all properties previously defined (dynamic parameter specification only).	

NTPGP Macro Syntax

With the `NTPGP` macro in the Natural parameter module, you first specify the name of the external program and then one or more properties for this program.

```
         NTPGP program-name,property-1,property-2,...
```

Notes:

1. The syntax elements of the macro `NTPGP` correspond to that of the profile parameter `PGP`, see *Syntax Elements* and *Property Values*.
2. The value `OFF` cannot be set with the macro `NTPGP`.
3. To provide property definitions for different programs, the macro `NTPGP` must be specified multiple times.

Examples of PGP Parameter

```
PGP=(TESTPGM1,S)
PGP=('ABX*',L,NOS)
PGP=(MYPROG7,OFF,L,STDL)
```

Examples of NTPGP Macros

```
         NTPGP TESTPGM1,S
         NTPGP ABX*,L
         NTPGP MYPROG,L,STDL
```

192 PLOG - Logging of Dynamic Parameters

This Natural profile parameter enables you to print a list of all profile parameters that were specified dynamically at the start of the session.

Possible settings	`ON`	**In batch mode:** At session start, a list of the dynamically specified profile parameters and their settings is written to the output data set `CMPLOG`. (If `CMPLOG` is not available, the list is written to the standard output data set `CMPRINT`.) **In online mode under TSO:** At session start, a list of the dynamically specified profile parameters and their settings is written to the output data set `CMPLOG`. (If `CMPLOG` is not available, the list is sent to the terminal.) **In online mode under CICS:** At session start, a list of the dynamically specified profile parameters and their settings is sent to the terminal.
	`OFF`	No list of dynamic profile parameters is written.
Default setting	`OFF`	
Dynamic specification	yes	**Note:** 1. When specified dynamically, the `PLOG` parameter applies to all subsequent dynamic profile parameters until the next `PLOG` specification. This allows you to exclude individual parameters from being printed, for example, if their settings contain passwords or other sensitive information that should not be printed. 2. All dynamic parameters which are specified between a `PLOG=OFF` specification and a subsequent `PLOG=ON` specification are not printed.
Specification within session	no	

Notes:

1. This Natural profile parameter only applies in batch mode, under TSO and under CICS.
2. Printing a list of all profile parameters that were specified dynamically at the start of the session may be useful to ascertain which dynamic profile parameters were actually used, particularly if profile parameters such as `PROFILE` or `SYS` are specified, which in turn “contain” other profile parameters (for a `PROFILE` or `SYS` parameter, the entire string of profile parameters activated by it is listed).

193 PM - Print Mode

The following topics are covered below:

Profile Parameter PM

The Natural profile parameter `PM` specifies how fields are to be printed or displayed.

Possible settings	`C, P, I, R,`
	or combinations thereof
	`CI, CR, PI, PR`
		`PM=C`	An alternative character set is to be used. It can be defined by the profile parameters `TAB1` and `TAB2`.

		`PM=P`	The primary (standard) character set is to be used.
		`PM=I`	Specifies inverse, that is, right-to-left direction (for example, for use in Middle Eastern countries). See also **Notes**.
		`PM=R`	This resets the `PM=I` setting to normal (left to right) display direction.
Default setting	`PR`		
Dynamic specification	yes		
Specification within session	yes	Terminal command `%V` or system command `GLOBALS`.	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.	

Notes:

1. `PM=I` affects any system controlled output screen items, that is, system variables and PF key lines. Moreover, all non-alphanumeric fields, for example, numeric and date are affected. In addition, for Natural Web I/O Interface terminals the field sequence is changed from left to right into right to left. The field inversion routine is supplied as assembler module `NATPM` in the Natural source library and can be modified in case of need.
2. For detailed information on how to use the setting `PM=I`, see *Bidirectional Language Support* in the *Unicode and Code Page Support* documentation.

Session Parameter PM

This session parameter `PM` is used to indicate how fields are to be displayed.

Possible settings	`PM=C`	An alternative character set is used (see the module `NATPM` in the Natural source library).
	`PM=D`	Defines DBCS-only fields that do not contain shift-out/shift-in characters (see *Double-Byte Character Sets* in the *Operations* documentation).
	`PM=I`	Field values are displayed in inverse direction; that is, from right to left (for example, for use in Middle East countries).
	`PM=N`	No hardcopy of the display can be made.
Default setting	none	The standard character set is used.

Applicable statements	`DEFINE DATA`
	`DISPLAY`
	`FORMAT`
	`INPUT`
	`MOVE LEFT/RIGHT JUSTIFIED`
	`PRINT`
	`WRITE`	

Note: More than one value may be specified.

Example:

```
LIMIT 1
   READ EMPLOYEES
   DISPLAY NOTITLE NAME
   DISPLAY NOTITLE NAME (PM=I)
   DISPLAY NOTITLE NAME
   END
```

Result:

```
NAME
   --------------------
   MORENO
                 ONEROM
   MORENO
```

194 POS22 - Version 2.2 Algorithm for POS System Function

This Natural profile parameter is obsolete and only accepted for compatibility reasons.

195 PRINT - Print File Assignments

This Natural profile parameter specifies the print files to be used during the session. It corresponds to the `NTPRINT` macro in the Natural parameter module.

Possible settings	See ***PRINT Parameter Syntax***.	
Default setting	See the default values of the different keyword subparameters described below. Depending on the access method and the environment, there may be different default settings.	
Dynamic specification	yes	The parameter `PRINT` can only be specified dynamically. In the Natural parameter module, use the macro `NTPRINT`.
Specification within session	no	

Notes:

1. The old dynamic parameter `PRINTER` can be used as a synonym for `PRINT`.
2. Within a session, up to 31 logical print files (numbered 1 to 31) and the hardcopy print file (Number 0) can be used.
3. The software components for accessing print files in different environments are called access methods. For the duration of a Natural session, each logical print file can be assigned to one access method only. The access method for a print file is determined by the keyword subparameter `AM`.
4. In z/OS under TSO and in batch mode, print files need not be predefined in the JCL. Provided they are defined by subparameter `AM=STD`, they can be allocated dynamically during the session in a Natural program using the `DEFINE PRINTER` statement or the application programming interface `USR2021` (in library `SYSEXT`).
5. See also *Print and Work File Handling with External Data Sets in a Server Environment* in the *Operations* documentation.

PRINT Parameter Syntax

With the `PRINT` parameter, you first specify one or more logical print file numbers, and then several keyword subparameters, which define the characteristics for these print files:

```
PRINT=((print-file-numbers),keyword-subparameter=value,...)
```

Where:

Syntax Element	Description
print-file-numbers	The file numbers must be specified first and enclosed in parentheses: ■ The numbers can be from 0 to 31. ■ They can be specified in any sequence. ■ Multiple numbers must be separated from one another by commas or blanks. ■ To specify a range of numbers, you can use a hyphen (-).
keyword-subparameters	The keyword subparameters (for the different environments) are described below. If any previous definition (or default) for the same print file exists, only the values for the specified keyword subparameters are overwritten, all other values remain unchanged.

Note: To provide different print file definitions, `PRINT` can be specified multiple times.

Examples:

```
PRINT=((2,12,18),AM=STD,DEST='PRINT**',OPEN=INITOBJ,CLOSE=CMD)
PRINT=((1,3,6-11,15),AM=NAF)
PRINT=((0),AM=STD,DEST=HARDCOPX)
```

NTPRINT Macro Syntax

With an `NTPRINT` macro, you first specify one or more logical print file numbers, and then several keyword subparameters which define the characteristics that are to apply to these print files:

```
          NTPRINT (print-file-numbers),keyword-subparameter=value,...
```

Where:

Syntax Element	Description
print-file-numbers	The file numbers must be specified first and enclosed in parentheses: ■ The numbers can be from 0 to 31. ■ They can be specified in any sequence. ■ Multiple numbers must be separated from one another by commas or blanks. ■ To specify a range of numbers, you can use a hyphen (-).

`keyword-subparameters`	The keyword subparameters (for the different environments) are described below. If any previous definition (or default) for the same print file exists, only the values for the specified keyword subparameters are overwritten, all other values remain unchanged.

Note: To provide different print file definitions, `NTPRINT` can be specified multiple times.

Examples:

```
        NTPRINT (2,12,18),AM=STD,DEST='PRINT**',OPEN=INITOBJ,CLOSE=CMD
        NTPRINT (1,3,6-11,15),AM=NAF
        NTPRINT (0),AM=STD,DEST=HARDCOPX
```

Keyword Subparameters for All Environments

The following keyword subparameters are available for all environments:

`AM` | `CCHAR` | `DEST` | `OPEN` | `CLOSE` | `ROUTE` | `CP` | `SHIFT`

AM - Type of Access Method

`AM=`*`value`* specifies the type of access method to be used.

Value	Access Method
`STD`	Standard sequential batch files (batch, TSO, TIAM).
`COMP`	Com-plete print files.
`CICS`	CICS transient data or temporary storage.
`NAF`	Natural Advanced Facilities.
`IMS`	IMS TM destinations.
`PC`	Entire Connection.
`USER`	Third-party vendor print interface.
`SMARTS`	SMARTS print file.
`ESS`	Entire System Server.
`NOM`	Entire Output Management. **Note:** Prints to an Entire Output Management container file without using the spool of the operating system. Refer to the *Entire Output Management* documentation for details.

Value	Access Method
`OFF`	Unassigned. No automatic assignments if `FAMSTD=OFF` is set. **Note:** `PRINT=OFF` is equivalent to: `PRINT=((1-31)), AM=OFF)`. It does not affect any of the other keyword subparameter specifications.
`0`	Unassigned. Automatic assignments if `FAMSTD=OFF` is set. This is the default value.

Notes:

1. For an online session, all print files to be used have to be assigned to a specific access method.
2. For a batch session, any print files not assigned to a specific access method will be automatically detected and assigned by the standard batch access method (`AM=STD`), provided that they have been predefined in the JCL. See also ***FAMSTD - Overwriting of Print and Work File Access Method Assignments***.
3. `PRINT=((0),AM=`*`xxx`*`)` or `NTPRINT (0),AM=`*`xxx`* determines the hardcopy print access method and is equivalent to the profile parameter `HCAM=`*`xxx`*.

CCHAR - Allow Output Control Characters

`CCHAR=`*`value`* allows you to define hexadecimal control characters for print file I/O to be passed through unchanged.

Value	Explanation
`a1` or `(`*`a1`*`,`*`a2`*`,...)`	A single hex character or a list of hex characters enclosed in brackets can be specified. The hex characters must be within `x'01'` through `x'3F'`. A hex character range *`a1`*`-`*`a2`* is allowed instead of a hex character. There is no default value.
`OFF`	`CCHAR=OFF` resets any previous `CCHAR` definitions.

Notes:

1. To avoid screen I/O errors, Natural automatically translates the output control characters `x'01'` through `x'3F'` to `'?'`. In some cases, however, certain control characters are required for special purposes. These can be specified with `CCHAR`.
2. For the specified print files, the `CCHAR` specification replaces the definitions in the output translation tables `NTTAB`, `NTTAB1` and `NTTABL` as contained in the configuration module `NATCONFG` or defined by the corresponding dynamic profile parameter or by the corresponding macro in the Natural parameter module.

Examples:

```
CCHAR=17
```

```
CCHAR=19-1B
```

```
CCHAR=(03-06,0A,1B,3A-3F)
```

DEST - External Data Set Name

`DEST=value` specifies the print destination (1 - 8 characters).

Access Method	Meaning of Keyword Subparameter DEST
`AM=STD`	`DEST` is the logical data set name (`DDNAME`, `LINK` name, `DTF` name). If the destination is to be for multiple files, two asterisks (**) have to be specified for the file number. These will be replaced by the corresponding logical file number for each print file. A `DEST` value including two asterisks must be enclosed in apostrophes when it is used as a dynamic parameter. The default value is `DEST='CMPRT**'` for z/OS, z/VSE, and `DEST='P**'` for BS2000 environments. Under z/VSE, only 7-character names are supported.
`AM=CICS`	There is no default value for print files under CICS. Here, the `DEST` subparameter is mandatory, that is, CICS print files defined without a valid `DEST` specification are ignored. The Natural CICS interface also supports a variable (see the `TERMVAR=&TID` default parameter setting in the `NTCICSP` generation macro) as part of the `DEST` value which, when being specified, is replaced by the actual CICS terminal ID. See also *Natural Print and Work Files under CICS* in the *TP Monitor Interfaces* documentation.
`AM=IMS`	Specifies the IMS TM destination.

Notes:

1. `DEST=value` corresponds to the `OUTPUT` value of the `DEFINE PRINTER` statement (and can be overwritten by a `DEFINE PRINTER OUTPUT` specification).
2. The meaning of this keyword subparameter depends on the access method.
3. `PRINT=((0),DEST=xxx)` or `NTPRINT (0),DEST=xxx` determines the hardcopy print destination and is equivalent to the Natural profile parameter `HCDEST=xxx`.

OPEN - Time of File Opening

`OPEN=`*`value`* specifies when the file is to be opened.

Value	Explanation: The file is opened ...
`INIT`	for output at session initialization.
`OBF`	according to the default `OPEN` value for the different environments (batch, CICS, Com-plete, TSO).
`OBJ`	when the execution of the first object which accesses the file starts. This is the general default, except for `AM=COMP` and `AM=IMS`.
`OBJ1`	when the execution of the first object on Level 1 that accesses the file starts. Otherwise, it is opened when it is first accessed.
`ACC`	when it is first accessed by a statement. This is the default for `AM=COMP` and `AM=IMS`.
`INITOBF`	for output at session initialization. Any subsequent re-opening of the file sets the default `OPEN` value for the different environments (batch, CICS, Com-plete, TSO).
`INITOBJ`	for output at session initialization. Any subsequent re-opening of the file will be performed when the execution of the first object which accesses the file starts.
`INITOBJ1`	when the execution of the first object on Level 1 that accesses the file starts. Otherwise, it is opened when it is first accessed.
`INITACC`	for output at session initialization. Any subsequent re-opening of the file will be performed when it is first accessed by a statement.

CLOSE - Time of File Closure

`CLOSE=`*`value`* specifies when the file is to be closed.

Value	Explanation: The file is closed ...
`OBJ`	either when processing of the object in which it was first accessed is finished or when command mode, `NEXT` mode or `MAINMENU` is reached.
`CMD`	when command mode, `NEXT` mode or `MAINMENU` is reached. This is the default for `AM=NAF`, `AM=COMP` and `AM=IMS`.
`FIN`	at session end (this is the default for `AM=STD`). **Note:** With `CLOSE=FIN`, a `DEFINE PRINTER` statement causes an error if the printer was opened already. A `CLOSE PRINTER` statement for the printer is ignored.
`USER`	only if the file is open and one of the following conditions is true: ■ a `CLOSE PRINTER` statement is issued, ■ a `DEFINE PRINTER` statement is issued,

Value	Explanation: The file is closed ...
	■ at session termination.

ROUTE - Logical Print File Routing

`ROUTE=`*`value`* specifies whether logical print file routing is done according to the `OUTPUT` clause of the `DEFINE PRINTER` statement.

Value	Explanation
`ON`	Print file routing is done. The target print file can be any available print file except `PC`. This is the default value.
`OFF`	No print file routing is done.
`am`	Print file routing is done to printers of the specified access method only. Possible value is any valid print file access method (see subparameter `AM` described above). `PC` is not allowed for *`am`*.

Note: Print file routing means that, if the name defined in the `OUTPUT` clause of a `DEFINE PRINTER` statement denotes a print file destination which is defined by a different logical printer, all print output is routed to this print file. If no printer with the specified name is found, the print output can be routed to any free printer.

CP - Code Page for Print Output

`CP=`*`value`* specifies the code page for the print output.

Value	Explanation
1 - 64 characters.	The name of the desired code page. Any character string is possible, but must be predefined by one of the code page parameters `CCSID`, `CCSN`, `IANA` or `ALIAS` of the macro `NTCPAGE` in the source module `NATCONFG`.

Notes:

1. It is assumed that all code page data, for example, Natural sources, contents of A-format fields, etc., are stored in this code page. If no code page is specified with the keyword subparameter `CP`, the code page resulting from the evaluation of the profile parameter `CP` is used.
2. If Natural code page support is disabled (for example, by profile parameter `CP=OFF`), any value specified for this parameter is ignored.

3. See also profile parameter `CP` and *Profile Parameters and Macros* in the *`Unicode and Code Page Support`* documentation.

SHIFT – Shift Data Record to the Right

`SHIFT=`*`value`* specifies whether the print records are shifted to the right by the number of blanks *`nnn`*.

Value	Explanation
`0` or `1 - 248`	The print records are shifted to the right and the specified number of blanks is inserted in front of the record starting with column 1.
`0`	The print records are not shifted. This is the default value.

Notes:

1. The ASA control character in column 0 is not affected.
2. The specified number of characters at the end of the records is lost due to shift out.

Keyword Subparameters for AM=STD in All Environments

The following keyword subparameters are available for access method `AM=STD` in all environments:

`RECFM` | `BLKSIZE` | `LRECL` | `TRUNC` | `PAD` | `PADCHRO` | `ASA` | `STRIP`

RECFM - Default Record Format of Data Set

`RECFM=`*`value`* specifies the default record format of the data set.

Supported Formats

Value	Format
`F`	Fixed
`V`	Variable
`U`	Undefined
`B`	Blocked
`S`	Spanned
`A`	ASA
`M`	Machine control characters

Possible Values or Combinations of Values

Value	Explanation
`RECFM=F`, `RECFM=FA`, `RECFM=FM`, `RECFM=FB`, `RECFM=FBA`, `RECFM=FBM`, `RECFM=V`, `RECFM=VA`, `RECFM=VM`, `RECFM=VB`, `RECFM=VBA`, `RECFM=VBM`, `RECFM=VBS`, `RECFM=VBSA`, `RECFM=VBSM`, `RECFM=U`, `RECFM=UA`, `RECFM=UM`	These values or combinations of values can be specified.
`RECFM=VBA`	Variable blocked with ASA. This is the default value.

Note: The `RECFM` specification only applies if no record format is predefined in the JCL or (z/OS only) in the data set DCB.

BLKSIZE - Default Block Size of Data Set

`BLKSIZE=`*`value`* specifies the default block size of the data set.

Value	Explanation
`0` or `8` - `32767`	Default block size of the data set (in bytes).
`1016`	This is the default value.

Note: The `BLKSIZE` specification only applies if no block size is predefined in the JCL or (z/OS only) in the data set DCB.

LRECL - Default Record Length of Data Set

`LRECL=`*`value`* specifies the default record length of the data set.

Value	Explanation
`0` or `5 - 254`	Record length of the data set (in bytes).
`0`	This is the default value.

Notes:

1. This subparameter can be used to check for truncation and padding.
2. For `RECFM=V(B)` the `LRECL` value includes a 4-byte record descriptor word.
3. If `LRECL=0` is defined, the following applies:
   With `RECFM=V(B)`, `LRECL` defaults to the minimum of `BLKSIZE-4` and `254`.
   With `RECFM=U`, `LRECL` defaults to `BLKSIZE`.
   With `RECFM=F(B)`, the maximum record length in the Natural program being executed is taken when the file is opened. If no record length from a program is available when the file is opened,

for example with `OPEN=INIT`, a record length of 132 is taken (plus 1 for ASA or a machine control character and/or plus 4 for a record-descriptor word if the record format is variable).

4. The `LRECL` specification only applies if no record length is predefined in the JCL or (z/OS only) in the data set DCB.

TRUNC - Truncation of Output Records

`TRUNC=`*`value`* specifies whether the output records are truncated.

Value	Explanation
`ON`	Output records that are longer than the record length (`LRECL`) of the data set will be truncated. This is the default value.
`OFF`	Error NAT1512 will be issued if an output record is longer than the data set record length.

PAD - Padding of Output Records

`PAD=`*`value`* specifies whether the output records are padded or not (applies only to data sets of fixed record length).

Value	Explanation
`ON`	Output records that are shorter than the record length (`LRECL`) of the data set will be padded with padding characters defined by keyword subparameter `PADCHRO`. This is the default value.
`OFF`	Error NAT1510 will be issued if an output record is shorter than the data set record length.

PADCHRO - Padding Character of Output Records

`PADCHRO=`*`value`* specifies the character which is used for padding if `PAD=ON` is defined for the print file.

Value	Explanation
`'x'`	One character *x* within single quotes
`x'xx'`	One hex character *xx*
`' '` (blank)	Blank or `x'40'` This is the default value.

ASA - Use of ASA Record Format

`ASA=value` specifies whether the ASA record format is used.

Value	Explanation
`ON`	An ASA character is included in the output print records. Under z/OS, this enforces ASA record format, regardless of the `RECFM` setting in the `DCB` or the `RECFM` subparameter. This is the default value.
`OFF`	No ASA character is included in the output print records. Under z/VSE batch access method (`AM=STD`), a valid ASA character must be supplied in column one of the output record if the output file is a spool file, otherwise error NAT1530 will be issued.

STRIP - Inhibit Removal of Trailing Blanks

`STRIP=value` can be used to inhibit the removal of trailing blanks.

Value	Explanation
`ON`	Trailing blanks are stripped off. This is the default value.
`OFF`	Trailing blanks are not stripped off.

Note: Trailing blanks are stripped off for batch sequential print files (access method `AM=STD`) if the data set is defined with variable record format (`RECFM=VB`) to reduce disk space. This may cause problems with subsequent applications accessing this data set, due to the missing blanks. These problems can be avoided by setting `STRIP=OFF`.

Keyword Subparameters for AM=STD in z/OS Environments

The following keyword subparameters are available for access method `AM=STD` in z/OS environments:

`REREAD` | `FREE` | `BUFNO` | `DISP` | `VMAX`

REREAD - Closing of Tape File Data Sets

`REREAD=value` specifies the `REREAD` option for the closing of the tape file.

Value	Explanation
`ON`	The `REREAD` option is set for the `CLOSE SVC`. This causes the volume to be repositioned to reprocess the data set. This is the default value.
`OFF`	The `REREAD` option is not set for the `CLOSE SVC`.

FREE - Data Set De-allocation at File Closure

`FREE=value` specifies whether the data set is de-allocated when the file is closed.

Value	Explanation
`ON`	The `FREE` option is set for the `CLOSE SVC`, which means that the data set is de-allocated when it is closed (and not at step termination).
`OFF`	The `FREE` option is *not* set for the `CLOSE SVC`. This is the default value.

BUFNO - Default Number of z/OS I/O Buffers of Data Set

`BUFNO=value` specifies the default number of z/OS I/O buffers of the data set.

Value	Explanation
`0` or `1 - 255`	Default number of z/OS I/O buffers of the data set.
`0`	In this case, z/OS allocates five I/O buffers. This is the default value.

Notes:

1. The number of I/O buffers can improve the performance of print file access dramatically. Note that the storage for I/O buffers is allocated below the 16 MB line.
2. The `BUFNO` specification applies only if the `BUFNO` parameter is not specified in the JCL for the data set.

DISP - Open Print File for Modification

`DISP=value` specifies whether the print file is opened for modification.

Value	Explanation
`MOD`	New records are added at the end of the file.
`NOMOD`	The print file is rewritten from the start. This is the default value.

Note: This subparameter corresponds to the JCL DD statement parameter `DISP=MOD`.

VMAX - Control LRECL for Variable Record Format

`VMAX=value` controls the `LRECL` setting for an output file with variable record format (`RECFM=V`).

Value	Explanation
`ON`	Providing a non-zero `BLKSIZE` value exists for the file, `VMAX=ON` sets `LRECL=BLKSIZE-4` for variable record format, regardless of the `LRECL` setting in the `DCB` or the `LRECL` subparameter.
`NAT`	`LRECL` is set to the length +4 of the largest record in the application program if this value is less than `LRECL` in the `DCB` for the data set.
`OFF`	`LRECL` from the `DCB` for the data set or the `LRECL` subparameter is used. This is the default value.

Keyword Subparameters for AM=STD in z/VSE Environments

The following keyword subparameters are available for access method `AM=STD` in z/VSE environments:

`SYSNR` | `LABEL` | `REWIND`

SYSNR - Logical VSE SYS Number

`SYSNR=value` specifies the logical VSE SYS number.

Value	Explanation
`1 - 99`	Logical VSE SYS number. Default value: - For print files 1 - 31, the SYS number is print file number plus 40. - For print file 0, that is the hardcopy printer, the default is `SYSLST`.

Example:

The z/VSE default SYS number for print file 11 is `11 + 40 >= SYS051`:

```
SYSNR=51
```

LABEL - Tape Label Processing

`LABEL=`*`value`* specifies the tape label processing.

Value	Explanation
`ON`	The tape is in standard label format. This is the default value.
`OFF`	The tape is unlabeled with front tape mark.
`NOTM`	The tape is unlabeled without front tape mark.

REWIND - Action at File Closure

`REWIND=`*`value`* specifies the action to be taken when a tape file is closed.

Value	Explanation
`ON`	The tape is rewound when the file is closed. This is the default value.
`OFF`	The tape is not rewound when the file is closed.
`UNLOAD`	The tape is unloaded when the file is closed.

Keyword Subparameters for AM=STD in BS2000 Environments

The following keyword subparameters are available for access method `AM=STD` in BS2000 environments:

`DISP` | `FREE`

DISP - File Open Mode

`DISP=value` specifies the open mode of the file.

Value	Explanation
`EXT`	The open mode is set to `EXTEND`.
`NOEXT`	The open mode is set to the default value `OUTPUT`. This is the default value.

FREE - Release Linkname at File Closure

`FREE=value` specifies whether the linkname of the file is released when the destination file is switched over to another one.

Value	Explanation
`ON`	The linkname is released.
`OFF`	The linkname is kept.

Example:

```
DEFINE PRINTER (1) OUTPUT 'P01'
WRITE (1) 'TEST'
CLOSE (1)
DEFINE PRINTER (1) OUTPUT 'FILE=REPORT01.NEW,LINK=LINKP01
```

If `FREE=ON` is set, the linkname is released; with `FREE=OFF`, it is kept.

Keyword Subparameters for AM=CICS

The following keyword subparameters are available for access method `AM=CICS`:

`TYPE` | `DISP`

TYPE - Type of CICS Storage Medium

`TYPE=`*`value`* specifies the type of CICS storage medium to be used.

Value	Explanation
`MAIN`	Temporary main storage.
`AUX`	Temporary auxiliary storage.
`TD`	Transient data.

Note: The default value used depends on the setting of subparameter `DEST`. If the `DEST` subparameter value matches a valid CICS transient data queue, the `TYPE` subparameter defaults to `TD`, otherwise `MAIN` will be taken as the default value.

DISP - CICS Temporary Storage Queue Disposition

`DISP=(`*`value1`*`,`*`value2`*`)` specifies the CICS temporary storage queue disposition.

Value Pair	Explanation
`(NEW,KEEP)`	The storage queue is deleted when the file is opened. This is the default value.
`(NEW,DELETE)`	The storage queue is deleted when the file is opened and when it is closed.
`(OLD,DELETE)`	The storage queue is deleted when the file is closed.
`(OLD,KEEP)`	The storage queue is not deleted.

Example:

```
DISP=(NEW,DELETE)
```

Note: The `DISP` specification does not apply to CICS extra-partition transient data queues.

Keyword Subparameters for AM=COMP (Com-plete)

The following keyword subparameter is available for access method `AM=COMP` (Com-plete):

`DRIVER`

DRIVER - Name of Com-plete Print Driver

`DRIVER=`*`value`* specifies the name of the Com-plete print driver to be used.

Keyword Subparameters for AM=SMARTS (Com-plete)

The following keyword subparameter is available for access method `AM=SMARTS` (Com-plete):

`DEST`

DEST - Logical Printer

`DEST=`*`value`* specifies the logical printer.

Value	Explanation
`print-server-queue`	The environment variable `SAG_APS_LPD_`*`xyz`* defines a logical printer under Com-plete, where *`xyz`* is the name of the print server queue. If the environment variable `SAG_APS_LPD_`*`xyz`* exists for the specified `DEST`, the output is directly routed to that line printer. For more information, see the *Complete Initialization and Startup Manual*, section *Defining Terminals and Printers*.
`printer-file-name`	If no print server queue for that printer is available, `DEST` specifies a printer file name. It specifies the location of the output file in the file system. The name of the output file is generated from the `UserId` and a sequence number. Since the `DEST` clause is restricted to an 8 character maximum, it is useless to define a file with absolute PFS path specification. The name specified in the `DEST` clause is relative to the print file root directory. The print file root directory is specified with the environment variable `NAT_PRINT_ROOT`.

Example:

```
NAT_PRINT_ROOT=/nat/printer
DEST=printer1
UserId=xyz
```

The first output will be written to file `/nat/printer/printer1/xyz1`.

To specify a file with absolute path definition, the `OUTPUT` clause of the `DEFINE PRINTER` statement must be used.

Keyword Subparameters for AM=IMS

The following keyword subparameters are available for `AM=IMS`:

`BLKSIZE` | `DRIVER`

For possible values and further information, see *Support of the Natural WRITE (n) Statement* in the *Natural under IMS TM* part of the *TP Monitor Interfaces* documentation.

BLKSIZE - Size of the Print Buffer

`BLKSIZE=`*`value`* specifies the size of the print buffer sent to the IMS TM destination.

DRIVER - Name of Natural IMS Print Driver

`DRIVER=`*`value`* specifies the name of the Natural IMS print driver to be used.

Keyword Subparameters for DEFINE PRINTER Statement

With the following keyword subparameters, you can set the default values for the `DEFINE PRINTER` statement options of the same names. When a printer is closed, all `DEFINE PRINTER` statement options are reset to their default values.

`PROFILE` | `NAME` | `FORMS` | `DISP` | `COPIES` | `CLASS` | `PRTY`

PROFILE - Name of Printer Control Characters Table

`PROFILE=value` specifies the name of the printer control characters table (`NTCCTAB` macro).

NAME - Name of Listing

`NAME=value` specifies the listing name.

FORMS - Name of Listing Forms

`FORMS=value` specifies the listing forms name.

DISP - Listing Disposition

`DISP=value` specifies the listing disposition (`HOLD`, `KEEP`, `DELETE` or `LEAVE`).

COPIES - Number of Copies

`COPIES=value` specifies the number of copies to be printed (`1 - 255`).

CLASS - Spool Class

`CLASS=value` specifies the spool class (1 byte).

PRTY - Listing Priority

`PRTY=value` specifies the listing priority (`1 - 255`).

196 PROFILE - Apply Parameter Profile

This Natural profile parameter can be used to apply a parameter profile.

Instead of having to specify a whole string of individual parameters each time you invoke Natural, you can specify the string of parameters once, store this string under a profile name and then invoke Natural with this parameter profile. The parameters defined with this profile are then passed to Natural as dynamic profile parameters.

You create and maintain these profiles with the SYSPARM utility (described in the *Utilities* documentation).

You can use the profile parameter `FPROF` to specify a system file for parameter profiles (`FPROF`).

Possible settings	See ***PROFILE Parameter Syntax***.	
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If the `PROFILE` parameter is specified in the Natural parameter module, it is evaluated *after* the other parameters in the parameter module, but *before* any dynamically specified profile parameters are evaluated; this means that parameters specified within the profile can be overridden by individually specified dynamic parameters.
2. To restrict the use of a profile, you can use the profile parameter `USER`.
3. Unlike other parameters, a `PROFILE` parameter specification cannot be overwritten by another `PROFILE`. So you can have multiple parameter profiles which are evaluated all in a sequence.
4. The `PROFILE` parameter cannot be used with `ADARUN MODE=SINGLE`.

This chapter covers the following topics:

PROFILE Parameter Syntax

The `PROFILE` parameter is specified using either of the following syntax options:

Syntax 1 - Default Database/File:

$$\text{PROFILE=}\left\{\begin{array}{l}\textit{profile-name}\\ \textbf{AUTO}\\ \textbf{PROGRAM}\\ \textbf{TERMINAL}\end{array}\right\}$$

Syntax 2 - Specified Database/File:

$$\text{PROFILE=(}\left\{\begin{array}{l}\textit{profile-name}\\ \textbf{AUTO}\\ \textbf{PROGRAM}\\ \textbf{TERMINAL}\end{array}\right\}\textit{,dbid,fnr,password,cipher-key}\text{)}$$

The elements of the syntax diagrams are described in the following section.

For explanations of the symbols used in the syntax diagrams, see Syntax Symbols (*Statements* documentation).

Storage Location of Parameter Profiles

Storage of parameter profiles depends on whether the database ID and file number are specified with the `PROFILE` parameter. The following applies:

- If the database ID and file number are not specified with `PROFILE` (**Syntax 1**), default settings are used:
  1. If the `FPROF` profile parameter is set, the parameter profile is read from the current `FPROF` system file.
  2. If the `FPROF` profile parameter is not set, the parameter profile is read from the current `FNAT` system file.
- If the database ID and file number are specified with `PROFILE` (**Syntax 2**), the specified values are used.

Syntax Element Description

Syntax Element	Value	Explanation
profile-name	1 - 8 characters	The name of the profile to be used. If you want that all profiles used during a session are read from a database and system file other than the default, specify the following before you use the first profile:

Syntax Element	Value	Explanation
		`PROFILE=(,dbid,fnr)`
`AUTO`	n/a	Natural takes the current TP user ID (as contained in the system variable `*INIT-USER`) as profile name, which means that the profile defined under the name corresponding to that ID is used. If no such profile is found, a profile named `AUTO` is used instead (if available). You can define such an `AUTO` profile as default profile for users without individual profiles.
`PROGRAM`	n/a	Natural takes the name of the program currently executing as Natural (as contained in the system variable `*INIT-PROGRAM`) as profile name, which means that the profile defined under this name is used. If no such profile is found, a profile named `PROGRAM` is used instead (if available). You can define such a `PROGRAM` profile as default profile for users without individual profiles.
`TERMINAL`	n/a	Natural takes the current terminal ID (as contained in the system variable `*INIT-ID`) as profile name, which means that the profile defined under the name corresponding to that ID is used. If no such profile is found, a profile named `TERMINAL` is used instead (if available). You can define such a `TERMINAL` profile as default profile for users without individual profiles.
dbid	`1 - 65535`, except `255`	The ID of the database (DBID) in which the Natural system file for parameter profiles is located. If *dbid* is not specified, the corresponding value of the profile parameter `DBID` is used.
fnr	`1 - 65535`	The file number (FNR) of the database in which the Natural system file for parameter profiles is located. If *fnr* is not specified, the corresponding value of the profile parameter `FNR` is used.
password	1 - 8 characters	The Adabas password if required for a password-protected Natural system file. If *password* is not specified, the corresponding value of the profile parameter `SYSPSW` is used.
cipher-key	8 characters	The Adabas cipher key if required for a password-protected Natural system file. If *cipher-key* is not specified, the corresponding value of the profile parameter `SYSCIP` is used.

197 PROGRAM - Non-Natural Program Receiving Control after Termination

This Natural profile parameter specifies a non-Natural back-end program which is to receive control after the termination of the Natural session.

Possible settings	1 - 8 characters	Non-Natural back-end program.
	numeric value	Setting a numeric value, for example `PROGRAM=0`, indicates “no back-end processing”. **Note:** This is particularly relevant when Natural is invoked by a front-end program, because a default may be taken if `PROGRAM` is blank or not specified; see *Front-End Invoked via XCTL* in the *TP Monitor Interfaces* documentation.
	`MSG`	This additional option determines that the Natural session termination message is issued before Natural passes control to the back-end program. The following syntax applies: `PROGRAM=(program-name,MSG)` **Note:** `MSG` is not evaluated under IMS TM and Com-plete.

	`NOMSG`	This additional option determines that the Natural session termination message is *not* issued before Natural passes control to the back-end program. The following syntax applies: `PROGRAM=(program-name,NOMSG)`
Default setting	`PROGRAM=(,NOMSG)`	
Dynamic specification	yes	
Specification within session	yes	The Natural back-end program can also be specified from within a Natural program by calling the Natural subprogram `CMPGMSET`, which is provided in the library SYSEXTP. The additional options `MSG` and `NOMSG` cannot be specified from within a Natural program.
Application programming interface	`USR4001N` (for mainframes)	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.
	`USR6204N` (for all platforms)	

Notes:

1. Data for the program specified with the `PROGRAM` parameter can be supplied with the `TERMINATE` statement.
2. For the conventions of calling non-Natural back-end programs, see *Back-End Program Calling Conventions* in the *Operations* documentation.

CICS-Specific Information:

In addition to back-end programs, the Natural CICS interface also supports back-end transactions which may be specified via `RET=XXXX` or `RTI=XXXX` or `STR=XXXX` instead of a program name, with `XXXX` being a valid CICS transaction ID.

- `RET=XXXX` indicates that control has to be passed to CICS together with a return transaction ID by a CICS `RETURN TRANSID ('XXXX')` command.
- `RTI=XXXX` indicates that control has to be passed to CICS with a return transaction ID by a CICS `RETURN TRANSID ('XXXX') IMMEDIATE` command.
- `STR=XXXX` indicates that a new transaction has to be started by a CICS `START TRANSID ('XXXX') TERMID (*INIT-ID)`, before relinquishing control via a CICS `RETURN` command.

Notes:

1. Back-end transactions are not supported if you start a Natural session with an `EXEC CICS LINK` command or a distributed program link (DPL). They are ignored if specified.

2. Return transactions (`RET=` or `RTI=`) are only supported for terminal-oriented sessions. They are ignored if specified for asynchronous sessions.

Examples:

```
PROGRAM=MYPGM
PROGRAM=(MYPGM,MSG)
PROGRAM=(,MSG)
```

198 PS - Page Size for Natural Reports

This Natural profile and session parameter specifies the maximum number of lines per page to be used for Natural reports created with the `DISPLAY` or `WRITE` statement.

Possible settings	`1 - 250`	Maximum number of lines per page.
	`0`	The physical page size is to be used. **Note:** 1. If `PS=0` is specified for the first report to be output (Report 0), the physical-device page-size minus 1 will be used. 2. If `PS=0` is specified for Reports 1 - 31, this will cause automatic new-page processing to be inhibited, that is, no automatic page-break processing will be performed.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`INPUT`
	`SET GLOBALS`
	`WRITE`	

Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. When used as a profile parameter, the `PS` parameter is honored in batch mode only and defines the physical page size.
2. In online mode, the physical page size is always set to the physical screen height.
3. See also *Page Size - PS Parameter* in the *Programming Guide*.
4. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.

199 PSEUDO - CICS Pseudo-Conversational Mode

This Natural profile parameter controls the mode of operation under CICS.

Note: When Natural is executing under control of the TP monitor CICS, two modes are possible: conversational and pseudo-conversational.

Possible settings	`ON`	`PSEUDO=ON` enables pseudo-conversational mode. **Note:** In this mode, a Natural session is a sequence of different transactions. After each output to the terminal, all Natural work areas are saved and the transaction is terminated. When the user responds to a message by pressing ENTER (or any other input key), a new transaction is initiated. The Natural work areas are restored, the terminal input is read and the Natural session is continued. The transaction identification of each following transaction can be set dynamically by calling the subroutine `CMTRNSET`, which is provided in the library `SYSEXTP`.
	`OFF`	`PSEUDO=OFF` disables pseudo-conversational mode and enables conversational mode. **Note:** 1. In conversational mode, a Natural session is one transaction which is active for as long as the Natural session is active. 2. Note for CICS: A specification of `PSEUDO=OFF` is ignored for Natural server sessions. See *Natural Server Sessions under CICS* in the *Operations* documentation.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Note: For more information, refer to Natural under CICS, section *TYPE - Thread Type for Group* in the *TP Monitor Interfaces* documentation.

200 RCA - Resolve Addresses of Statically Linked Modules

This Natural profile parameter controls the dynamic loading of modules that have been defined as statically linked to the Natural nucleus during initialization of the Natural session.

Possible settings	`ON`	At Natural startup, the list of all modules that have been defined as statically linked to the Natural nucleus is inspected and a load request is issued for all modules whose addresses are unresolved. If a load request fails, no error message is issued. **Note:** The use of `RCA=ON` is *not* recommended, as it causes significant processing overhead at Natural startup.
	`OFF`	No dynamic loading of static non-Natural programs is performed.
	module name or list of module names (see ***Examples*** below)	If a single module name or a list of module names is specified, the list of modules that have been defined as statically linked to the Natural nucleus is extended by the specified module(s). A load request is issued for the modules even if they are already linked to the Natural nucleus. This makes it possible to replace modules that have already been linked statically to the Natural nucleus. If a load request fails, an error message is issued.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. Use the profile parameter `CSTATIC` to define modules as being statically linked to the Natural nucleus.

2. If the external name of the module to be loaded is different from the internal one that is used by the `CALL` statement, you can use either the profile parameter `RCALIAS` or the macro `NTALIAS` to define which external name is to be used for the load request.
3. Under CICS: A PPT entry has to be defined to allow the load request for module that is to be dynamically loaded. Statically linked modules are called using standard linkage conventions rather than `EXEC CICS LINK` requests.

Examples

For a single module name, you may specify:

```
RCA=module-name
```

In a list of module names, each module name must be separated from the next by a comma and the list must be enclosed within parentheses:

```
RCA=(module-name1,module-name2,module-name3)
```

201

RCALIAS - External Name Definition for Statically Linked Modules

This Natural profile parameter can be used to define the external names of modules defined by profile parameter `RCA` as being statically linked to the Natural nucleus and requested for dynamic loading during the initialization of a Natural session. `RCALIAS` corresponds to the `NTALIAS` macro in the Natural parameter module.

Possible settings	See *RCALIAS Parameter Syntax*.	
Default setting	`OFF`	See *RCALIAS Parameter Syntax*.
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTALIAS` is used instead.
Specification within session	no	

RCALIAS Parameter Syntax

The parameter syntax of `RCALIAS` is as follows:

RCALIAS=(*internal-module-name*,*external-module-name*,*internal-module-name*,*external-module-name*)

Or:

RCALIAS=OFF

Where:

Syntax Element	Value	Explanation
internal-module-name	1 - 12 characters.	The internal name of the module (used by the `CALL` statement) that must also be defined by the profile parameter `RCA` or `CSTATIC` (only if `RCA=ON`).
external-module-name	1 - 12 characters.	The corresponding external alias name for the load request during session initialization.
-	`OFF`	No external names for modules defined by profile parameter `RCA` as being statically linked to the Natural nucleus are defined.

Note: With the profile parameter `RCALIAS` you can specify multiple name pairs; see *Example of RCALIAS Parameter*.

NTALIAS Macro Syntax

The `NTALIAS` macro is specified as follows:

```
         NTALIAS internal-module-name,external-module-name
```

Where:

Syntax Element	Value	Explanation
internal-module-name	1 - 12 characters.	The internal name of the module (used by the `CALL` statement) that must also be defined by the profile parameter `RCA` or `CSTATIC` (only if `RCA=ON`).
external-module-name	1 - 12 characters.	The corresponding external alias name for the load request during session initialization.

Notes:

1. A separate `NTALIAS` macro must be specified for each name pair; see ***Examples of NTALIAS Macros***.
2. The value `OFF` cannot be specified with the `NTALIAS` macro; it can only be specified dynamically with the profile parameter `RCALIAS`.

Example of RCALIAS Parameter

```
RCA=(MODULE1,MODULE2),RCALIAS=(MODULE1,ALIAS1,MODULE2,ALIAS2)
```

Examples of NTALIAS Macros

```
         NTALIAS MODULE1,ALIAS1
         NTALIAS MODULE2,ALIAS2
```

202 RCFIND - Handling of Response Code 113 for FIND Statement

This Natural profile parameter specifies the action to be taken if Adabas Response Code 113 (requested ISN not found) is returned during the execution of a `FIND` statement processing loop.

Possible settings	`ON`	Response Code 113 causes the program to be terminated.
	`OFF`	Response Code 113 will be ignored, and processing of the `FIND` loop will continue by reading the next record.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

203 RCGET - Handling of Response Code 113 for GET Statement

This Natural profile parameter specifies the action to be taken if Adabas Response Code 113 (requested ISN not found) is returned during the execution of a `GET` statement.

Possible settings	`ON`	Response Code 113 causes the program to be terminated.
	`OFF`	Response Code 113 will be ignored, the system variable `*ISN` will be set to `0`, and processing will continue.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

204 RDACT - (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

205 RDC - Configure the Natural Data Collector

This Natural profile parameter can be used to configure the Natural Data Collector and its trace recording function, which is used by the SYSRDC utility and the Profiler utility. It corresponds to the `NTRDC` macro in the Natural parameter module.

Possible settings	See ***RDC Parameter Syntax***.	
Default setting	`OFF`	See ***RDC Parameter Syntax*** and ***Keyword Subparameters***.
Dynamic specification	yes	The parameter `RDC` can be specified dynamically only. In the Natural parameter module, use the macro `NTRDC`.
Specification within session	yes	See *Calling the CMRDC Interface* to start or stop the trace recording and to select the events (see also the keyword subparameter `EVENT`).

The following topics are covered below:

RDC Parameter Syntax

The `RDC` parameter is specified as follows:

RDC=(**ON**,*keyword-subparameter*=*value*,*keyword-subparameter*=*value*,**...**)

Or:

RDC=**OFF**

Where:

Value	**Explanation**
`ON`	The trace recording is started automatically during Natural session start.
`OFF`	The user must start trace recording by one of the means described in *Activating the Natural Data Collector* in the *SYSRDC Utility* documentation or in the *Profiler Utility* documentation. This is the default value.
keyword-subparameter	See ***Keyword Subparameters***.

Note: If specified, `ON` or `OFF` must be the first value.

NTRDC Macro Syntax

The `NTRDC` macro is specified as follows:

```
         NTRDC ON,                                                     *
               EVENT=value,                                            *
               EXIT=value,                                             *
               FNAT=value,                                             *
               SIZE=value,                                             *
               XEVENT=value
```

Or:

```
         NTRDC OFF
```

See ***Keyword Subparameters***.

For a description of the values `ON` and `OFF`, see ***RDC Parameter Syntax***.

Keyword Subparameters

`EVENT` | `EXIT` | `FNAT` | `SIZE` | `XEVENT`

EVENT - Natural Data Collector Events to be Recorded

`EVENT=(value)` determines the Natural Data Collector events to be recorded in the Natural Data Collector buffer.

Value	Explanation
`(event,event,...)`	Each *event* can be a one- or two-letter event type. Only the specified events are recorded. **Note:** 1. For information on possible event types, see *Data-Collecting Events* in the *SYSRDC Utility* documentation. 2. When only one value is specified, the enclosing brackets can be omitted. 3. If only one letter is specified but there are more events beginning with this letter, then all these events are recorded. For example, `EVENT=P` means that the events `PL`, `PS` and `PT` are recorded; that is, the definition is equivalent to `EVENT=(PL,PS,PT)`.

Value	Explanation
`ALL`	All events are recorded, provided that the Natural Data Collector buffer was defined properly. This is the default value.

Notes:

1. This definition can be replaced within the Profiler utility or by using `CALL 'CMRDC' 'T'` in any Natural program.
2. For further information, see *Profiler Utility* and *Calling the CMRDC Interface* in the *SYSRDC Utility* documentation.

EXIT - Define Natural Data Collector User Exits

`EXIT=(value)` is used to define user exits for the Natural Data Collector of the SYSRDC utility and, optionally, a work area size for each user exit.

Value	Explanation

`(name,name,...)`
or
`(name,size,name,size,...)`
	`name` is the name of the user exit.
	There is no default.

	Optionally, the `size` of the exit work area can be specified after the exit name. Possible size values: `400 - 32760`. The default size value is `400`.

Notes:

1. If linked, the exit gets control from the Natural Data Collector at certain points within Natural. Specific session information is passed to the exits.
2. If the subparameter `EXIT` is specified dynamically, the exits must be defined in the profile parameter `CSTATIC` or `RCA` (`RCA` can also be specified dynamically).
3. As an alternative to this keyword subparameter, you can use the equivalent Natural profile parameter `RDCEXIT`.
4. For details, refer to *User Exits for External Monitoring/Accounting* in the *SYSRDC Utility* documentation.

FNAT - Trace Recording in the Natural System File

`FNAT=`*`value`* controls the trace recording while Natural system file programs are executing.

Value	Explanation
`ON`	Trace recording when running in the Natural system file.
`OFF`	No trace recording in the Natural system file. This is the default value.

SIZE - Size of the Natural Data Collector Buffer

`SIZE=`*`value`* specifies the size of the Natural Data Collector buffer, which is used by the SYSRDC utility and the Profiler utility, and it controls the trace recording function of the Natural Data Collector.

Value	Explanation
`0` or `2 - 128`	Buffer size (in KB). ▪ To activate the Natural Data Collector (without trace recording), you specify `SIZE=2`. ▪ To also activate the trace recording, set `SIZE` to a value greater than `2`. ▪ The Profiler utility needs a buffer size greater than 2 to activate the trace recording. ▪ If the requested space is not available, the Natural Data Collector cannot be used.
`0`	Deactivates the Natural Data Collector. This is the default value.

Note: As an alternative to this keyword subparameter, you can use the equivalent Natural profile parameter `RDCSIZE`.

XEVENT - Natural Data Collector Events to Call User Exits

`XEVENT=(`*`value`*`)` determines the Natural Data Collector events for which user exits are called. This can help to reduce the CPU consumption and the number of TCB/SRB switches if a Natural for zIIP add-on product is installed.

Value	Explanation
`(event,event,...)`	Each *event* can be a one- or two-letter event type. User exits are called for the specified events only. **Note:** 1. For information on possible event types, see *Data-Collecting Events* in the *SYSRDC Utility* documentation. 2. When only one value is specified, the enclosing brackets can be omitted. 3. If only one letter is specified but there are more events beginning with this letter, then all these events are recorded. For example, `XEVENT=P` means that the events `PL`, `PS` and `PT` are recorded; that is, the definition is equivalent to `XEVENT=(PL,PS,PT)`.
`ALL`	All user exits are called for all events. This is the default value.

Examples of RDC Parameter

```
RDC=(ON,EVENT=(D,P,U),XEVENT=(P,I,A),SIZE=80,FNAT=ON)
```

```
RDC=(ON,EVENT=(PS),SIZE=80,FNAT=ON,EXIT=(MYEXIT,2000,RDCEX1))
```

Example of NTRDC Macro

```
         NTRDC ON,                                                     *
               EVENT=(D,P,U),                                          *
               XEVENT=(P,I,A),                                         *
               SIZE=80,                                                *
               FNAT=ON,                                                *
               EXIT=(MYEXIT,2000,RDCEX1)
```

206 RDCEXIT - Define Natural Data Collector User Exits

This Natural profile parameter is used to define user exits for the Natural Data Collector of the SYSRDC utility and, optionally, a work area size for each exit.

`RDCEXIT` can only be specified dynamically. In the Natural parameter module, you need to use the equivalent macro `NTRDC` and its subparameter `EXIT` instead.

Possible settings	See ***RDCEXIT Parameter Syntax***.	
Default setting	none	
Dynamic specification	yes	**Note:** To dynamically specify `RDCEXIT`, the exits must be defined in the profile parameters `CSTATIC` or `RCA` (`RCA` can also be specified dynamically). Optionally, the size of the exit work area can be specified after the exit name.
Specification within session	no	

Notes:

1. Alternatively, you can use the equivalent Natural subparameter `EXIT` of profile parameter `RDC` or macro `NTRDC`.
2. If linked, the exit gets control from the Natural Data Collector at certain points within Natural. Specific session information is passed to the exits. For details, refer to *User Exits for External Monitoring/Accounting* in the *SYSRDC Utility* documentation.

RDCEXIT Parameter Syntax

RDCEXIT=(*name*,*name*,...)

Where:

name is the name of the user exit. In the Natural parameter module, the specified exit names will automatically be appended to the list of specifications contained in the profile parameter `CSTATIC`.

There is no default value.

Or:

Optionally, the *size* of the exit work area can be specified after the exit name.

RDCEXIT=(*name*,*size*,*name*,*size*,...)

Possible *size* values: `400 - 32760`.

The default *size* value is `400`.

Example:

```
RDCEXIT=(MYEXIT,2000,RDCEX1)
```

207 RDCSIZE - Size of Buffer for the Natural Data Collector

This Natural profile parameter specifies the size of the Natural Data Collector buffer, which is used by the SYSRDC utility and the Profiler utility, and it controls the trace recording function of the Natural Data Collector.

Possible settings	`2 - 128`	Buffer size (in KB). ■ To activate the Natural Data Collector (without trace recording), you specify `RDCSIZE=2`. ■ To also activate the trace recording, set `RDCSIZE` to a setting greater than `2`. ■ The Profiler utility needs a buffer size greater than 2 to activate the trace recording. ■ Automatic activation of the trace recording function during session initialization takes place only if profile parameter `RDC` or macro `NTRDC` is set to `ON`. ■ If the requested space is not available, the Natural Data Collector cannot be used.
	`0`	Deactivates the Natural Data Collector.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. As an alternative, you can use the equivalent Natural subparameter `SIZE` of profile parameter `RDC` or macro `NTRDC`.

2. Alternatively, you can specify the `RDCSIZE` value in Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.
3. For further information, see *Activating the Natural Data Collector* in the *SYSRDC Utility* documentation.

208 RDNODE - (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

209 RDPORT - (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

210 READER - z/VSE System Logical Units for Input

This Natural profile parameter specifies the z/VSE system logical units which are to be used by Natural for input.

Possible settings	See *READER Parameter Syntax*.	
Default setting	See *READER Parameter Default Setting*.	
Dynamic specification	yes	
Specification within session	no	

Note: Overwriting of a system logical unit number only applies if the relevant file is a card file.

READER Parameter Syntax

```
READER=(n,device,...)
```

Where:

Syntax Element	Value/Explanation
n	`0` for `CMSYNIN`. `1` for `CMOBJIN`
device	Either `SYSRDR` or `SYSIPT`.

READER Parameter Default Setting

The default setting is:

```
READER=(0,SYSRDR,1,SYSIPT)
```

Notes:

1. By default, the primary input stream (`CMSYNIN`) is read from `SYSRDR` and the input stream (`CMOBJIN`) is read from `SYSIPT`, if required.
2. If `CMSYNIN` or `CMOBJIN` are disk or tape files, the associated `READER` subparameter is ignored.

211 RECAT - Dynamic Recataloging

This Natural profile parameter specifies the action to be taken if Natural detects an inconsistency in the global data area definition as defined in the program currently being executed; that is, the global data area in the program does not correspond to the definition of the global data area currently in use.

Possible settings	`ON`	An error message is issued if an inconsistency in a program and/or global data area is detected. If the adjusted object invokes an object from a steplib library that also has to be adjusted, the object from the steplib library will be copied to the library of the invoking object. Natural automatically adjusts the object and disables the system commands `CATALOG` and `SAVE`. `RECAT=ON` is not possible for programs cataloged with the Natural Optimizer Compiler.
	`OFF`	Natural issues an error message.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

212

REINP - Issue Internal REINPUT Statement for Invalid Data

This Natural profile and session parameter can be used to prevent an internal `REINPUT` for invalid data.

Possible settings	`ON`	An internal `REINPUT` statement is issued when invalid data have been entered.
	`OFF`	An internal `REINPUT` statement is not issued when invalid data have been entered.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. By default, Natural automatically issues an internal `REINPUT` statement if invalid data have been entered in response to an `INPUT` statement. With this parameter, you can switch this mechanism off. This will allow you to handle such input errors yourself in your application.
2. Within a Natural session, the profile parameter `REINP` can be overridden by the session parameter `REINP`.

213 RELO - Storage Thread Relocation

This Natural profile parameter controls the relocation of the Natural thread after a terminal I/O in a thread environment (CICS, Com-plete, IMS TM, *open*UTM and Natural as a Server).

Possible settings	`ON`	The Natural thread and all the buffers contained therein can be relocated to another storage area if the original storage area has been occupied by another user after a terminal I/O.
	`OFF`	No relocation is performed. The Natural thread and all the buffers therein remain located at the same virtual address after the terminal I/O. **Note:** This setting applies to CICS, Com-plete and server environments only. In all other thread environments, Natural cannot guarantee that the thread remains located at the same address.
	`FORCE`	This will force a relocation of the Natural thread and all the buffers contained therein to another storage area. This can be useful for testing purposes in some environments. **Note:** This setting does not apply under *open*UTM.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Notes for CICS:

- When using `TYPE=GETM` threads under CICS, `RELO=OFF` has the same effect as the `PSEUDO=OFF` setting of the `PSEUDO` profile parameter. See also `TYPE` (thread type for group) in the section *Natural under CICS* in the *TP Monitor Interfaces* documentation.
- A specification of `RELO=OFF` is ignored for *Natural Server Sessions under CICS* using `TYPE=GETM` threads.

214 RFILE - File for Recordings

This Natural profile parameter specifies where recordings (that is, the data recorded by the Recording function) are stored.

Possible settings	`SPAD`	Recordings will be stored in the scratch-pad file. (If no scratch-pad file is defined, the recordings will be stored in the system file `FUSER`.)
	`FUSER`	Recordings will be stored in the system file `FUSER`.
	`FNAT`	Recordings will be stored in the system file `FNAT`.
Default setting	`SPAD`	
Dynamic specification	yes	
Specification within session	no	

Note: For details on the Recording function, see *Recording Utility* in the *Utilities* documentation.

215 RI - Release ISNs

This Natural profile parameter specifies whether ISNs (internal sequence numbers) for records which were read and placed in hold status but were not updated are to be retained in hold status.

Possible settings	`ON`	Natural releases the ISN of each record which has been placed in hold status but was not updated (for example because the record was rejected as a result of a `WHERE` clause or an `ACCEPT`/`REJECT` statement). This reduces the number of ISNs which are contained in the hold queue. **Note:** This may, however, cause additional performance overhead as an Adabas call is required for each ISN released.
	`OFF`	The ISN of each record which has been placed in hold status is *not* released until the end of the transaction.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Note: In nested processing loops, a record which due to `RI=ON` is released in an inner processing loop is no longer kept in hold status for any outer loop.

216 RJESIZE - Initial Size of NATRJE Buffer

This Natural profile parameter specifies the initial size of the `NATRJE` buffer.

Possible settings	`1 - 2097151`	Buffer size in KB. **Note:** If the initial size is not sufficient, Natural automatically increases (repeatedly, if necessary) the buffer size in increments of 8 KB.
	`0`	Disables the NATRJE utility.
Default setting	`8`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. With the NATRJE utility (described in the *Utilities* documentation), JCL jobs can be collected and then submitted all at once. `RJESIZE` is used to set the initial size of the buffer which holds the JCL jobs before they are submitted.
2. Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.

217 RM - Retransmit Modified Fields

This Natural profile parameter controls the retransmission of modified fields.

Possible settings	`ON`	Natural always sends back all modified fields.
	`OFF`	Natural sends back modified fields only if they have been changed.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. Some TP monitors translate input data automatically to upper-case characters.
2. As Natural's screen optimization only retransmits modified data back to the screen, the TP-monitor translation may cause input for a field which has been modified not to be retransmitted.

218 RNCONST - Renumber Line Numbers in Constants

This Natural profile parameter can be used to renumber the line number references in alphanumeric and Unicode constants within a Natural source. See also *Renumbering of Source-Code Line Number References* in the *Programming Guide*.

Possible settings	`ON`	The line number references within alphanumeric and Unicode constants are renumbered.
	`OFF`	The line number references within alphanumeric and Unicode constants are not renumbered. They remain as they are.
Default setting	`OFF`	
Dynamic specification	no	
Specification within session	no	
Applicable statements	none	
Applicable command	`RENUMBER`	

Note: The setting of `RNCONST` affects the execution behavior of the `RENUMBER` system command.

219

ROSY - Read-Only Access to System Files

This Natural profile parameter disables modifications on the Natural system files `FNAT`, `FUSER`, `FDIC` and `FSEC`.

Possible settings	`ON`	No data can be written to, modified on or deleted from the system files. Natural issues an error message instead of performing any action that would modify any of these system files.
	`OFF`	Data can be written to, modified on and deleted from the system files.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If your system files are specified as read-only (`ROSY=ON`), the Natural utilities/functions Recording and `NATPAGE` cannot be used, because they write data to the Natural system files `FNAT` and/or `FUSER`.
2. Therefore, it is recommended that you allocate and use a so-called scratch-pad file to hold these temporary data. The scratch-pad file is optional and must be defined as recoverable by using the macro `NTLFILE` or the profile parameter `LFILE`. The above functions then will write their data to this file rather than to the system files `FNAT` and/or `FUSER`.
3. With `ROSY=OFF`, a scratch-pad file should also be defined if you use the Recording and `NATPAGE` functions with database transaction logic, as that might lead to unpredictable results with `FNAT/FUSER`.
4. If a system file is specified as read-only (`RO` Option) in the corresponding profile parameter `FNAT`, `FUSER` or `FSEC`, it is not possible to override this setting by setting `ROSY=OFF`, in order to enable modifications or updates on the affected write-protected system file.

220 RPC - Remote-Procedure-Call Settings

This Natural profile parameter is used to control the handling of Natural RPC. It corresponds to the macro `NTRPC` in the Natural parameter module.

Possible settings	See *RPC Parameter Syntax*.	
Default setting	none	See: *Keyword Subparameters for Client and Server* *Keyword Subparameters for Client Only* *Keyword Subparameters for Server Only*
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTRPC` is used instead.
Specification within session	no	

The following topics are covered below:

Note: For information on Natural RPC, see the *Natural RPC (Remote Procedure Call)* documentation and *Setting Up a Natural RPC Environment*.

RPC Parameter Syntax

The parameter syntax of `RPC` is as follows:

RPC=(*keyword-subparameter*=*value*,*keyword-subparameter*=*value*,...)

For names and values of *keyword-subparameter*, see ***Keyword Subparameters for Client and Server***, ***Keyword Subparameters for Client Only*** and ***Keyword Subparameters for Server Only***.

NTRPC Macro Syntax

On the client side, the syntax of the `NTRPC` macro in the Natural parameter module is as follows:

```
         NTRPC ACIVERS=value,                                          *
               AUTORPC=value,                                          *
               COMPR=value,                                            *
               CPRPC=value,                                            *
               DFS=value,                                              *
               MAXBUFF=value,                                          *
               RDS=value,                                              *
               RPCSDIR=value,                                          *
               RPCSIZE=value,                                          *
               SERVER=value,                                           *
```

```
                TIMEOUT=value,                                                 *
                TRYALT=value
```

On the server side, the syntax of the `NTRPC` macro in the Natural parameter module is as follows:

```
          NTRPC ACIVERS=value,                                                 *
                CPRPC=value,                                                   *
                LOGONRQ=value,                                                 *
                MAXBUFF=value,                                                 *
                NTASKS=value,                                                  *
                RPCSIZE=value,                                                 *
                RPCUCT=value,                                                  *
                SERVER=value,                                                  *
                SRVCMIT=value,                                                 *
                SRVNAME=value,                                                 *
                SRVNODE=value,                                                 *
                SRVRTRY=value,                                                 *
                SRVTERM=value,                                                 *
                SRVUSER=value,                                                 *
                SRVWAIT=value,                                                 *
                TRACE=value,                                                   *
                TRANSP=value
```

See *Keyword Subparameters for Client and Server*, *Keyword Subparameters for Client Only* and *Keyword Subparameters for Server Only*.

Keyword Subparameters for Client and Server

The following keyword subparameters are available for both client and server:

`ACIVERS` | `CPRPC` | `MAXBUFF` | `RPCSIZE` | `SERVER`

ACIVERS - Define API Version for Use with EntireX Broker ACI

This keyword subparameter has been deprecated and is ignored. The highest possible API version is negotiated dynamically by the RPC nucleus and EntireX.

CPRPC - Define Code Page Name

`CPRPC=`*`value`* specifies the name of the code page used by the EntireX Broker.

Note: Currently, it applies only to the Natural RPC facility when the transport protocol ACI (that is EntireX Broker) is used.

Possible settings	1 - 40 characters	Valid code page name of EntireX Broker.
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. For information on the EntireX Broker, refer to the section about Software AG's Internationalization in the EntireX Broker documentation.
2. See also *Unicode and Code Page Support*, *Configuration and Administration of the Unicode/Code Page Environment* and *Profile Parameters and Macros*.

MAXBUFF - Default Buffer Size

`MAXBUFF=`*`value`* specifies the default buffer sizes in a Natural RPC environment.

Possible settings	`1 - 2097147`, but less than or equal to `RPCSIZE=`*`value`* `- 4`	Default buffer size in KB. The default buffer size must be equal to or less than the value (minus 4) specified with the keyword subparameter `RPCSIZE` (for the server side, see *Dependency on Number of Parameters on Server Side*).
	`0`	No buffer is allocated.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. On the server side, `MAXBUFF` determines the size of the buffer provided by the server to receive the client request and send back the result. The buffer must be large enough to hold the largest data area received by all client requests and all results sent back to the client. If the size of the buffer is too small for a request, a temporary buffer with the required size is allocated and used

for this request. For further information, see *Interface Objects and Automatic RPC Execution* in the *Natural RPC (Remote Procedure Call)* documentation.

2. On the client side, `MAXBUFF` determines the size of the buffer provided for the execution of Natural RPC calls. This buffer is used to build the client request and receive the result from the server. The buffer must be large enough to hold the largest data area received by all client requests and all results sent back to the client. If the size of the buffer is too small for a request, a temporary buffer with the required size is allocated and used for this request.
3. On the client side, you need not specify `MAXBUFF` if you use an interface object generated with the SYSRPC utility and `COMPAT NONE`, and if the parameters neither contain dynamic fields, nor X-arrays or group structures.
4. The size of the data exchanged between the client and server is provided by the **Interface Object Generation** function of the SYSRPC utility. To calculate the size for RPC execution, you may use the `SYSRPC CSMASS` command; see *Calculating Size Requirements* in the *SYSRPC Utility* documentation.

Dependency on Number of Parameters on Server Side

On the server side, the difference between `RPCSIZE` and `MAXBUFF` depends on the maximum number of parameters *n* in the PDA and can be calculated as follows:

- If group structures are present:

  `MAXBUFF = RPCSIZE - (3 + n/10)`

- If no group structures are present:

  `MAXBUFF = RPCSIZE - (3 + n/20)`

  Example:

  If *n*=`100` and `RPCSIZE=128`, then `MAXBUFF=120`.

RPCSIZE - Size of Buffer Used by Natural RPC

`RPCSIZE=`*`value`* specifies the size of the buffer used by the Natural RPC.

Possible settings	`1 - 2097151`	Buffer size in KB. **Note:** If the specified size is not large enough, the buffer will be increased on request.
	`0`	Natural RPC cannot be used.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

SERVER - Start Natural Session as an RPC Server Session

`SERVER=`*`value`* specifies whether or not the Natural session will be started as an RPC server session.

Possible settings	`ON`	The Natural session will be started as an RPC server session.
	`OFF`	The Natural session will not be started as an RPC server session.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Keyword Subparameters for Client Only

The following keyword subparameters are available for the client only:

`AUTORPC` | `COMPR` | `DFS` | `RDS` | `RPCSDIR` | `TIMEOUT` | `TRYALT`

Note: See also *Set the RPC Client-Specific Natural Parameters* in the *Natural RPC (Remote Procedure Call)* documentation.

AUTORPC - Automatic Natural RPC Execution

`AUTORPC=`*`value`* determines whether or not Natural RPC will automatically try to execute a subprogram remotely (on the server side) which was not found locally (on the client side).

Possible settings	`ON`	Natural RPC will automatically try to execute it remotely.
	`OFF`	Natural RPC will not automatically try to execute it remotely. **Note:** With `AUTORPC=OFF`, you can execute `CALLNAT`s remotely using interface objects.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	At runtime, this value can be overwritten using the **Parameter Maintenance** function of the SYSRPC utility.

Note: If you want to use a remote `CALLNAT` statement to execute a subprogram on an EntireX RPC server, we strongly recommend that you set `AUTORPC=OFF` and use an interface object. For details, see *Interface Objects and Automatic RPC Execution* in the *Natural RPC (Remote Procedure Call)* documentation.

COMPR - Set RPC Buffer Compression

`COMPR=value` sets the RPC buffer compression.

Possible settings	`0`	No compression will be performed.
	`1`	The send buffer contains modifiable fields and output fields and the format buffer. The reply buffer contains modifiable fields and input fields.
	`2`	Same as `COMPR=1`, additionally the reply buffer also contains the format buffer.
Default setting	`1`	
Dynamic specification	yes	
Specification within session	yes	At runtime, this value can be overwritten using the Parameter Maintenance function of the SYSRPC utility.

`COMPR` is effective only, if the automatic Natural RPC execution is used (`AUTORPC=ON`) and the `CALLNAT` is executed without an interface object. If an interface object is used, the compression has already been set during interface object generation. For details, see *Using Compression* in the *Natural RPC (Remote Procedure Call)* documentation.

DFS - Specify RPC Client's Default Server Address

`DFS=value` defines an RPC default server address by specifying up to 5 positional subparameters.

Possible settings	See *DFS Subparameter Syntax*.	
Default setting	none	See *DFS Subparameter Syntax*.
Dynamic specification	yes	
Specification within session	yes	At runtime, this value can be overwritten using the Natural application programming interface `USR2007N`.
Application programming interface	`USR2007N`	See *Application Programming Interfaces for Use with Natural RPC* in the *Natural RPC (Remote Procedure Call)* documentation and *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

`DFS` determines the server name, the server node, the logon indicator and the transport protocol. The default server address will be used only if no appropriate server is found in the service directory. For further information, see *Specifying RPC Server Addresses* in the *Natural RPC (Remote Procedure Call)* documentation.

DFS Subparameter Syntax

The `DFS` syntax is as follows:

```
DFS=(server-name,server-node,logon-indicator,transport-protocol-name,service-directory-indicator)
```

Where:

Syntax Element	Value	Explanation
server-name	1 - 32 characters	Valid server name. See also the keyword subparameter `SRVNAME`. There is no default, the value must be specified.
server-node	1 - 192 characters	Node name. See also the keyword subparameter `SRVNODE`. There is no default, the value must be specified.
logon-indicator	`L`	The client initiates a Natural logon to the server with the library name of the current library on the client. On Windows platforms: Instead of specifying `L`, check the selection box.
	(blank)	Blank means that no server logon will be executed. If nothing is specified, this is the default.
transport-protocol-name	`ACI`	The transport protocol to be used. `ACI` is the only possible value and the default.
service-directory-indicator	`SERVDIR`	A service directory must be present before the keyword subparameter `DFS` is evaluated.
	`NOSERVDIR`	No service directory is used before the keyword subparameter `DFS` is evaluated; that is, a service directory needs not be available on the client side. If nothing is specified, `SERVDIR` is the default.

RDS - Define Remote Directory Server

`RDS=`*`value`* allows you to define up to 10 remote directory servers in a Natural RPC environment. For each remote directory server, you can specify up to 5 positional subparameters.

Possible settings	See ***RDS Subparameter Syntax***.	
Default setting	none	See ***RDS Subparameter Syntax***.
Dynamic specification	yes	
Specification within session	no	

RDS Subparameter Syntax

The `RDS` syntax is as follows:

For 1 Server:

```
RDS=(server-name,server-node-name,subprogram,logon-indicator,transport-protocol-name)
```

For 2 to 10 Servers:

```
RDS=((server-name,server-node-name,subprogram,logon-indicator,transport-protocol-name)(serve
name,subprogram,logon-indicator,transport-protocol-name)...(server-name,server-node-name,subp
```

Where:

Syntax Element	Value	Explanation
server-name	1 - 8 characters	The server name. There is no default, the value must be specified.
server-node-name	1 - 8 characters	The server node name. There is no default, the value must be specified.
subprogram	1 - 8 characters	The name of the subprogram titled `CALLNAT`, which is to be used as an interface. The default name is `RDSSCDIR`.
logon-indicator	`L`	The client initiates a Natural logon to the server with the library name of the current library on the client. On Windows platforms: Instead of specifying `L`, check the selection box.
	(blank)	Blank means that no server logon will be executed. If nothing is specified, this is the default.
transport-protocol-name	`ACI`	The name of the transport protocol to be used. `ACI` is the only possible value and the default.

RPCSDIR - Library for Service Directory

`RPCSDIR=`*`value`* specifies the name of the Natural library (or one of its steplibs) used by the RPC client at runtime.

Possible settings	1 - 8 characters	Valid Natural library name.
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

`RPCSDIR` is evaluated by the SYSRPC utility functions Service Directory Maintenance and Server Command Execution.

TIMEOUT - Wait Time for RPC Server Response

`TIMEOUT=value` specifies the number of seconds the client is to wait for an RPC server response.

Possible settings	`0 -32767`	Timeout in seconds. **Note:** If this time is exceeded, the remote procedure call will be terminated with a corresponding error message.
Default setting	`55`	
Dynamic specification	yes	
Specification within session	yes	At runtime, this value can be overwritten using the Parameter Maintenance function of the SYSRPC utility.

TRYALT - Try Alternative Server Address

`TRYALT=value` specifies whether an RPC client should try to execute an RPC request on an alternative server or not.

Possible settings	`ON`	If a request could not be executed on the node you specified, the RPC client tries to find an alternative server address to send that request to.
	`OFF`	No such attempt will be made.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	At runtime, this value can be overwritten using the Parameter Maintenance function of the SYSRPC utility.

For further information, see *Specifying RPC Server Addresses* in the *Natural RPC (Remote Procedure Call)* documentation.

Keyword Subparameters for Server Only

The following keyword subparameters are available for the server only:

`LOGONRQ` | `NTASKS` | `RPCUCT` | `SRVCMIT` | `SRVNAME` | `SRVNODE` | `SRVRTRY` | `SRVTERM` | `SRVUSER` | `SRVWAIT` | `TRACE` | `TRANSP`

Note: See also *Set the RPC Server-Specific Natural Parameters* in the *Natural RPC (Remote Procedure Call)* documentation.

LOGONRQ - Logon for RPC Server Request Required

`LOGONRQ=value` determines whether or not logon data are required for an RPC server request.

Possible settings	`ON`	A logon is required; that is, the server only accepts requests from clients which include logon data in the RPC server request. For conversational requests, the logon data is only necessary when the conversation is opened. **Note:** If the Natural RPC server runs under Natural Security, you are strongly recommended to set `LOGONRQ=ON`. For further information, see *Using Natural RPC with Natural Security* in the *Natural RPC (Remote Procedure Call)* documentation.
	`OFF`	A logon is *not* required. Logon data will be processed nevertheless.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

Note: For Natural clients, the logon data can be requested either by setting the `LOGON` option of the `SYSRPC` Service Directory Maintenance or by using the **logon indicator** of the keyword subparameter `DFS`.

NTASKS - Number of Server Tasks to be Started

`NTASKS=value` specifies the minimum number of server tasks to be started during server initialization, and the maximum number of server tasks that max be active at any time.

Possible settings	See *NTASKS Subparameter Syntax*.	
Default setting	`1,1`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. `NTASKS` applies only to servers started in batch mode under z/OS or z/VSE, and to servers started by an RPC server front-end under CICS.
2. If the server has to handle a large number of client requests, you can use this keyword subparameter to improve the throughput by starting multiple (identically named) replicas of the same server task.
3. For further information, see the *Natural RPC (Remote Procedure Call)* documentation, and especially *Considerations for Mainframe Natural RPC Servers with Replicas*.

NTASKS Subparameter Syntax

```
NTASKS=(min,max)
```

Or:

```
NTASKS=min
```

Where:

Syntax Element	Value	Explanation
min	`1 - 99`	Minimum number of server tasks to be started during server initialization.
max	`1` - *n* `0` (unlimited)	Maximum number of server tasks that may be active at any time. **Note:** The maximum number *max* of server tasks applies only to servers started by an RPC server front-end.

Note: If you do not specify a *max* value, *max* defaults to `1` if *min* is set to `1`, and it defaults to `0` if *min* is set to a value greater than `1`.

RPCUCT - Translate Subprogram Name into Upper Case

`RPCUCT=`*value* specifies whether or not the Natural RPC server will translate the name of the remote `CALLNAT` subprogram to be executed into upper case.

Possible settings	`ON`	The name of the remote `CALLNAT` subprogram to be executed by the Natural RPC server will be translated into upper case before the `CALLNAT` is invoked. With this option, non-Natural RPC clients are supported that use mixed case characters in their subprogram names. **Note:** On UNIX and Windows platforms, an implicit upper case translation is already done by Natural itself. `RPCUCT=ON` is therefore the compatibility mode for Natural RPC servers on mainframes and Natural RPC servers on UNIX and Windows platforms.
	`OFF`	The name of the remote `CALLNAT` to be executed by the Natural RPC server is not changed. If the name contains lower case characters a NAT00082 error message is to be expected.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

SRVCMIT - Server Commit Time

`SRVCMIT=`*`value`* specifies the time at which a Natural RPC server automatically commits an RPC conversation or a non-conversational RPC request.

Possible settings	`B`	The Natural RPC server automatically commits a database transaction before the reply is sent to the client. **Note:** If the reply fails, the database transaction is already committed.
	`A`	The Natural RPC server automatically commits a database transaction after the reply has been successfully sent to the client. **Note:** If the reply fails, the database transaction is rolled back.
Default setting	`B`	
Dynamic specification	yes	
Specification within session	no	

Note: This parameter is only evaluated if the profile parameter `ETEOP` is set to `ON`.

SRVNAME - Name of RPC Server

`SRVNAME=`*`value`* specifies the name of the RPC server, with which it registers on the node specified with the keyword subparameter `SRVNODE`.

Possible settings	1 - 32 characters	Valid server name. In case of an EntireX Broker node, the value of `SRVNAME` corresponds to the value of the `SERVER` attribute of a service entry in the broker attribute file, as shown below: `CLASS=RPC, SERVICE=CALLNAT, SERVER=`*`srvname`* See ***Example***.
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Example:

```
SRVNAME='PRODUCTION_SERVER'
```

SRVNODE - Name of Node

`SRVNODE=`*`value`* specifies the name of the node upon which an RPC server registers.

Possible settings	1 - 192 characters	Node name. In case of an EntireX Broker node, a node name may refer to an Entire Net-Work node or to a TCP/IP address. Note that the EntireX Broker stub in use must support the naming notation. For details about the structure of node names and their support by the EntireX Broker stubs, refer to the EntireX documentation. See *Examples*.
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Examples:

The examples below are based on the EntireX notation.

```
SRVNODE=ETB001                             /* Entire Net-Work node */
SRVNODE=PCBROKER                           /* host name for a TCP/IP address */
SRVNODE=’157.189.160.95:1958:TCP’          /* TCP/IP address with port number */
SRVNODE=’tcpip://host.com:1958’            /* host name with port number */
```

Notes:

1. If a host name is used for the TCP/IP address, the name must either be known to your DNS server or it must be defined in the hosts file of your TCP/IP configuration.
2. If the port number is omitted, either a default port number is used by the EntireX Broker stub or a host name must be used, and the host name must be known to your DNS server or must be defined in the services file of your TCP/IP configuration.

SRVRTRY - Number of Connect/Reconnect Attempts

`SRVRTRY=`*`value`* specifies the number of attempts for an RPC server to connect/reconnect (`REGISTER`) to an EntireX Broker that is not active, and the wait time between two successive attempts.

Possible settings	See *SRVRTRY Subparameter Syntax*.	
Default setting	`0,60`	No attempts.
Dynamic specification	yes	
Specification within session	no	

SRVRTRY Subparameter Syntax

The `SRVRTRY` syntax is as follows:

```
SRVRTRY=(attempts,wait-time)
```

Or:

```
SRVRTRY=attempts
```

Note: If only a value for *`attempts`* is specified, the parentheses may be omitted.

Where:

Syntax Element	Value	Explanation

`attempts`	`0`
	or
	`1 - 2147483647`
		Number of attempts to connect/reconnect to an EntireX Broker that is not active (EntireX Broker message 02150148).
		Note:
		1. The specification of *`attempts`* enables you to start a Natural RPC server before the required EntireX Broker has been started and to shut down an EntireX Broker temporarily without implicitly terminating all Natural RPC servers.
		2. If the EntireX Broker is still not active after the number of attempts specified in *`attempts`* or if *`attempts`* is zero, the RPC server terminates.

`wait-time`	`0`
	or
	`1 - 3600`
		Wait time in seconds between two successive attempts.

Examples:

1. `RPC=(SRVRTRY=(20,10))`

   Or:

   ```
           NTRPC SRVRTRY=(20,10)
   ```

   20 attempts with a wait time of 10 seconds between two successive attempts.
2. `RPC=(SRVRTRY=500)`

   Or:

   ```
           NTRPC SRVRTRY=500
   ```

   500 attempts with a wait time of 60 seconds between two successive attempts.

Note: For further information, see the Natural *Natural RPC (Remote Procedure Call)* documentation, and especially *Considerations for Mainframe Natural RPC Servers with Replicas*.

SRVTERM - Server Termination Event

`SRVTERM=`*`value`* specifies the event at which a Natural RPC server is automatically terminated.

Possible settings	`NEVER`	A Natural RPC server is never automatically terminated. **Note:** To terminate a Natural RPC server, refer to *Terminating a Natural RPC Server* and *Using Application Programming Interface USR2075N* (for the EntireX Broker Service) in the *Natural RPC (Remote Procedure Call)* documentation.
	`TIMEOUT`	A Natural RPC server is automatically terminated if the wait time for the next client request outside of an RPC conversation is exceeded. **Note:** `TIMEOUT` should only be set if you use an Attach Manager to dynamically start Natural RPC servers on request.
Default setting	`NEVER`	
Dynamic specification	yes	
Specification within session	no	

SRVUSER - User ID for RPC Server Registry

`SRVUSER=`*`value`* specifies the user ID needed to register a Natural RPC server on the node specified with the keyword subparameter `SVRNODE`.

Note: In case of an EntireX Broker node, `SRVUSER` is also used to logon to the EntireX Broker. A password is either taken from Natural Security (see `*NSC` below) or specified via the application programming interface `USR2072N`.

Possible settings	*`user-ID`*	Valid user ID. 1 to16 characters.
	`*USER`	If `SRVUSER` is set to `*USER`, the Natural server uses the current Natural user ID (see system variable `*USER`) to logon to the node.
	`*NSC`	If `SRVUSER` is set to `*NSC` and Natural Security is installed, the Natural server uses the current Natural user ID (see system variable `*USER`) and the password defined for this user ID in Natural Security to logon to the node.
Default setting	*`timestamp`*	If the user ID is omitted, the timestamp will be used.
Dynamic specification	yes	
Specification within session	no	

SRVWAIT - Wait Time of RPC Server

`SRVWAIT=`*`value`* specifies the number of seconds the server is to wait for a Natural RPC client request.

Possible settings	`0`
	or
	`1 - 32767`
		Wait time in seconds.
		Note:
		1. If this time is exceeded, the RPC server is informed by the node to which the RPC server has registered. The RPC server writes a corresponding message to the Natural RPC server trace file, and continues to wait for an RPC client request.
		2. If TCP/IP is used to communicate with the node, a non-zero value will also avoid an indefinite wait in TCP/IP if the node cannot respond for any reason.

Default setting	`0`	Unlimited wait time. **Note:** In case of an EntireX Broker node, the wait time is set to the `SERVER-NONACT` value of the corresponding Entirex Broker attribute file.
Dynamic specification	yes	
Specification within session	no	

TRACE - Define Trace Level for Natural RPC Servers

`TRACE=value` activates the Natural RPC trace facility and determines the trace level to be used.

Possible settings	`0`	Nothing is traced.
	`1`	Only messages (inclusive Natural errors) are traced.
	`(1,E)`	Messages are traced in the event of an error only.
	`2`	All messages and data from/to client are traced.
	`(2,E)`	Messages and data from/to client are traced in the event of an error only.
	`3 - 9`	The values `3 - 9` are also accepted. These values are for future use and behave like `TRACE=2`.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

For further information, see *Using the Server Trace Facility* in the *Natural RPC (Remote Procedure Call)* documentation.

TRANSP - Server Transport Protocol

`TRANSP=value` specifies which server transport protocol is used. If ACI is used, you can additionally specify the transport method.

Note: The use of `TRANSP` is no longer required as you may now specify the full node name with `SRVNODE`. It is still supported for compatibility reasons.

Possible settings	`ACI`	ACI is used. The transport method is defined by the EntireX Broker.
	`(ACI,TCP)`	ACI is used with TCP/IP.
	`(ACI,NET)`	ACI is used with Entire Net-work, i.e. using the Adabas protocol.
	`(ACI,TCP-NET)`	Trying to use ACI with TCP. If not available, ACI is used with NET.
	`(ACI,NET-TCP)`	Trying to use ACI with NET. If not available, ACI is used with TCP.
Default setting	`ACI`	
Dynamic specification	yes	
Specification within session	no	

RPC Parameter Example

For the client:

```
RPC=(RPCSIZE=80,MAXBUFF=30,AUTORPC=ON,DFS=(MYSERV,MYNODE,,ACI))
```

For the server:

```
RPC=(RPCSIZE=80,MAXBUFF=30,SRVNAME=MYSERV,SRVNODE=MYNODE,SERVER=ON)
```

NTRPC Macro Example

For the client:

```
         NTRPC RPCSIZE=80,                                              *
               MAXBUFF=30,                                              *
               AUTORPC=ON,                                              *
               DFS=(MYSERV,MYNODE1,,ACI),                               *
               RDS=((SRVX,NODEX),(SRVY,NODEY))
```

For the server:

```
         NTRPC RPCSIZE=80,                                              *
               MAXBUFF=30,                                              *
               SRVNAME=MYSERV,                                          *
               SRVNODE=MYNODE,                                          *
               SERVER=ON
```

221 RUNSIZE - Size of Runtime Buffer

This Natural profile parameter specifies the size of the Natural runtime buffer.

Possible settings	`10 - 64`	Buffer size in KB.
Default setting	`16`	
Dynamic specification	yes	
Specification within session	no	

Note: Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.

The Natural runtime buffer contains information on the following items:

- the file translation table (profile parameter `TF`),
- log information of the most recent command,
- the environment stack (for user settings),
- information about the last database error,
- PF-key names when working in SAA mode,
- various internal work areas and control information.

222 SA - Sound Terminal Alarm

This Natural profile and session parameter specifies whether the terminal alarm feature is to be used.

Possible settings	`ON`	The terminal alarm sound is output each time the user is prompted for input by Natural. **Note:** The use of this feature requires that the terminal alarm hardware feature has been installed for the terminal.
	`OFF`	No terminal alarm is used for input prompting, however, the alarm may still be activated with the *ALARM Option* of the `REINPUT` statement.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Note: Within a Natural session, the profile parameter `SA` can be overridden by the session parameter `SA`.

223 SB - Selection Box

Selection boxes in an `INPUT` statement are available on mainframe computers only. For other platforms, selection boxes may be defined in the map editor only.

Selection boxes can be attached to input fields. They are a comfortable alternative to help routines attached to fields, since you can code a selection box direct in your program. You do not need an extra program as with help routines.

You may define a selection box clause for every `INPUT` variable of type alpha, regardless if this field is an input or output field, or both.

The syntax is:

SB=*operand1* [,*operand1*]...

Where:

operand1 represents a value operand which is used to fill up the selection box with items.

Operand	**Possible Structure**						**Possible Formats**													**Referencing Permitted**	**Dynamic Definition**
operand1	C	S	A				A													yes	no

With SB, you specify the values to be displayed within the selection box.

To assign a selection box to a field, specify the attribute `SB` for an alpha `INPUT` field in your Natural program using the following example syntax:

```
INPUT #FLD (SB='value1', #ITEM1, #ITEM2(1:3), #ITEM3(*))
```

The following topics are covered below:

Syntactical Considerations

It is possible to assign both a selection box and a help routine to a field.

Selection boxes can be defined for every variable field in an `INPUT` statement. Exceptions are the following:

System Variables	For example: `*PROGRAM`, `*COM`
Named Constants (mainframe only)	Defined with a `CONST` clause of `DEFINE DATA` statement.

In addition to the `SB` attribute, other attributes can be defined as well, for example: `AD` or `CD`.

The selection box field does not have to be modifiable, as is the case with `AD=A` or `AD=M`. In other words, it is possible to provide a selection box (and select values) even for a write-protected output field, such as `AD=O`. If you use `AD=O`, the user is forced to choose from a set of predefined values, which themselves appear in a selection box.

Runtime Considerations

Selection Box Position

When a program containing a selection box is executed, the selection box is positioned on the screen according to the same positioning algorithm used for help windows; that is, the size and position of the selection box are determined automatically, "near" the field.

Selection Box Attributes

The color and intensified attributes assigned to the field are also applied to the values displayed in the corresponding selection box.

Edit Masks in Selection Boxes

If an edit mask has been defined for the field, the edit mask is applied to all selection box values.

To define an edit mask for a field:

Using the `INPUT` statement, you can define an edit mask for a field. This is demonstrated in following code example.

```
DEFINE DATA
LOCAL
1 A(A4)
END-DEFINE
MOVE 'ABCD' TO A
*
SET KEY PF1 = HELP
FORMAT KD=ON
*
INPUT A (AD=M EM=X.X.X.X SB='1234','WXYZ')
WRITE A
END
```

Selection Box Line Sizes

The line size of the selection box matches the field length to which the box corresponds.

If a value intended for the selection box exceeds the line size of the selection box, the value is truncated.

Sequence of Selection Box Values

Selection box values are displayed in the order they appear in the `SB` attribute.

Features

How a Selection Box is Displayed

For a field with attached selection box, an indicator “V” is displayed next to the field.

Invoking Selection Boxes

To open a selection box, there are two ways to open a selection box:

- Enter a question mark (?) in the V-field and press ENTER.

  Or position the cursor on the V-field and press the help key (for example, PF1), if you have defined it to do so. See the next section for more details.

To define a help key to invoke the selection box, you can define a help-key (for example, PF1) to make invoking the selection box much simpler.

This is done by adding the following line of code to your program:

```
SET KEY PF1=HELP
```

Scrolling in a Selection Box

There are two ways to scroll in a selection box:

- By putting the cursor on the `MORE` line and pressing ENTER.
- Or by using the terminal commands `%W-` and `%W+` assigned to PF-keys (for example, PF7/PF8).

Selecting a Value in a Selection Box

A value is selected from the selection box and copied into the field by putting the cursor on the value and pressing ENTER.

Duplicate Lines in a Selection Box

Lines with the same content which directly follow each other are suppressed.

For example, the following code

```
INPUT #FLD (SB='123', '456', 'XYZ', 'XYZ', 'XYZ', 'ABC', 'DEF')
```

produces the following output to the selection box:

```
123
456
XYZ
ABC
DEF
```

In the preceding example, `XYZ` is only displayed once. The other occurrences are considered redundant since they directly follow one another.

However, this line of code

```
INPUT #FLD (SB='123', 'XYZ', '456', 'XYZ', 'ABC', 'XYZ', 'DEF')
```

produces the following output to the selection box:

```
123
XYZ
456
XYZ
ABC
XYZ
DEF
```

In this case, all three occurrences of `XYZ` are displayed, since they do not directly follow one another.

Blank Lines in Selection Boxes

A blank line is only displayed the first time it appears; all subsequent blank lines are suppressed.

Restrictions

The number of operands in the `SB` clause is limited to 20.

The maximum number of values in a selection box is 248. When that limit has been reached, further values are not displayed. No error message is issued when the limit has been exceeded.

224

SCTAB - Scanner Characters

This Natural profile parameter can be used to overwrite the definitions in the scanner character-type table `NTSCTAB`, which is contained in the configuration module `NATCONFG`. `SCTAB` corresponds to the `NTSCTAB` macro in the Natural parameter module.

Possible settings	See *SCTAB Parameter Syntax*.	
Default setting	As specified within the macro `NTSCTAB` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTSCTAB` is used instead.
Specification within session	no	

Note: The `NTSCTAB` scanner character type table defines the properties of characters used in mask definitions for the `MASK` option and recognized as delimiters in the `EXAMINE` and `SEPARATE` statements.

SCTAB Parameter Syntax

The `SCTAB` parameter is specified as follows:

```
SCTAB=(character1,attribute-type1,attribute-type2,...,character2,attribute-type1,attribute-type2,...)
```

Note: It is possible to specify more than one character in the list of values. You must enclose the entire string of characters/attribute pairs in parentheses.

Or:

```
SCTAB=OFF
```

Where:

Syntax Element / Value	Explanation	
character	You specify a character, and after it its attribute type(s). You can specify the character either as the one-byte character itself (enclosed in apostrophes) or as the hexadecimal representation of that character.	
attribute-type	Attribute types can be:	
	`UPPER`	upper-case alphabetical
	`LOWER`	lower-case alphabetical
	`NUM`	numeric
	`HEX`	hexadecimal

Syntax Element / Value	Explanation	
	`ALFANUM`	alphanumeric
	`SPECIAL`	special
	`NDELIM`	non-delimiter
`OFF`	With `SCTAB=OFF`, all (static and dynamic) definitions are reset to the values specified within the macro `NTSCTAB` in `NATCONFG`.	

Note: If the `CP` profile parameter is set to a value other than `OFF`, values specified with `SCTAB` are ignored, except for the attribute `SPECIAL`. The attribute `SPECIAL` is a Natural-only attribute. Therefore, it can be set by `SCTAB`. All other attributes are set according to the code page. See also *Translation Tables* in the *Unicode and Code Page Support* documentation.

NTSCTAB Macro Syntax

The `NTSCTAB` macro is specified as follows:

```
        NTSCTAB=character,attribute-type,attribute-type...
        NTSCTAB=character,attribute-type,attribute-type...
```

Notes:

1. For an explanation of the syntax elements, see ***SCTAB Parameter Syntax***.
2. The value `OFF` cannot be specified with the macro `NTSCTAB`, but only dynamically with the profile parameter `SCTAB`.
3. For each character to be overwritten, you have to specify a separate `NTSCTAB` macro; see ***Examples of NTSCTAB Macro***.

Example of SCTAB Parameter

```
SCTAB=(5E,LOWER,NDELIM,'ß',SPECIAL,7B,SPECIAL,'Ä',UPPER,NDELIM)
```

Examples of NTSCTAB Macro

```
         NTSCTAB  5E,LOWER,NDELIM
         NTSCTAB  'ß',SPECIAL
         NTSCTAB  7B,SPECIAL
         NTSCTAB  'Ä',UPPER,NDELIM
```

225

SELUNIT - Activate Selected Natural Features

This Natural profile parameter selects and activates or deactivates single or multiple new or changed Natural features supplied as selectable units.

For more information, see *Selectable Units for New Natural Features* in the *Operations* documentation.

Possible settings	`OFF`	All selectable units (Natural features) are deactivated; any keyword subparameter settings are switched off.
	`U`*n*`={ON|OFF}`	One or more selectable units (Natural features) are activated (`ON`) or deactivated (`OFF`) with the keyword subparameter `U`*n*: see ***SELUNIT Parameter Syntax***. The `SHOWSU` system command lists all selectable units available in your Natural environment and the selectable units that have been specified and activated. You can also use the `SU` line command/keyword of the `SYSPROD` system command to check whether selectable units are active.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

The setting of `SELUNIT` is ignored if no selectable units are provided in a Natural version.

Selectable Units Available

Selectable units are currently not available.

SELUNIT Parameter Syntax

The following syntax applies:

```
SELUNIT={OFF|({Un={ON|OFF}},...) }
```

Where:

n is a number from 1 to 24, each representing a specific feature as announced in the current Natural *Release Notes* for Mainframes.

Example of SELUNIT Parameter

```
SELUNIT=(U1=ON,U4=OFF)
```

226 SENDER - Screen Output Destination for Asynchronous Processing

This Natural profile parameter specifies the destination where output from an asynchronous application is to be displayed.

Possible settings	1 to 8 characters	Output destination, for example, printer.
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Natural under CICS, Com-plete, IMS TM and *open*UTM.
2. The destination specified applies to hardcopy output and primary reports.
3. Any additional reports are sent to the destinations specified with the `DEFINE PRINTER` statement (just as in a synchronous online session).

Platform-specific Characteristics

The following platform-specific characteristics apply:

Platform:	**Comment:**
CICS	The profile parameter `SENDER` specifies the CICS transient data (TD) destination and the terminal or printer for terminal output from asynchronous sessions. If the specified destination does not exist, the session output is sent to the specified terminal or printer. If the specified terminal or printer does not exist either, the session terminates abnormally. The default terminal output format for asynchronous sessions is a 3270 data stream. If the `SENDER` terminal specification is not a 3270 device, the Natural application must switch to the

Platform:	Comment:
	correct terminal type before the first output statement (for example, by specifying `SET CONTROL 'T=PRNT'` for a printer or by starting with profile parameter `TTYPE=PRNT`). If you are routing all output to a (spool) destination, such as CSSL, the Natural application must be switched to line mode, for example, by specifying `SET CONTROL 'T=`*xxxx*`'` or by starting with profile parameter `TTYPE=`*xxxx*, where *xxxx* is `BTCH` or `ASYL`. In this case, two other profile parameters are relevant: `EJ` and `INTENS`. ■ If you set `EJ=ON`, all lines are routed with a leading ASA control character. ■ With `EJ=OFF`, there is no leading ASA control character. ■ `INTENS` should be set to `1`, particularly if you have set `EJ=OFF`. **Note:** For further CICS-specific functionality, see *Asynchronous Natural Processing under CICS* in the *TP Monitor Interfaces* documentation.
Com-plete	See *Asynchronous Natural Processing under Com-plete/SMARTS*.
IMS TM	The profile parameter `SENDER` specifies the default `LTERM`. This `LTERM` is always used when no other printer has been specified. You should always dynamically define a `SENDER` parameter in the `NIIBOOT` module. This is important when Natural tries to output error messages when starting a session: if no `SENDER` parameter is specified, there is no valid `LTERM` and `NATIMS` terminates the session.
*open*UTM	The profile parameter `SENDER` specifies the ID for the initialization of an asynchronous transaction; that is, the ID which identifies the transaction as asynchronous. If output from an asynchronous transaction is to be printed, the setting specified with the `SENDER` parameter also identifies the printer on which the output is to be printed. **Note:** For further *open*UTM-specific functionality, see *Asynchronous Transaction Processing under openUTM* in the *TP Monitor Interfaces* documentation.

Note: For further information, see also the profile parameter `OUTDEST` and Asynchronous Processing in the *Operations* documentation.

227 SF - Spacing Factor

This Natural profile and session parameter specifies the default number of spaces to be inserted between field settings of columns on Natural reports created using a `DISPLAY` statement.

Possible settings	`1 - 30`	Number of spaces. **Note:** The `SF` parameter cannot be set to `0`; that is, at least one blank character must be placed between report columns.
Default setting	`1`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Within a Natural session, the profile parameter `SF` can be overridden by the session parameter `SF`.
2. Under Natural Security, the setting of this parameter can be overridden by the *Session Parameters* option of the Library Profile.
3. See also *Column Spacing - SF Parameter and nX Notation* in the *Programming Guide*.

228 SG - Sign Position

This session parameter determines whether or not a sign position is to be allocated for a numeric field.

Possible settings	`ON`	A sign position will be allocated.
	`OFF`	No sign position will be allocated. **Note:** 1. `SG=OFF` causes numeric fields with negative values to be output without a minus (-) sign. 2. `SG=OFF` does not prevent you from entering negative values in input fields.
Default setting	`ON`	
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`INPUT`
	`PRINT`
	`WRITE`	

Applicable command	none	

Notes:

1. If the `EM` (edit mode) parameter is specified, it overrides the `SG` parameter.
2. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

Example:

```
FORMAT SG=OFF
```

229 SHAPED - Control of Character Shaping

This Natural profile and session parameter determines whether base characters in their basic forms (unshaped) are converted to their shaped forms before they are stored.

Character shaping is required to correctly represent characters of a bidirectional language (such as Arabic), for example, when using a browser or editing text with the NaturalONE source editor.

Possible settings	`ON`	Unshaped characters are converted to their shaped forms; shaped characters are not converted.
	`OFF`	Unshaped characters are not converted to their shaped forms; shaped characters are converted to their unshaped forms.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	

See also *Bidirectional Language Support* in the *Unicode and Code Page Support* documentation.

230 SKEY - Storage Key for Program Execution

This Natural profile parameter only applies on z/OS and z/VSE platforms.

`SKEY` determines the storage key of the TP monitor interface (Com-plete or CICS) under which Natural runs.

For details on Com-plete storage protect keys, see the related Com-plete documentation. For details on CICS key execution, see the related IBM documentation.

Possible settings	`ON`	Natural runs in the storage key of the TP monitor under which the Natural program executes: ■ Under Com-plete: Natural uses the storage protect key 8 of Com-plete to obtain storage for program execution. ■ Under CICS: Natural uses CICS key even if `EXECKEY(USER)` is defined in CICS. `EXECKEY(USER)` with `SKEY=ON` does not have the same effect as `EXECKEY(CICS)`, in particular, when CICS transaction isolation is active.
	`OFF`	Natural runs in a storage key that is different from the TP monitor under which the Natural program executes. Under CICS, Natural uses the execution key (`EXECKEY`) as defined in CICS. We recommend this setting to obtain the maximum benefits from the CICS storage protection facility if used.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	

231

SL - Source Line Length

This Natural profile and session parameter specifies the number of characters to be interpreted on each Natural source code line. This also applies to the line mode editor which is activated with the system command `EDT`.

Possible settings	`20 - 250`	**In batch mode:** The number of characters to be processed on each line in the data sets `CMSYNIN` and `CMOBJIN`. **Note:** For details on these data sets, refer to the operating-system-specific parts of the section *Natural in Batch Mode* in the *Operations* documentation.
		In online mode: The number of characters to be interpreted when using the Natural program editor in `EDT` mode (as activated with the system command `EDT`).
Default setting	`72`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	none	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Within a Natural session, the profile parameter `SL` can be overridden by the session parameter `SL`.

2. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the Library Profile.

232

SLOCK - Source Locking

This Natural profile parameter is used to specify how concurrent updates of Natural source members are to be handled (see also *Locking of Source Objects* in the *Editors* documentation).

Possible settings	`PRE`	Activates locking of source objects that are edited either locally or in a SPoD environment, or using Natural ISPF, or in mixed environments. **Note:** 1. This is the recommended setting when working in mixed environments. 2. In order to lock a source member against concurrent updates, a specific record is written to the Natural system file `FUSER` or `FNAT` (depending on where the source member to be edited is located). For a DDM, the lock record is written to the Natural system file `FDIC`.
	`SPOD`	Locking of source objects occurs only in a remote development environment basing on Natural Single Point of Development (SPoD). **Note:** 1. This setting provides compatibility with previous Natural versions that supported locking under SPoD. 2. In order to lock a source member against concurrent updates, a specific record is written to the Development Server File (`FDIC`) system file. 3. For a DDM, the locked record is written to the Natural system file `FDIC`.
	`POST`	When setting `SLOCK=POST`, the source object which is being edited can be read into the source work area and modified by multiple users. However, only the user who saves a modification first can update the source object. **Note:**

		1. This is done by comparing the time stamp of the source object stored in the database with the time stamp of the source object when it is read into the source work area. 2. All other users receive appropriate error messages when trying to save the source. 3. This concept is not compatible with the SPoD locking concept of previous Natural versions.
	`OFF`	Deactivates all locking mechanisms.
Default setting	`SPOD`	
Dynamic specification	yes	
Specification within session	no	

233 SM - Programming in Structured Mode

This Natural profile and session parameter specifies whether or not structured mode must be used.

Possible settings	`ON`	Forces the use of structured mode syntax.
	`OFF`	Programming can be done in either structured mode or reporting mode.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	none	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. If structured mode (`SM=ON`) is specified by profile parameter `SM`, an attempt to change this setting with system command `GLOBALS` and session parameter `SM` will be rejected (Reporting mode not permitted).
2. Within a Natural session, the profile parameter setting `SM=OFF` can be overridden by the session parameter `SM=ON`.
3. Under Natural Security, the setting of the mode option in the library's security profile determines whether the `SM` profile parameter can be used; see also *Programming mode* in the *Natural Security* documentation.
4. Under Natural Security, this parameter may be disabled by Natural Security to the effect that structured mode is invariably in effect for a given library.

234

SORT - Control of Sort Program

This Natural profile parameter is used to control the sort program used for the processing of `SORT` statements. It corresponds to the `NTSORT` macro in the Natural parameter module.

Possible settings	For an explanation of the individual sort options, see ***SORT Parameter Syntax***; for their possible settings, see ***Keyword Subparameters***.	
Default setting	See ***Keyword Subparameters***.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTSORT` is used instead.
Specification within session	no	

Notes:

1. The keyword subparameters of `SORT` or `NTSORT` can be used to specify options that control the handling of the sort program to be used when a `SORT` statement is executed.
2. The sort program to be used can be either Natural's internal one (the default for all environments) or an external one. The type of sort program used depends on the setting of the keyword subparameter `EXT`.

SORT Parameter Syntax

The `SORT` parameter is specified as follows:

```
SORT=(keyword-subparameter=value,keyword-subparameter=value,...)
```

For names and values of the keyword subparameters, see ***Keyword Subparameters***.

NTSORT Macro Syntax

The `NTSORT` macro is specified as follows:

```
         NTSORT EXT=value,                                          *
               EXTEOJ=value,                                        *
               EXTNAME=value,                                       *
               EXTOPT=value,value,...,                              *
               STORAGE=value,                                       *
               WRKSIZE=value
```

For names and values of the keyword subparameters, see ***Keyword Subparameters***.

Keyword Subparameters

`EXT` | `EXTEOJ` | `EXTNAME` | `EXTOPT` | `STORAGE` | `WRKSIZE`

EXT - Use of External Sort Program

`EXT=`*`value`* specifies if an external sort program is to be used or not.

Value	Explanation
`ON`	An external sort program will be used. The use of an external sort program is possible only in batch environments, including IMS TM/BMP, TSO and TIAM.
`OFF`	The Natural sort program will be used. This is the default value.

Note: We recommend that you set `EXT=ON` if your environment supports the use of an external sort program. This avoids the need to adapt `WRKSIZE` to the amount of data to be sorted.

EXTEOJ - Action in the Event of an Error

`EXTEOJ=`*`value`* specifies the action to be taken if an error is detected during the execution of the external sort program.

Value	Explanation
`ON`	If an error is detected, sort processing is terminated. `ON` requires that the sort program used is able to detect a return code of 16 from both the `E15` and `E35` sort exit routines.
`OFF`	If an error is detected, Natural withholds further calls to the sort program and ignores each record as it is passed to the `E35` sort exit routine (this is the default).

EXTNAME - Name of External Sort Program

`EXTNAME=`*`value`* specifies the name of the external sort program to be used.

Value	Explanation
1 to 8 characters.	External sort program name.
`SORT`	This is the default value.

Note: This subparameter does not apply under BS2000.

EXTOPT - Additional Options for External Sort Program

`EXTOPT=(`*`value`*`,`*`value`*`,...)` specifies additional options for the external sort program.

Note: This subparameter does not apply under BS2000.

Natural generates the necessary field and format parameters and passes them to the external sort program. With `EXTOPT`, you can specify additional parameters to be passed to the external sort program. You can only specify parameters that are part of the control statement syntax of your external sort program.

You can specify up to two option strings which are delimited by a slash (/). The first option string is appended to the `SORT` control statement, the second option string is used to build an `OPTION` control statement. You may omit the option string before or after the slash. If the option string after the slash is omitted no `OPTION` control statement is generated at all.

The whole option string must be enclosed in single quotes ('...'). For compatibility reasons, it is still possible to have the option string enclosed in brackets instead.

For compatibility reasons, a single option string without a leading or trailing slash is handled differently. Depending on the underlying operating system, the options are appended to the following control statements:

z/OS:	`SORT` control statement
z/VSE:	`OPTION` control statement

Example:

The additional parameters can be specified as in the following example:

```
EXTOPT=(SIZE=E2000000,NOEQUALS,DYNALLOC=(3350,8))
EXTOPT='SIZE=E2000000,NOEQUALS,DYNALLOC=(3350,8)'
EXTOPT='SIZE=E2000000,NOEQUALS,DYNALLOC=(3350,8)/NOCHECK'
EXTOPT='/NOCHECK'
EXTOPT='WORK=4/'
```

STORAGE - Type of Storage Medium

`STORAGE=`*`value`* specifies the type of storage medium to be used by Natural's internal sort program if there is not enough storage available in the work pool (`WRKSIZE`). If the number of records exceeds this storage, the internal sort program tries to use intermediate storage to additionally process records.

Value	Explanation
`MAIN`	Only the remaining storage of `WRKSIZE` is used, no other intermediate storage is available. This is the default setting.
`BP`	The sort buffer pool is used as intermediate storage.
`SD`	SD files are used as intermediate storage. This value is evaluated under Com-plete only.
`SMARTS`	SMARTS portable file system is used.

SD Files under Com-plete/SMARTS

- The SD files used for sort processing are allocated as temporary SD files. They are allocated for a stack level. This means, the name syntax of the temporary SD files is:

  ```
  &&STsnnn
  ```

  Where:

`&&`	Indicator for a temporary SD file
`ST`	Standard prefix for the temporary SD file used for sort processing
`s`	Stack level
`nnn`	Sequence number within a single sort run

- SMARTS work files are located in the SMARTS Portable File System. The directory path must be specified with the SMARTS environment variable `$NAT_WORK_ROOT`. For a SMARTS work file used for sort processing, a `SORT` directory is created with subdirectories for each user who performed a sort: `$NAT_WORK_ROOT/SORT/`*`userid`*. The name of the work file used for sort processing corresponds to the name of the temporary SD file under Com-plete.

Usage of Sort Buffer Pool

- The use of a sort buffer pool only makes sense if you cannot further increase the `WRKSIZE` to hold the sort records. This typically applies in online environments with a storage thread of limited size. In all other cases, you only need to specify a sufficient `WRKSIZE`.
- If you want to use a sort buffer pool, define the `SORT` keyword subparameter `STORAGE=BP` to indicate that a sort buffer pool is to be used for any additional storage beyond the defined `WRKSIZE`. Simultaneously, use the profile parameter `BPI` or the parameter macro `NTBPI` to make a buffer pool of `TYPE=SORT` and `NAME=`*`name`* known to Natural, for example: `BPI=(TYPE=SORT,NAME=XYZ)`. When a name is specified with the `BPI` keyword subparameter `NAME`, reference is made to a global sort buffer pool, whereas a local sort buffer pool can be specified by `NAME=' '` (blank).

WRKSIZE - Size of Work Buffer Used by Sort Program

`WRKSIZE=value` specifies the size of the work buffer used by the sort program.

The work buffer specified by `WRKSIZE` accommodates internal sort control data. The remaining storage is used to collect and sort the records. The size of the sort control data depends on various factors (the `WRKSIZE` itself, the sort record length, the number of sort keys, their size and format) and can therefore not be calculated in a formal way.

Value	Explanation
`0` or `10 - 2097151`	Size of work buffer (in KB). `WRKSIZE=0` means that no sort operations can be performed.
`10`	This is the default value.

Note:

Examples of SORT Parameter

Example 1:

```
SORT=(EXT=OFF,WRKSIZE=1024)
```

The internal Natural sort program and a work buffer of 1 MB are used for sort processing.

Example 2:

```
SORT=(EXT=ON,EXTOPT='/EQUALS')
```

An external sort program with an `OPTION EQUALS` control statement is used for sort processing.

Examples of NTSORT Macros

```
         NTSORT EXT=OFF,                                                *
               WRKSIZE=1024

         NTSORT EXT=ON,                                                 *
               EXTOPT='/EQUALS'
```

The first example defines an internal Natural sort program and a work buffer of 1 MB. The second example defines an external sort program with an `OPTION EQUALS` control statement.

235 SOSI - Shift-Out/Shift-In Codes for Double-Byte Character Set

This Natural profile parameter is relevant for Asian countries which use double-byte character sets (DBCS).

If the profile parameter `CP` is set to a multi-byte code page (MBCS), the logical shift-in and shift-out characters will be supplied with the code page and therefore `SOSI` will be ignored.

Possible settings	See ***Positional Subparameters*** below.	
Default setting	none	
Dynamic specification	yes	
Specification within session	no	

The following topics are covered below:

SOSI Parameter Syntax

The parameter syntax of `SOSI` is as follows:

SOSI=(*logical-shift-out*,[*physical-shift-out*],*logical-shift-in*,
[*physical-shift-in*],[*SO/SI-display-length*])

A shift-out code is used to indicate the point at which the code of character representation is shifted out of normal (single-byte) mode into double-byte mode.

A shift-in code is used to indicate the point at which the code of character representation is shifted from double-byte mode back into normal (single-byte) mode.

Positional Subparameters

The positional subparameters are described below:

Subparameter	Explanation
logical-shift-out	The logical shift-out character must be a single character. Specify the hexadecimal representation of the logical shift-out character. Usually, the value `0E` is used for IBM hardware and the value `28` is used for Fujitsu hardware.
physical-shift-out	The value of the physical shift-out character must be chosen depending on the screen hardware that is used. The length of the physical shift-out character may be one or two bytes. Specify the hexadecimal representation of the physical shift-out character.

Subparameter	Explanation
	The default value is the logical shift-out character.
logical-shift-in	The logical shift-in character must be a single character. Specify the hexadecimal representation of the logical shift-in character. Usually, the value `0F` is used for IBM hardware and the value `29` is used for Fujitsu hardware.
physical-shift-in	The value of the physical shift-in character must be chosen depending on the screen hardware that is used. The length of the physical shift-in character may be one or two bytes. Specify the hexadecimal representation of the physical shift-in character. The default value is the logical shift-in character.
SO/SI-display-length	The number of bytes occupied on the screen by the physical shift-out/shift-in characters. Possible values are 0 and 1. The default value is 1. For IBM hardware, the value 1 must be used. For Fujitsu hardware, the value 0 must be used.

Conversion of Logical Shift-Out/Shift-In Characters

Logical shift-out/shift-in characters are converted into the corresponding physical shift-out/shift-in characters before data is transferred to the screen.

Physical shift-out/shift-in characters are converted into the corresponding logical shift-out/shift-in characters before data entered on the screen is transferred to the Natural application.

Automatic Adaptation of Translation Tables

If code page support is disabled (that is, the profile parameter `CP` is set to `CP=OFF`), the entries for the logical shift-out/shift-in characters are updated in the translation tables provided by the following macros and profile parameters:

Table	Macro	Profile Parameter
Standard (primary) output translation table	`NTTAB`	`TAB`
Alternative (secondary) output translation table	`NTTAB1`	`TAB1`
Secondary input translation table used when the session parameter `PM` is set to `C`.	`NTTAB2`	`TAB2`
SYS* output translation table	`NTTABL`	`TABL`

If the characters into which the logical shift-out/shift-in characters are to be translated still have their default value (`?` = `X'6F'`) in the respective translation table at Natural startup (that is, they have not been modified by one of the macros or profile parameters mentioned above), they will be updated so that logical shift-out/shift-in characters will not be not translated for input and output.

For detailed information on the translation tables, see *Translation Tables* in the *Operations* documentation.

Compatibility of SOSI Profile Parameter with SO and SI Profile Parameters of Previous Natural Versions

The subparameter logical-shift-out corresponds to the profile parameter `SO` and the subparameter logical-shift-in corresponds to the profile parameter `SI` that were available with previous Natural versions.

Specifying `SOSI=(xx,xx,yy,yy,1)` is equivalent to specifying `SO=xx,SI=yy`.

SOSI Parameter Examples

For IBM hardware, you should use `SOSI=(0E,0E,0F,0F,1)`, which is equivalent to `SOSI=(0E,,0F,,1)`.

For Fujitsu hardware, you should use `SOSI=(28,28,29,29,0)`, which is equivalent to `SOSI=(28,,29,,0)`.

To execute an application that has been created for IBM hardware (with the parameter setting `SOSI=(0E,0E,0F,0F,1)` applied) on Fujitsu hardware without changing the application, use `SOSI=(0E,4028,0F,2940,1)`.

236 SRETAIN - Retain Source Format

This Natural profile parameter specifies the encoding format for new and existing Natural sources when they are saved.

Possible settings	`ON`	The original code page of an existing Natural source is retained. If an existing Natural source without code page information is saved, it will not receive any code page information. When a new Natural source is created, it will be saved in the default code page format as defined with the profile parameter `CP`.
	`OFF`	Natural sources will be saved in the default code page format.
	`(ON,EXCEPTNEW)`	The original code page of an existing Natural source is retained. If an existing Natural source without code page information is saved, it will not receive any code page information. When a new Natural source is created, it will be saved without code page information. `(ON,EXCEPTNEW)` retains the compatibility of newly created Natural sources with existing applications that have been created with earlier Natural versions that did not provide code page support. Note: The value `(ON,EXCEPTNEW)` is supported on mainframe computers only.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. For the code page of a Natural source that is saved or stowed, the resulting encoding depends on the settings of the profile parameters `SRETAIN` and `CP`. See *Code Page Support for Editors, System Commands and Utilities on the Mainframe* in the *Unicode and Code Page Support* documentation.
2. See also *Profile Parameters and Macros* in the `Unicode and Code Page Support` documentation.

237 SSIZE - Size of Source Area Allocated by the Editors

This Natural profile parameter determines the size of the buffer used by the Software AG Editor.

Possible settings	`40 - 512`	Buffer size in KB.
	`0`	**Note:** If `SSIZE=0` or if the required space is not available, the Software AG Editor cannot be used.
Default setting	`64`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. Alternatively, you can use the equivalent Natural profile parameter `DS` or the macro `NTDS` to specify the size of the buffer.
2. If you have defined an Editor work file with a record length greater than 4 KB (default), you should use an `SSIZE` value greater than 64 KB. There are two work file record buffers allocated within the `SSIZE`. Therefore, you should add two times your work file record buffer size minus 4 KB to your `SSIZE`. Example: Your Editor work file has a record length of 10 KB. Then use at least `SSIZE=76` (that is, `64+2*(10-4)`).
3. For further information about the SAG Editor work file, see *Operating the Software AG Editor*, *Editor Work File* in the *Operations* documentation.

238 STACK - Place Data/Commands on the Stack

This Natural profile parameter is used to place data/commands on the Natural stack.

Note:

Possible settings	any character string	See below.
Default setting	`HELLO`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If `STACK` is used, a colon (:) must not be specified with the profile (or session) parameters `DC`, `HI`, `IA`, `ID` and `STACKD`.
2. The stack can contain a sequence of Natural commands and/or user-specified commands, together with their data, for execution at the beginning of the Natural session.
3. The command stack is processed before the user is prompted for input on the screen (TP mode) or data are read from `CMSYNIN`/`CMOBJIN` files; see *Natural in Batch Mode* in the *Operations* documentation.
4. If an `INPUT` statement is encountered during stack processing, the corresponding input screen is generated only if the required input data were not supplied with the command when the stack was created. Any reports generated during stack processing are displayed as usual.
5. Each system or user-defined command can be optionally followed by data which are used to satisfy requests for information required during the processing of the command. The character string provided as data for the `STACK` parameter must be enclosed in parentheses. If the command is a user command (that is, the name of a user program), any data provided resolve the data requirements of `INPUT` statements within the user program.

Conventions:

- Multiple settings for one `INPUT` statement are separated by the input delimiter character specified with the `ID` profile parameter. The default setting is a comma (,).
- Data for multiple `INPUT` statements are separated by a colon (:).
- Commands are separated by the delimiter character specified with the `STACKD` profile parameter. The default setting is a semicolon (;).

Examples:

```
STACK=(LOGON USER1;UCMD1 A,B;UCMD2 C,D:E;FIN)
STACK=OFF                                     No STACK data.
STACK=UCMND Execute command UCMND             No embedded blanks.
STACK=(CMD DATA:DATA;CMD...)                  Place commands/data on stack.
```

Note: Since some commands (for example, `GLOBALS`) do not read parameters by `INPUT`, a blank character should be used rather than a colon to delimit a command from the first parameter data element.

```
STACK='LOGON SYSTEM'
```

Note: Because the macro assembler does not allow embedded blanks within parentheses, the character string must be enclosed in apostrophes when specified as static parameter.

239

STACKD - Stack Delimiter Character

This Natural profile parameter specifies the character to be used as the command delimiter for the `STACK` parameter and for command input under the Natural Development Server (product code: NDV) in a Natural Single Point of Development environment.

Possible settings	any special character	Character to be used as the command delimiter. **Note:** The character must not be the same as the one specified with the `ID` profile/session parameter (input delimiter character), `DC` profile/session parameter (decimal character) or `IA` profile/session parameter (input assign character).
Default setting	`;` (semicolon)	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. To avoid that the value specified for the `STACK` parameter or the data passed as command input under the Natural Development Server is not interpreted as intended, the `STACKD` parameter value should be set to a character that is not contained in the data passed if the data contains the default value of the stack delimiter character (see example below). The `STACKD` parameter should be changed to a character other than the default character if the `ID` parameter has been set to the semicolon. For downward compatibility reasons, this restriction does not apply to `STACKD=;` (the default setting).
2. The character specified may be enclosed with single quotes.
3. If the input delimiter character is to be a comma, it must be specified as `ID=','`, because the comma (,) separates individual parameters.

Example:

```
STACKD='/',ID=';' STACK=(DUMP IOB;+100/FIN)
```

To avoid that the semicolon after `DUMP IOB` is interpreted as a command delimiter, `STACKD` is set to `'/'`.

240 STEPLIB - Additional Steplib Library

This Natural profile parameter specifies the name of an additional Natural steplib (concatenated library) to be used.

Possible settings	1 to 8 characters	Steplib name.
Default setting	`SYSTEM`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Note: For further information, see *Steplib Libraries* and *Search Sequence for Object Execution* in the *Using Natural* documentation.

241 SUBSID - Subsystem ID under z/OS and z/VSE

This Natural profile parameter is available under z/OS and z/VSE only. It identifies the Natural subsystem to be used.

Possible settings	1 - 4 characters	Natural subsystem. **Note:** If you specify less than 4 characters, blanks will be appended so as to get a 4-byte setting.
Default setting	`NAT8`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. For the purposes of the Natural CICS Interface (see `ROLLSRV`, `SIPSERV`, `SUBSID`), the Natural profile parameter `SUBSID` is ignored if it is specified in a parameter string by a profile parameter `SYS` or `PROFILE` or in an alternate Natural parameter module (as specified with the profile parameter `PARM`).
2. For information on the Natural subsystem, see *Natural Subsystem under z/OS* or *Natural Subsystem under z/VSE* in the *Operations* documentation.

242 SYNERR - Control of Syntax Errors

This Natural profile parameter specifies whether or not syntax errors will be passed to the error transaction program.

Possible settings	`ON`	Syntax errors are passed to the error transaction program.
	`OFF`	Syntax errors are not passed to the error transaction program.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR4007N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. The error transaction program is defined either with the profile parameter `ETA` or by a user program by way of assignment to the system variable `*ERROR-TA` or, if Natural Security is installed, within the Natural Security library profile; see *Components of a Library Profile* in the *Natural Security* documentation.
2. For further information, see *Using an Error Transaction Program* in the *Programming Guide*.

243

SYS - Define and Activate a Set of Dynamic Profile Parameters

This Natural profile parameter enables you to activate a set of dynamic profile parameters which is predefined in the Natural parameter module. This avoids the repeated specification of long sequences of profile parameters for the Natural session start.

Possible settings	See *SYS Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	This parameter can only be specified dynamically.
Specification within session	no	

Notes:

1. The specified parameter set must be defined in the Natural parameter module currently active, e.g. in an alternative parameter module, if it is specified by the `PARM` parameter before the `SYS` parameter.
2. A parameter set is evaluated right in its position of `SYS` in the parameter string, as you would have included the defined parameter string at this position instead.
3. As an alternative, you can use the profile parameter `PROFILE` which provides a similar functionality.
4. In the Natural parameter module, you use `NTSYS` macros to predefine sets of dynamic profile parameters. You identify such a set of parameters by giving it a unique set name.

The following topics are covered below:

SYS Parameter Syntax

The parameter syntax of `SYS` is as follows:

```
SYS=set-name
```

Where:

Syntax Element	Explanation
set-name	Is the parameter set name defined by the `NTSYS` macro in the Natural parameter module. Identifies the subsequent set of parameters. The set name can be 1 to 8 characters long and must begin with an alphabetical character.

NTSYS Macro Syntax

The `NTSYS` macro is specified as follows:

```
         NTSYS set-name,'parameter-string'
```

If the length of the parameter string exceeds 255 characters, continue on the next line(s) or define a second or more parameter strings.

```
         NTSYS set-name,'parameter-string',                            *
               'parameter-string',                                     *
               'parameter-string'
```

To specify more than one set of parameters, you must use an `NTSYS` macro for each set.

```
         NTSYS set-name1,'parameter-string'
         NTSYS set-name2,'parameter-string'
```

Where:

Syntax Element	Explanation
set-name	Identifies the subsequent set of parameters. It can be 1 to 8 characters long and must begin with an alphabetical character.
parameter-string	Consists of individual profile parameters and their values.

Notes:

1. The entire set of parameters you specify with an `NTSYS` macro must constitute a valid string of dynamic parameters. The specified parameter string is not checked for validity during the assembly of the Natural parameter module.
2. All parameter strings of one `NTSYS` macro are concatenated to one set of parameters.
3. An apostrophe within a substring is represented by two apostrophes.

Example of NTSYS Macro

```
         NTSYS SET1,'FUSER=(,50),LC=ON,NC=ON,ULANG=2,                  *
               CMPO=(TQMARK=OFF),STACK=(LOGON ULIB1)'
         NTSYS SET2,'FUSER=(,51),ULANG=4,WH=ON,KC=ON,                  *
               STACK=(LOGON ULIB2)'
```

244

SYSCIP - Adabas Cipher Key for Natural System Files

This Natural profile parameter provides an Adabas default cipher key for access to Natural system files (`FNAT`, `FUSER`, `FDIC`, `FSEC`, `FPROF`, `FSPOOL`, `FREG`), which have been loaded with the “ciphered” option.

Possible settings	1 - 8 digits	The Adabas default cipher key for access to password-protected Natural system files. **Note:** 1. The cipher key specified with the `SYSCIP` parameter applies to all Natural system files for which no individual cipher keys are specified. 2. If the Natural system files are not ciphered, set `SYSCIP` to blanks.
	blanks	
Default setting	blanks	
Dynamic specification	yes	**Note:** If you specify the `SYSCIP` parameter dynamically in conjunction with any of the individual system file parameters `FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF` and `FREG`, you must specify the `SYSCIP` parameter *`before`* any individual system file parameter.
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Adabas databases.
2. Cipher keys for individual system files can be specified with the subparameter *`cipher-key`* of the profile parameters `FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF` and `FREG`.

245

SYSPSW - Adabas Default Password for Natural System Files

It provides an Adabas default password for access to Natural system files (`FNAT`, `FUSER`, `FDIC`, `FSEC`, `FPROF`, `FSPOOL` and `FREG`), which have been password-protected.

Possible settings	1 - 8 characters	The Adabas default password for access to password-protected Natural system files. **Note:** 1. If a Natural system file is password-protected, a password which permits updates to the file must be specified. 2. The password specified with the `SYSPSW` parameter applies to all Natural system files for which no individual passwords are specified. 3. If the Natural system files are not password-protected, set `SYSPSW` to blanks. 4. If profile parameter `OPRB` is specified, the `SYSPSW` password is used for the initial Adabas open call and must permit access and/or update to all the files specified in `OPRB`, as required.
	blanks	
Default setting	blanks	
Dynamic specification	yes	**Note:** If you specify `SYSPSW` dynamically in conjunction with any of the individual system file parameters `FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF` and `FREG`, you must specify `SYSPSW` *before* any individual system file parameter.
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Adabas databases.
2. Passwords for individual system files can be specified with the subparameter `password` of the profile parameters `FNAT`, `FUSER`, `FDIC`, `FSEC`, `FSPOOL`, `FPROF` and `FREG`.

246 TAB - Standard Output Character Translation

This Natural profile parameter can be used to overwrite the definitions in the translation table `NTTAB` as contained in the configuration module `NATCONFG`. `TAB` corresponds to the `NTTAB` macro in the Natural parameter module.

Possible settings	See *TAB Parameter Syntax*.	
Default setting	As specified within the macro `NTTAB` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTAB` is used instead.
Specification within session	no	

Notes:

1. The `NTTAB` table in `NATCONFG` is the standard output translation table.
2. If the `CP` profile parameter is set to a value other than `OFF`, values specified with `TAB` are ignored. See also *Translation Tables* in the *Unicode and Code Page Support* documentation.

The following topics are covered below:

TAB Parameter Syntax

The `TAB` parameter is specified as follows:

```
TAB=(a1,a2,b1,b2,c1,c2,...)
```

You specify pairs of characters, the first character of a pair being the character to be translated, the second character of a pair being the character into which the first character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the hexadecimal representation of that character.

Or:

```
TAB=OFF
```

With `TAB=OFF` all (static and dynamic) definitions are reset to the values specified in the macro `NTTAB` in `NATCONFG`.

NTTAB Macro Syntax

The `NTTAB` macro is specified as follows:

```
         NTTAB a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see ***TAB Parameter Syntax***.
2. The value `OFF` cannot be specified with the macro `NTTAB`, but only dynamically with the profile parameter `TAB`.

Example of TAB Parameter

With the `TAB` parameter, you must enclose the entire string of character pairs in parentheses, for example:

```
TAB=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

Example of NTTAB Macro

```
         NTTAB 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

247 TAB1 - Alternative Output Translation

This Natural profile parameter can be used to overwrite the definitions in the translation table `NTTAB1` as contained in the configuration module `NATCONFG`. `TAB1` corresponds to the `NTTAB1` macro in the Natural parameter module.

Possible settings	See *TAB1 Parameter Syntax*.	
Default setting	As specified within the macro `NTTAB1` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTAB1` is used instead.
Specification within session	no	

Note: The `NTTAB1` table in `NATCONFG` is the alternative output translation table for the secondary character set used when the profile/session parameter `PM=C` is set.

TAB1 Parameter Syntax

The `TAB1` parameter is specified as follows:

TAB1=(*a1*,*a2*,*b1*,*b2*,*c1*,*c2*,...)

You specify pairs of characters, the first character of a pair being the character to be translated, the second character of a pair being the character into which the first character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the hexadecimal representation of that character.

Or:

TAB1=OFF

With `TAB1=OFF` all (static and dynamic) definitions are reset to the values specified in the macro `NTTAB1` in `NATCONFG`.

NTTAB1 Macro Syntax

The `NTTAB1` macro is specified as follows:

```
        NTTAB1 a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see ***TAB1 Parameter Syntax***.
2. The value `OFF` cannot be specified with the macro `NTTAB1`, but only dynamically with the profile parameter `TAB1`.

Example of TAB1 Parameter

With the `TAB1` parameter, you must enclose the entire string of character pairs in parentheses, for example:

```
TAB1=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

Example of NTTAB1 Macro

```
        NTTAB1 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

248

TAB2 - Alternative Input Translation

This Natural profile parameter can be used to overwrite the definitions in the translation table `NTTAB2` as contained in the configuration module `NATCONFG`. `TAB2` corresponds to the `NTTAB2` macro in the Natural parameter module.

Possible settings	See *TAB2 Parameter Syntax*.	
Default setting	As specified within the macro `NTTAB2` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTAB2` is used instead.
Specification within session	no	

Note: The `NTTAB2` table in `NATCONFG` is the alternate input translation table for the secondary character set used when the profile/session parameter `PM` is set to `PM=C`.

The following topics are covered below:

TAB2 Parameter Syntax

The TAB2 parameter is specified as follows:

```
TAB2=(a1,a2,b1,b2,c1,c2,...)
```

You specify pairs of characters, the first character of a pair being the character to be translated, the second character of a pair being the character into which the first character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the hexadecimal representation of that character.

Or:

```
TAB2=OFF
```

With `TAB2=OFF` all (static and dynamic) definitions are reset to the values specified in the macro `NTTAB1` in `NATCONFG`.

NTTAB2 Macro Syntax

The `NTTAB2` macro is specified as follows:

```
         NTTAB2 a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see ***TAB2 Parameter Syntax***.
2. The value `OFF` cannot be specified with the macro `NTTAB2`, but only dynamically with the profile parameter `TAB2`.

Example of TAB2 Parameter

With the `TAB2` parameter, you must enclose the entire string of character pairs in parentheses, for example:

```
TAB2=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

Example of NTTAB2 Macro

```
         NTTAB2 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into 'Ü'.

249 TABA1 - EBCDIC-to-ASCII Translation

This Natural profile parameter can be used to overwrite the definitions in the EBCDIC-to-ASCII translation table `NTTABA1` as contained in the configuration module `NATCONFG`. `TABA1` corresponds to the `NTTABA1` macro in the Natural parameter module.

Possible settings	See *TABA1 Parameter Syntax*.	
Default setting	As specified within the macro `NTTABA1` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTABA1` is used instead.
Specification within session	no	

The following topics are covered below:

TABA1 Parameter Syntax

The `TABA1` parameter is specified as follows:

TABA1=(*a1*,*a2*,*b1*,*b2*,*c1*,*c2*,...)

You specify pairs of characters, the first character of a pair being an EBCDIC character to be translated, the second character of a pair being the ASCII character into which the EBCDIC character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the two-byte hexadecimal representation of that character.

Or:

TABA1=OFF

With `TABA1=OFF` all (static and dynamic) definitions are reset to the values specified in the macro `NTTABA1` in `NATCONFG`.

NTTABA1 Macro Syntax

The `NTTABA1` macro is specified as follows:

```
         NTTABA1 a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see ***TABA1 Parameter Syntax***.
2. The value `OFF` cannot be specified with the macro `NTTABA1`, but only dynamically with the profile parameter `TABA1`.

Example of TABA1 Parameter

With the `TABA1` parameter, you must enclose the entire string of character pairs in parentheses, for example:

```
TABA1=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

Example of NTTABA1 Macro

```
         NTTABA1 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

250 TABA2 - ASCII-to-EBCDIC Translation

This Natural profile parameter allows you to overwrite the definitions in the ASCII-to-EBCDIC translation table `NTTABA2` as contained in the configuration module `NATCONFG`. `TABA2` corresponds to the `NTTABA2` macro in the Natural parameter module.

Possible settings	See *TABA2 Parameter Syntax*.	
Default setting	As specified within the macro `NTTABA2` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTABA2` is used instead.
Specification within session	no	

TABA2 Parameter Syntax

The `TABA2` parameter is specified as follows:

```
TABA2=(a1,a2,b1,b2,c1,c2,...)
```

You specify pairs of characters, the first character of a pair being an ASCII character to be translated, the second character of a pair being the EBCDIC character into which the ASCII character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the two-byte hexadecimal representation of that character.

Or:

```
TABA2=OFF
```

With `TABA2=OFF` all (static and dynamic) definitions are reset to the values specified within the macro `NTTABA2` in `NATCONFG`.

NTTABA2 Macro Syntax

The `NTTABA2` macro is specified as follows:

```
         NTTABA2 a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see *TABA2 Parameter Syntax*.
2. The value `OFF` cannot be specified with the macro `NTTABA2`, but only dynamically with the profile parameter `TABA2`.

Example of TABA2 Parameter

With the `TABA2` parameter, you must enclose the entire string of character pairs in parentheses, for example:

```
TABA2=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

Example of NTTABA2 Macro

```
         NTTABA2 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

251 TABL - SYS Library Output Translation

This Natural profile parameter can be used to overwrite the definitions in the translation table `NTTABL` as contained in the configuration module `NATCONFG`. `TABL` corresponds to the `NTTABL` macro in the Natural parameter module.

Possible settings	See *TABL Parameter Syntax*.	
Default setting	As specified within the macro `NTTABL` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTABL` is used instead.
Specification within session	no	

Note: The `NTTABL` table in `NATCONFG` is used to translate output produced by programs located in `SYS...` libraries.

TABL Parameter Syntax

The `TABL` parameter is specified as follows:

```
TABL=(a1,a2,b1,b2,c1,c2,...)
```

You specify pairs of characters, the first character of a pair being the character to be translated, the second character of a pair being the character into which the first character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the two-byte hexadecimal representation of that character.

Or:

```
TABL=OFF
```

With `TABL=OFF` all (static and dynamic) definitions are reset to the values specified within the macro `NTTABL` in `NATCONFG`.

NTTABL Macro Syntax

The `NTTABL` macro is specified as follows:

```
         NTTABL a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see ***TABL Parameter Syntax***.
2. The value `OFF` cannot be specified with the macro `NTTABL`, but only dynamically with the profile parameter `TABL`.

Example of TABL Parameter

With the `TABL` parameter, you must enclose the entire string of character pairs in parentheses, for example:

```
TABL=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

Example of NTTABL Macro

```
         NTTABL 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

252 TC - Trailing Characters

With this session parameter, you can specify trailing characters that are to be displayed immediately to the right of a field output with a `DISPLAY` statement. The width of the output column is enlarged accordingly.

Possible settings	any character	Up to 10 characters may be specified.
Default setting	none	
Specification within session	yes	
Applicable statements	`FORMAT`	Parameter may be specified dynamically with the `FORMAT` statement.
	`DISPLAY`	Parameter may be specified at statement level and/or at element level.
Applicable command	none	

Notes:

1. Trailing characters may optionally be specified enclosed within apostrophes, in which case any characters can be specified. Any character string specified which contains a closing parenthesis or a quotation mark must be enclosed within apostrophes.
2. The parameter `TC` can also be used with U format fields. For information on Unicode format, see also *Unicode and Code Page Support in the Natural Programming Language*, *Session Parameters, EMU, ICU, LCU, TCU versus EM, IC, LC, TC*.
3. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

Examples:

```
FORMAT TC=*
DISPLAY (TC='*B*')
```

253 TCU - Unicode Trailing Characters

With this session parameter, you can specify trailing characters that are to be displayed immediately to the right of a field output with a `DISPLAY` statement. The width of the output column is enlarged accordingly.

Possible settings	any character	Up to 10 characters may be specified.
Default setting	none	
Specification within session	yes	
Applicable statements	`FORMAT`	Parameter may be specified dynamically with the `FORMAT` statement.
	`DISPLAY`	Parameter may be specified at statement level and/or at element level.
Applicable command	none	

Notes:

1. Trailing characters may optionally be specified enclosed within apostrophes, in which case any characters can be specified. Any character string specified which contains a closing parenthesis or a quotation mark must be enclosed within apostrophes.
2. The session parameter `TCU` is identical to the session parameter `TC`. The difference is that the trailing characters are always stored in Unicode format. This allows you to specify trailing characters with mixed characters from different code pages, and assures that always the correct character is displayed independent of the installed system code page.

See also:

- *Parameters to Influence the Output of Fields* in the *Programming Guide*
- *Unicode and Code Page Support in the Natural Programming Language, Session Parameters, EMU, ICU, LCU, TCU versus EM, IC, LC, TC*

254 TD - Time Differential

This Natural profile parameter specifies a time differential to be applied to the Natural time/date setting to ensure that the current local time/date is used, rather than the computer center time/date.

Possible settings	`AUTO`	During session initialization, Natural compares the physical (store clock) and logical (system environment) machine times and uses the difference between the two as the setting for the `TD` parameter. For a time change to take effect for Natural (for example, to change time to summer time or back to winter time), it is therefore sufficient to reset the logical machine time. This affects only those sessions which were started after the time change.

	`+/-`*`hh`*
	`(+/-`*`hh`*`,`*`mm`*`)`
	`(+/-`*`hh`*`,`*`mm`*`,`*`ss`*`)`
		Hours, minutes and seconds from `(-23,59,59)` to `(+23,59,59)`. A plus (optional) or minus sign indicates, whether the `TD` value is to be added or subtracted.
		The specified time is added to or subtracted from the physical machine time to set the time/date to be used by Natural.

	1 to 32 characters	Name of the time zone to be used. This must be defined as a valid time zone in the `NTTZ` macro of the `NATCONFG` module, see *Configuration Tables - Module NATCONFG*. If the time changes according to the DST definition in `NATCONFG` for the time zone, this is honored during a running session.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Note: This parameter is applicable in an environment in which remote nodes are being used in a computer network.

z/VSE-Specific Information

With VSE-type operating systems, `// ZONE` and `//DATE` JCL statements are honored with `TD=AUTO`. This can also affect the setting of the profile parameter `DD`.

See also the profile parameters `YD` and `DD`.

Examples

```
TD=6            (6 hours ahead)
TD=(5,30)       (5 hours and 30 minutes ahead)
TD=(-6,12,30)   (6 hours, 12 minutes and 30 seconds behind)
TD='USA-EST'    (eastern time zone as defined in NTTZ macro)
```

255 TF - Translation of Database ID/File Number

This Natural profile parameter can be used to translate a database ID/file number into another database ID/file number during the execution of an application. It corresponds to the macro `NTTF` in the Natural parameter module.

Possible settings	See *TF Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTF` is used instead.
Specification within session	no	
Application programming interface	`USR1034N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation. * Recommended.
	`USR2005N` *	

Notes:

1. This parameter applies to user files only. It does not apply to system files.
2. This feature is relevant when developing an application in a production environment. It enables you to develop an application in a test database and then transfer the finished application to the production database without having to change or re-compile the application. The Natural objects are cataloged with the production DBID/FNR, but whenever a database access is executed, the production DBID/FNR is translated into the test DBID/FNR according to the `TF` parameter specifications; that is, the test database is used. This means that testing can take place in the actual production environment, but not with production data.
3. The `TF` parameter or the `NTTF` macro can be specified several times so as to specify different combinations of file numbers.

   If multiple `TF` parameters are specified, the following applies:

   - The list of `TF` definitions is searched in the sequence they are defined. The first entry that exactly matches in DBID and FNR (no asterisk) is used.
   - If no exact match is found, the `TF` definitions are searched for a second time. This time, the first wildcard entry (either DBID or FNR is an asterisk) that matches is used.

TF Parameter Syntax

The `TF` profile parameter is specified as follows:

```
TF=(production-dbid,production-fnr,test-dbid,test-fnr)
```

Where:

Syntax Element	Value	Explanation
production-dbid	0 - 254 or 256 - 65535 or *	Identification of the production database. An asterisk (*) can be used as a wildcard for all database IDs. **Note:** Database ID 255 is reserved for logical system files for Software AG products, see profile parameter `LFILE`.
production-fnr	1 - 65535 or *	File number of the production database. An asterisk (*) can be used as a wildcard for all file numbers.
test-dbid	0 - 254 or 256 - 65535 or *	Identification of the test database. An asterisk (*) can be used, which leaves the database ID unchanged.
test-fnr	1 - 65535 or *	File number of the test database. An asterisk (*) can be used, which leaves the file number unchanged.

NTTF Macro Syntax

The `NTTF` macro is specified as follows:

```
         NTTF production-dbid,production-fnr,test-dbid,test-fnr
```

For explanations of the syntax elements and possible values, see ***TF Parameter Syntax***.

Example of TF Parameter

```
TF=(777,39,17,88),TF=(251,*,9,*)
```

Example of NTTF Macro

Equivalent specification in the Natural parameter module:

```
         NTTF 777,39,17,88
         NTTF 251,*,9,*
```

256 THPINIT - Name of Pre-initialized Storage Thread Model

This parameter only applies if Natural for zIIP is installed with an additional module for pre-initialized thread support.

This Natural profile parameter can be used to define the name of a pre-initialized storage thread model to improve processing performance for Natural session initialization.

Possible settings	1 - 8 characters	The name of a pre-initialized storage thread model to be used for session initialization.
Default setting	`' '`	A pre-initialized storage thread model is not used for session initialization.
Dynamic specification	yes	
Specification within session	no	

Static Natural session initialization mainly comprises dynamic profile parameter evaluation and buffer allocation/initialization, which are CPU-intensive and often repeated in exactly the same way for many sessions. The `THPINIT` profile parameter is used to save the contents of the Natural storage thread after static session initialization and to reuse the contents as a storage thread model for other sessions. All subsequent sessions with the same `THPINIT` definition then skip static initialization and run with the defined pre-initialized storage thread model.

`THPINIT` applies to thread environments only. However, if the `THSIZE` profile parameter is used to define a thread for batch or TSO, `THPINIT` can also be used for these environments.

If the Natural Roll Server is used (z/OS only), a pre-initialized storage thread model is only available for the environment (for example, CICS, IMS TM, batch or TSO) under which the model was generated. If the Natural Roll Server is not used, a pre-initialized storage thread model is stored and available in the current region only.

Important Usage Considerations:

- If a pre-initialized storage thread model is used, only the profile parameters `THPINIT` and `NUCNAME` are evaluated. This applies to both static (Natural parameter module) and dynamic (at session start) parameter definitions. As a consequence, warnings or error messages are not returned for any other invalid or erroneous parameters.
- Profile parameter logging (`PLOG=ON`) is not performed for a session that runs with a pre-initialized storage thread model, even when `PLOG` was specified and saved for the original session.
- A generic value specified with the profile parameter `PROFILE` parameter (set to `AUTO`, `PROGRAM` or `TERMINAL`) is assigned to the value of the session for which the pre-initialized storage thread model was saved, but not to the value of the current session.
- For a session that uses a pre-initialized storage thread model, static and dynamic parameter definitions set for the original session (in which the model was saved) can be displayed by using the Natural user exits USR4004N and USR8203N.
- A saved pre-initialized storage thread model cannot be updated or replaced. The `SYSTP R` utility function can be used to display and delete saved pre-initialized thread storage session records from the Natural Roll Server, if available. The user ID prefix of the Natural Roll Server is `$THP`.
- `THPINIT` definitions in parameter profiles (`PROFILE` or `SYS` profile parameter) and alternative Natural parameter modules (`PARM` profile parameter) are ignored. `THPINIT` is only evaluated at an early stage of session initialization, before parameter profiles and alternative parameter modules are evaluated.
- The physical terminal screen size must be equal to the screen size of the pre-initialized storage thread model.
- The current storage thread size must be greater than or equal to the thread size of the pre-initialized storage thread model.

Example:

```
THPINIT=MYTHREAD
```

257 THSEPCH - Thousands Separator Character

This Natural profile and session parameter is used to specify the character to be used as thousands separator at runtime. Then the thousands separator character replaces the dynamic thousands separators in edit masks.

Possible settings	any character	At runtime, the dynamic thousands separator is replaced with this character. **Note:** 1. If the thousands separator character is to be a comma, it must be enclosed in quotes, that is, `THSEPCH=','` when using the dynamic parameter facility, because a comma is used to separate individual parameters. 2. If the thousands separator character is to be a quote, it must be specified as two quotes enclosed in quotes, that is, `THSEPCH=''''`.
Default setting	, (comma)	**Note:** By default, a comma is used as thousands separator.
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	none	
Applicable command	`GLOBALS`	

Note: In the Natural source, the dynamic thousands separator is always represented by a comma (,) or a period (.).

See also:

- Option `THSEP` of system command `COMPOPT` in the *System Commands* documentation.
- Keyword subparameter `THSEP` of profile parameter `CMPO` or macro `NTCMPO`.

- *Customizing Separator Character Displays* in the *Programming Guide*.

258

THSIZE – Thread Size

To improve the processing performance, this Natural profile parameter can be used to define the size of a storage thread for running in z/OS or z/VSE batch mode or under TSO.

Possible settings	`256` - `2097151` or `0`	Thread size in KB.
Default setting	`0`	No thread used.
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter does not have any effect on other environments where threads are used already, for example, server environments.
2. The storage thread is allocated during session start. Natural tries to satisfy all storage requests (`GETMAIN`/`FREEMAIN`) within this thread instead of passing it to the operating system. The thread must be large enough to contain all fixed size buffers. If the thread is full, variable sized buffers may be allocated outside the thread, provided that profile parameter `OVSIZE` allows this.
3. The profile parameter `THSIZE` is ignored if it is specified in a parameter string activated by a `SYS` or `PROFILE` profile parameter or in an alternative Natural parameter module (as specified with the `PARM` profile parameter).
4. If Natural Batch for zIIP is installed and active (z/OS batch and TSO only), an appropriate setting of `THSIZE` can reduce the number of switches into TCB mode, because of the reduced number of physical `GETMAIN`s. The same applies if profile parameter `WPSIZE` is used.

Example:

```
THSIZE=5000
```

259

TMODEL - IBM 3270 Terminal Model

This Natural profile parameter controls the IBM 3270 terminal model number for online environments, for example, under IMS TM.

Possible settings	`0`	The screen size is determined by the environment-dependent driver module. **Note:** 1. If possible, it gets the screen size information from its subsystem. 2. Otherwise, the definitions of default Model 2 are used, for example, under IMS TM or for the Natural Web I/O Interface.
	`2`	The screen size is 24 lines and 80 columns.
	`3`	The screen size is 32 lines and 80 columns.
	`4`	The screen size is 43 lines and 80 columns.
	`5`	The screen size is 27 lines and 132 columns.
	`(lines,cols)`	This syntax is allowed for NWO server terminals only. The number of lines (`lines`) can be 24 through 250, and the number of columns (`cols`) can be 80 through 250.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter is for IBM mainframes only, or for the Natural Web I/O Interface.
2. It is used to determine the number of lines and columns of the terminal screen.

3. If your `TMODEL` specification is incompatible with the physical terminal screen size, the output data may be displayed incorrectly or hardware errors may occur.
4. This Natural profile parameter can be also used under the Natural Development Server (NDV) and for the Natural Web I/O Interface (NWO) server in any operating system environment for defining the terminal screen size for the Natural Web I/O Interface.
5. For further information, refer to the Natural Development Server documentation, Configuring the Natural Development Server, or to the *Natural Web I/O Interface* documentation.
6. The terminal screen size has a direct influence on the storage required for the terminal I/O buffers used by Natural.
7. Under CICS, this parameter is ignored for terminal bound sessions, because the terminal screen size is defined by the CICS terminal control table. If your `TMODEL` specification is incompatible with the physical terminal screen size, the output data may be displayed incorrectly or hardware errors may occur.

260 TPF (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

261 TQ - Translate Quotation Marks

This parameter has been replaced by the keyword subparameter `TQMARK` of profile parameter `CMPO`.

262 TRACE - Define Components to be Traced

This Natural profile parameter can be used to define the components for which trace data are to be written. It corresponds to the macro `NTTRACE` in the Natural parameter module.

Caution: Do not use this parameter without prior consultation of Software AG Support.

Possible settings	See *TRACE Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTTRACE` is used instead.
Specification within session	no	

Notes:

1. This Natural profile parameter is intended primarily for Software AG internal use for debugging purposes. It *does not* activate trace recording.
2. To activate trace recording, use the profile parameters `ITRACE` (internal trace) or `ETRACE` (external trace). During the session, you can use the corresponding terminal commands `%TRI` and `%TRE`.
3. The setting lists of multiple `TRACE` parameter specifications are *not* concatenated; that is, a `TRACE` parameter overrides any previously specified `TRACE` parameter and any `NTTRACE` macro definitions.

TRACE Parameter Syntax

The `TRACE` parameter is specified as follows:

```
TRACE=(trace-id,trace-id,...)
```

Where:

Syntax Element	Value	Explanation
trace-id	1-8 bytes each.	Trace IDs define the names of the Natural components to be traced. Component names have to be entered in upper case.

NTTRACE Macro Syntax

The `NTTRACE` macro is specified as follows:

```
        NTTRACE trace-id,trace-id,...
```

Notes:

1. For an explanation of the syntax elements, see ***TRACE Parameter Syntax***.
2. Multiple specifications of the `NTTRACE` macro are concatenated to one trace list.

Example of TRACE Parameter

```
TRACE=(NATGETM,NATFREM,DYNPARMS)
```

This specification defines traces to be written for the Natural nucleus components "Storage Acquisition", "Storage Release" and "Dynamic Parameter Evaluation".

Example of NTTRACE Macro

```
        NTTRACE NATGETM,NATFREM,DYNPARMS
```

This specification defines traces to be written for the Natural nucleus components "Storage Acquisition", "Storage Release" and "Dynamic Parameter Evaluation".

263 TS - Translate Output from Programs in System Libraries

This Natural profile and session parameter is used to translate output from Natural system libraries (that is, libraries whose names begin with `SYS`) using a translation table. This may be necessary for locations which have non-standard lower-case usage (for example, Middle East or Far East countries).

Important: The `TS` parameter applies only to primary output (`CMPRINT`, see Natural in Batch Mode in the *Operations* documentation).

Possible settings	`ON`	Output is translated. **Note:** With `TS=ON`, the profile parameter `LC=OFF` and the session parameter `AD=T`, both of which translate input to upper case, are ignored, as they would cause undesired character translation for special character sets.
	`OFF`	Output is not translated.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Error messages or warnings are translated if the English version of the text is displayed. If the text is displayed in the local language (for example, Hebrew), it is not translated into upper-

case characters. The translation of messages and warnings does not depend on the library from where the program is executed.

2. Within a Natural session, the profile parameter `TS` can be overridden by the session parameter `TS`.
3. The translation table can be modified with the `NTTABL` macro or the corresponding dynamic profile parameter `TABL`.

Support of TS=ON for Natural under IMS TM Messages

All Natural under IMS TM messages are translated into upper case if `TS=ON` is specified in the Natural session.

Support of TS=ON for RPC Server Trace

All messages in the Natural RPC server trace are translated into upper case if `TS=ON` is specified in the Natural RPC server session. The trace of the data from/to the client is not affected by `TS=ON` and remains unchanged.

Other Parameters to Provide Upper Case Translation

In addition to honoring `TS=ON`, several Natural components provide an `UCTRAN` parameter to provide for translation of messages into upper case, even if the setting of the `TS` parameter is not (or not yet) available. These components are:

- Authorized Services Manager
- Roll Server
- Global Buffer Pool Manager under z/OS and z/VSE
- Natural Com-plete/SMARTS Interface
- Natural RPC

  See *Startup Parameters* in z/OS Batch Mode and *Startup Parameters* under CICS in the *Natural RPC (Remote Procedure Call)* documentation.

For the Natural Development Server, the configuration parameter `UPPERCASE_SYSTEMMESSAGES` with similar functionality is available, for details, see *Configuring the Natural Development Server* in the *Natural Development Server* documentation.

264

TSIZE - Size of Buffer for Adabas Text Retrieval

This Natural profile parameter specifies the size of the buffer to be used for the Adabas Text Retrieval facility.

Possible settings	`1-2097151`	Buffer size in KB. **Note:** If the requested space is not available, the Adabas Text Retrieval facility cannot be used.
	`0`	Adabas Text Retrieval facility is not used.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Note: Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.

265 TSOP - Parameters for Natural TSO Interface

This Natural profile parameter is used to specify the parameters for the Natural TSO Interface. It corresponds to the macro `NTTSOP` in the Natural parameter module.

Possible settings	See ***TSOP Parameter Syntax***.	
Default setting	See ***Keyword Subparameters***.	
Dynamic specification	yes	The parameter `TSOP` can only be specified dynamically. In the Natural parameter module, use the macro `NTTSOP`.
Specification within session	no	

TSOP Parameter Syntax

The `TSOP` parameter is specified as follows:

```
TSOP=(keyword-subparameter=value,keyword-subparameter=value,...)
```

For information on subparameter names and values, see ***Keyword Subparameters***.

NTTSOP Macro Syntax

The `NTTSOP` macro is specified as follows:

```
         NTTSOP ABEXIT=value,                                         *
               ALTSCRN=value,                                         *
               LBPNAME=value,                                         *
               LEHDLR=value,                                          *
               NDBFSRV=value,                                         *
               PA2=value,                                             *
               SUBPOOL=value,                                         *
               SWAPKEY=value,                                         *
               TIOBSZ=value
```

See ***Keyword Subparameters***.

Keyword Subparameters

`ABEXIT` | `ALTSCRN` | `LBPNAME` | `LEHDLR` | `NDBFSRV` | `PA2` | `SUBPOOL` | `SWAPKEY` | `TIOBSZ`

ABEXIT - Abend Processing

`ABEXIT=`*`value`* specifies the mode of abend processing within Natural.

Value	Explanation
`ESTAE`	Natural intercepts all abends and issues the appropriate error messages. This is the default value.
`SPIE`	Only program checks (`S0C`*`x`* abends) are intercepted.
`OFF`	Natural does not intercept any abends or program checks at all.

Notes:

1. The setting `ABEXIT=OFF` corresponds to profile parameter `DU=FORCE`.
2. The setting `ABEXIT=OFF` is not recommended because some functions, which require the abend interception, will not work any longer. The usage of profile parameter `MT` will cause an abend `U0322` instead of error NAT0953 when the CPU time limit is reached.

ALTSCRN - Session Screen Mode

`ALTSCRN=`*`value`* specifies whether the 3270 alternate screen size is to be used.

Value	Explanation
`ON`	The alternate screen size is to be used. This is the default value.
`OFF`	The default screen size is to be used.

Notes:

1. There are 2 sets of screen heights/widths from the VTAM LOGMODE definition for the terminal, default and alternate screen size. Usually, the default screen size is 24 lines and 80 columns. The alternate screen size depends on the 3270 terminal model (2, 3, 4 or 5).
2. The screen size can be overwritten with Natural profile parameter `TMODEL`.

LBPNAME - Sharing of Local Buffer Pools

`LBPNAME=value` controls the sharing of the local buffer pool when running multiple Natural sessions within the same TSO region. It defines the name of the shared buffer pool environment and is used to locate the shared local buffer pool.

Value	Explanation
1 - 8 characters or ' ' (blank)	Name of the shared buffer pool.
' ' (blank)	The local buffer pools are not shared. This is the default value.

Note: When running multiple Natural sessions in a z/OS batch or TSO region concurrently, each session allocates storage for a separate local buffer pool. Except for the Natural z/OS batch mode server, the local buffer pools are not shared by default; that is, if the different sessions use the same Natural objects, these have to be loaded once for each session separately. If a shared buffer pool name is specified, all Natural sessions will share the same local buffer pool.

LEHDLR - Use of an LE Error Handler for Calling LE Subprograms

`LEHDLR=value` specifies whether Natural uses an LE error handler for the call of LE subprograms.

Value	Explanation
`ON`	An LE error handler is set up by Natural during the call of LE subprograms. This means, if an unhandled LE error occurs during the execution of an LE subprogram, Natural will get control and can handle it (by issuing error NAT0954). This is the default value.
`OFF`	No setting of an LE error handler is done by Natural during the call of LE subprograms. This means, if an unhandled error occurs during the execution of a LE subprogram, the LE enclave is terminated and so the Natural session is lost.

Notes:

1. For information on LE runtime options, see the description of source module `NATLEOPT` in the *Installation for z/OS* documentation.
2. For information on Natural running with the IBM Language Environment, refer to *Natural Execution - Miscellaneous Topics*, *LE Subprograms* in the *Operations* documentation.

NDBFSRV - Natural for DB2 File Server

`NDBFSRV=`*`value`* specifies whether the Natural for DB2 file server is to be used.

Value	Explanation
`OFF`	The Natural for DB2 file server will not be used. This is the default value.
`ON`	The Natural for DB2 file server is invoked at each terminal I/O.

PA2 - Behavior of PA2 Key

`PA2=`*`value`* specifies how the PA2 key shall work.

Value	Explanation
`ON`	The PA2 key value is passed to the Natural application.
`OFF`	The PA2 key value is used to redisplay the terminal screen and is not passed to the Natural application. This is the default value.

Note: The `PA2` subparameter specification is irrelevant if the PA2 key is defined as the split screen swap key by subparameter `SWAPKEY`.

SUBPOOL - Storage Subpool for GETMAIN Requests

`SUBPOOL=`*`value`* defines the storage subpool for `GETMAIN` requests.

Value	Explanation
`1 - 127`	Subpool number.
`0`	This is the default value.

Notes:

1. The subparameter `SUBPOOL` is honored only in the Natural parameter module which is linked to the TSO driver, but not in an alternative parameter module which is activated using a `PARM=` specification.
2. As the subparameter `SUBPOOL` is evaluated during session initialization only, it cannot be specified as a dynamic subparameter.

SWAPKEY - TSO/ISPF Split Screen Feature Support

`SWAPKEY=value` defines the TSO/ISPF split screen swap key for Natural, which is assigned to PF9 on most of the panels of Software AG product Natural ISPF (Integrated Structured Programming Facility).

Value	Explanation
PF1 - PF24	Defines the PF- or PA-key which shall be used to swap to the next TSO/ISPF session.
PA1 - PA3	
`OFF`	By default, no swap key is defined; that is, no split screen feature support is generated.

Notes:

1. The specified key cannot be used by Natural applications. Usually, the `SWAP` command in TSO/ISPF is assigned to the PF9 key on most TSO/ISPF panels.
2. Split screen support requires the TSO/ISPF interface module `ISPLINK` from the ISPF load library. You can include `ISPLNK` in the link step for `NATTSO`. If not linked, Natural tries to load it dynamically when the key defined with `SWAPKEY` is first used in the session. If `ISPLINK` is not contained in the load library, Natural treats `SWAPKEY` as set to `OFF`.

TIOBSZ - Size of the Terminal I/O Buffer

`TIOBSZ=value` specifies the size of the terminal I/O buffer.

Value	Explanation
`4 - 32`	Size of the terminal I/O buffer in KB.
`8`	This is the default setting.

Note: The terminal I/O buffer is allocated below the 16 MB line.

Example of TSOP Parameter

```
TSOP=(LBPNAME=NATTEST1,TIOBSZ=12,SWAPKEY=PF9)
```

Example of NTTSOP Macro

```
         NTTSOP LBPNAME=NATTEST1,TIOBSZ=12,SWAPKEY=PF9
```

266 TTYPE - Terminal Type

This Natural profile parameter allows you to specify the terminal type used - in TP environments in which this information is not supplied automatically - so that Natural can activate the appropriate converter routine for attribute sequences to operate that type of terminal.

Possible settings	1-4 characters	The setting specified with the `TTYPE` parameter must be defined as a valid terminal device type in the `NTDVCE` macro of the `NATCONFG` module, see *Configuration Tables - Module NATCONFG*.
Default setting	`3270`	Under z/OS, z/VSE.
	See Text.	Under BS2000: The setting defined in PDN, unless overridden by the parameter `T975X`. **Note:** See *Parameters in Macro NAMTIAM* in the *TP Monitor Interfaces* documentation.
Dynamic specification	yes	
Specification within session	yes	The `TTYPE` parameter has the same function as the terminal command `%T=`.

Note: If you use the `TTYPE` parameter, it is no longer necessary to execute a program containing a `SET CONTROL 'T=...'` statement at the start of the session in order to set the terminal type.

267 UC - Underlining Character

This session parameter determines the character that is used as underlining character for the following:

- column headings generated by `DISPLAY` statements;
- page titles/trailers produced by `WRITE TITLE`/`WRITE TRAILER` statements with `UNDERLINED` option.

Possible settings	any character	See also Note.
	`OFF`	
Default setting	-	Hyphen (-).
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`WRITE TITLE`
	`WRITE TRAILER`	

Applicable command	none	

Note: If you do not wish column headers to be underlined, you have the following options:

`UC=`	A blank line will be output instead of underlining.
`UC=OFF`	The field values will be output immediately below the heading line, without any blank line in between. You can specify `UC=OFF` only at the statement level of a `DISPLAY` statement; in this case, you cannot make any other `UC` specifications for individual fields in that statement.

Examples:

```
FORMAT UC=*
DISPLAY (UC= ) NAME AGE (UC=+)
```

Note: See also *Underlining Character for Titles and Headers - UC Parameter* in the *Programming Guide*.

268 UCONMAX - Maximum Number of Concurrent Sessions for a User

This Natural profile parameter is used to specify the maximum number of concurrent sessions for a user.

The Natural `FREG` system file is used to register the sessions started with a value > 0 for the Natural profile parameter `UCONMAX`.

Possible settings	`1 - 32767`	Maximum number of concurrent sessions for a user.
	`0`	`UCONMAX=0` indicates that no limit is to be in effect.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. If the specified limit is exceeded, the Natural session is rejected and the user is notified with a corresponding Natural error message.
2. This Natural profile parameter is in effect for servers supporting the Natural Development Server and the *Natural Web I/O Interface*.

269

UDB - User Database ID

This Natural profile parameter specifies the DBID to be used for a database access at runtime. This database ID specified with the `UDB` parameter replaces DBID 0 when Natural objects are executed.

Possible settings	`0` or `1 - 65535`, except `255`	Valid database ID. **Note:** Database ID 255 is reserved for logical system files for Software AG products, see profile parameter `LFILE`.
Default setting	database ID applicable for `FUSER`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation. * Recommended.
	`USR1040N` *	

Notes:

1. The DBID 0 and the databases selected with the `UDB` parameter must be of the same type (ADA/ADA, SQL/SQL or XML/XML, for example).
2. If no DBID is specified in the DDM used, the DBID specified with the `UDB` profile parameter determines which database is accessed. Thus, it is possible to have different user environments without multiple `FUSER` files being required.
3. If no DBID is specified in the DDM and the `UDB` profile parameter is not specified, the DBID that applies to the `FUSER` system file is used.

270 ULANG - User Language

This Natural profile parameter specifies the language to be used for date edit masks, system messages, user messages, help texts, helproutines, and multi-lingual maps. The setting is used to set the Natural system variable `*LANGUAGE`.

Possible settings	`1 - 60`	Natural language code. **Note:** 1. For example, 1 is assigned to English, 2 is assigned to German, 3 is assigned to French. 2. For a detailed list of language codes, see the table in the documentation of the `*LANGUAGE` variable.
Default setting	`1`	
Dynamic specification	yes	
Specification within session	no	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. See also the note on language code related adaptation of profile parameter `CP` when set to `ON`.
2. Within the session, the language code can be specified using the terminal command `%L=`.

See also:

- *Configuration Tables - Module NATCONFG* in the *Operations* documentation for additional information about language indicators and possible settings.
- *Screen Design*, *Skill-Sensitive User Interfaces* in the *Programming Guide*.

271 UNIIO (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

272 UPSI - Control of the User Program Switch Indicator

This Natural profile parameter adjusts the setting of the User Program Switch Indicator (UPSI) specified in the JCL. The `UPSI` profile parameter is mainly used for debugging and tracing in Natural under z/VSE to avoid undesired results if a `UPSI` setting in a JCL statement is interpreted differently by a non-Natural program.

Possible settings	1 - 8 characters	Any combination of the characters 0, 1 and X. The syntax of the character string is the same as for the `UPSI` string in the JCL control statement.
Default setting	`XXXXXXXX`	The `UPSI` string in the JCL is not adjusted to the Natural parameter setting: see ***Rules for UPSI Adjustments***.
Dynamic specification	yes	
Specification within session	no	

Rules for UPSI Adjustments:

The `UPSI` profile parameter adjusts the `UPSI` specification in the JCL control statement according to the following rules:

UPSI Parameter Setting	UPSI Specification in the JCL
`0`	The corresponding bit in the `UPSI` string is set to 0.
`1`	The corresponding bit in the `UPSI` string is set to 1.
`X`	The corresponding bit in the `UPSI` string remains unchanged.
Unspecified rightmost positions	The corresponding bits in the `UPSI` string remain unchanged.

Examples of UPSI Adjustments:

1. `UPSI` in the JCL:

```
11X0X001
```

`UPSI` parameter setting:

```
X0101
```

Resulting setting used by Natural:

```
10101001
```

2. `UPSI` in the JCL:

```
11X0X
```

`UPSI` parameter setting:

```
X0101111
```

Resulting setting used by Natural:

```
10101111
```

Related Topics:

- *Debugging Facilities for Natural under z/VSE* in the *Operations* documentation
- *Using the UPSI Parameter* in *Natural CICS Interface Debugging Facilities* in the *TP Monitor Interfaces* documentation

273 USER - Restrict Use of Profile Parameter Strings and Modules

This profile parameter can be used to restrict the use of dynamic parameter strings as specified in a `SYSPARM` profile, `NTSYS` macro or parameter data set (`CMPRMIN`), or to restrict an alternative Natural parameter module. It corresponds to the macro `NTUSER` in the parameter module.

Possible settings	See *USER Parameter Syntax*.	
Default setting	none	
Dynamic specification	yes	This parameter can only be specified dynamically. To restrict the use of an alternative Natural parameter module, the corresponding macro `NTUSER` is used instead.
Specification within session	no	

Notes:

1. The `USER` parameter applies only to the string of dynamic parameters specified *after* it.
2. When the dynamic profile parameters are evaluated and the `USER` parameter is encountered, Natural checks if the current user ID (that is, the current value of the system variable `*INIT-USER`) is contained in the list of user IDs specified with the `USER` parameter. If it is not, the user receives a corresponding error message, and the processing of dynamic profile parameters is terminated immediately.

USER Parameter Syntax

The parameter syntax of `USER` is as follows:

```
USER=(user-id,user-id,...),profile-parameter-string
```

Where:

Syntax Element	Explanation
user-id	The IDs of the users who will be allowed to use the subsequently specified string of profile parameters.
profile-parameter-string	String of profile parameters.

To restrict the use of a SYSPARM profile

- Specify the `USER` parameter as the first parameter in the profile. The subsequent string of profile parameters in the profile, that is, the entire profile, can then only be used by the user specified with the `USER` parameter.

To restrict the use of a parameter string defined in an NTSYS macro or in a CMPRMIN data set

- Specify the `USER` parameter as the first parameter in the parameter string defined in an `NTSYS` macro or in a `CMPRMIN` data set.

NTUSER Macro Syntax

The `NTUSER` macro in a Natural parameter module is specified as follows:

```
          NTUSER user-id,user-id,user-id,...,profile-parameter-string
          NTUSER user-id,user-id,...,profile-parameter-string
```

Notes:

1. For an explanation of the syntax elements, see ***USER Parameter Syntax***.
2. The `NTUSER` macro applies to the Natural parameter module in which it is specified. The default Natural parameter module linked to the environment-dependent Natural nucleus cannot be restricted.

To restrict the use of an alternative Natural parameter module

- Specify the macro `NTUSER` in the alternative parameter module.

Note: When an alternative parameter module is to be used, Natural loads the alternative parameter module specified by the `PARM` parameter and checks if the current user ID (that is, the current value of the system variable `*INIT-USER`) is contained in the list of user IDs specified by the `NTUSER` macro in the alternative parameter module. If it is not, the user receives a corresponding error message, and the alternative parameter module is discarded.

Example of USER Parameter

The following is an example of protecting a specific system file `FNAT`:

```
USER=(ADMIN1,ADMIN2),FNAT=(12,177,SECPASSW,74832055)
```

Example of NTUSER Macro

The following is an example of protecting a Natural parameter macro:

```
NTPRM ...
...
NTUSER ADMIN1,ADMIN2
```

274 USERBUF (Internal Use)

This parameter is reserved for internal use by Natural.

Caution: Do not change its setting.

275 UTAB1 - Lower-to-Upper-Case Translation

This Natural profile parameter can be used to overwrite the definitions in the translation table `NTUTAB1` as contained in the configuration module `NATCONFG`.

Possible settings	See ***UTAB1 Parameter Syntax***.	
Default setting	As specified within the macro `NTUTAB1` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTUTAB1` is used instead.
Specification within session	no	

Notes:

1. `UTAB1` corresponds to the `NTUTAB1` macro in the Natural parameter module.
2. The `NTUTAB1` table is used for lower-to-upper-case translation.
3. If the `CP` profile parameter is set to a value other than `OFF`, values specified with `UTAB1` are ignored. See also *Translation Tables* in the *Unicode and Code Page Support* documentation.

UTAB1 Parameter Syntax

The `UTAB1` parameter is specified as follows:

```
UTAB1=(a1,a2,b1,b2,c1,c2,...)
```

You specify pairs of characters, the first character of a pair being the character to be translated, the second character of a pair being the character into which the first character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the hexadecimal representation of that character.

With the `UTAB1` parameter, you must enclose the entire string of character pairs in parentheses; see ***Example of UTAB1 Parameter***.

Or:

```
UTAB1=OFF
```

With `UTAB1=OFF` all (static and dynamic) definitions are reset to the values specified in the macro `NTUTAB1` in `NATCONFG`.

NTUTAB1 Macro Syntax

The `NTUTAB1` macro is specified as follows:

```
         NTUTAB1 a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see ***UTAB1 Parameter Syntax***. For an example, see ***Example of NTUTAB1 Macro***.
2. The value `OFF` cannot be specified with the macro `NTUTAB1`, but only dynamically with the profile parameter `UTAB1`.

Example of UTAB1 Parameter

```
UTAB1=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

Example of NTUTAB1 Macro

```
         NTUTAB1 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

276 UTAB2 - Upper-to-Lower-Case Translation

This Natural profile parameter allows you to overwrite the definitions in the translation table `NTUTAB2` as contained in the configuration module `NATCONFG`.

Possible settings	See *UTAB2 Parameter Syntax*.	
Default setting	As specified within the macro `NTUTAB2` in `NATCONFG`.	
Dynamic specification	yes	This parameter can only be specified dynamically. In the Natural parameter module, the macro `NTUTAB2` is used instead.
Specification within session	no	

Notes:

1. `UTAB2` corresponds to the `NTUTAB2` macro in the Natural parameter module.
2. The `NTUTAB2` table is used for upper-to-lower case translation.
3. If the `CP` profile parameter is set to a value other than `OFF`, values specified with `UTAB2` are ignored. See also *Translation Tables* in the *Unicode and Code Page Support* documentation.

UTAB2 Parameter Syntax

The `UTAB2` parameter is specified as follows:

```
UTAB2=(a1,a2,b1,b2,c1,c2,...)
```

You specify pairs of characters, the first character of a pair being a upper-case character to be translated, the second character of a pair being the lower-case character into which the upper-case character is to be translated.

You can specify each character either as the one-byte character itself (enclosed in apostrophes) or as the two-byte hexadecimal representation of that character.

With the `UTAB2` parameter, you must enclose the entire string of character pairs in parentheses; see *Example of UTAB2 Parameter*.

Or:

UTAB2=OFF

With `UTAB2=OFF` all (static and dynamic) definitions are reset to the values specified within the macro `NTUTAB2` in `NATCONFG`.

NTUTAB2 Macro Syntax

The `NTUTAB2` macro is specified as follows:

```
         NTUTAB2 a1,a2,b1,b2,c1,c2,...
```

Notes:

1. For an explanation of the syntax elements, see ***UTAB2 Parameter Syntax***. For an example, see ***Example of NTUTAB2 Macro***.
2. The value `OFF` cannot be specified with the macro `NTUTAB2`, but only dynamically with the profile parameter `UTAB2`.

Example of UTAB2 Parameter

```
UTAB1=(5E,'Ä','ö',78,FF,00,'ü','Ü')
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

Example of NTUTAB2 Macro

```
         NTUTAB2 5E,'Ä','ö',78,FF,00,'ü','Ü'
```

In this example, the character represented by `H'5E'` is translated into `'Ä'`, `'ö'` into the character represented by `H'78'`, the character represented by `H'FF'` into the character represented by `H'00'`, and `'ü'` into `'Ü'`.

277

VSAM - Parameters for Natural for VSAM

This Natural profile parameter is used to specify the parameters for Natural for VSAM. It corresponds to the `NTVSAM` macro in the Natural parameter module, where, in addition, the macros `NTVEXIT`, `NTVLSR` and `NTVTVSD` are used.

Possible settings	See *VSAM Parameter Syntax*.	
Default setting	See default values of *keyword subparameters*.	
Dynamic specification	yes	The parameter `VSAM` can only be specified dynamically. In the Natural parameter module, use the macro `NTVSAM` and, in addition, the macros `NTVEXIT`, `NTVLSR` and `NTVTVSD`.
Specification within session	no	

VSAM Parameter Syntax

The `VSAM` parameter is specified as follows:

VSAM=(*keyword-subparameter=value,keyword-subparameter=,...*)

See *Keyword Subparameters*.

NTVSAM Macro Syntax

The `NTVSAM` macro is specified as follows:

```
         NTVSAM BTSUPP=value,                                          *
               CLSUPP=value,                                           *
               DDMCHK=value,                                           *
               DDSWITE=value,                                          *
               DFBE=value,                                             *
               DFBN=value,                                             *
               ENADIS=value,                                           *
               ENAUNE=value,                                           *
               ETSUPP=value,                                           *
               FORMAT=(value1,value2),                                 *
               KEYLGH=value,                                           *
               OPSUPP=value,                                           *
               PATH=value,                                             *
               PSIGNF=value,                                           *
               RETRY=(value1,value2),                                  *
               RLS=value,                                              *
               ROLLSIZ=value,                                          *
               SFILE=value,                                            *
               TAFE=value,                                             *
               TAFN=value,                                              *
               TIMEOUT=value,                                           *
               TSAE=value,                                              *
               TVS=value,                                               *
               UPDL=value
```

See *Keyword Subparameters*.

Note: The keyword subparameters `EXIT`, `LSR` and `TVSD` are not available in the `NTVSAM` macro. In the Natural parameter module, use the macros `NTVSAM`, `NTVEXIT`, `NTVLSR` and `NTVTVSD` instead.

NTVEXIT Macro Syntax

The `NTVEXIT` macro is specified as follows:

```
         NTVEXIT  file-name,exit-name,workarea-size
```

For details, see *EXIT - File User Exits*.

NTVLSR Macro Syntax

The `NTVLSR` macro is specified as follows:

```
         NTVLSR  file-name,subpool
```

For details, see *LSR - Local Shared Resources Subpools*.

NTVTVSD Macro Syntax

The `NTVTVSD` macro is specified as follows:

```
         NTVTVSD  file-name,option
```

For details, see *TVSD – Activate DFSMS Transactional VSAM Services*.

Keyword Subparameters

`BTSUPP` | `CLSUPP` | `DDMCHK` | `DDSWITE` | `DFBE` | `DFBN` | `ENADIS` | `ENAUNE` | `ETSUPP` | `EXIT` | `FORMAT` | `KEYLGH` | `LSR` | `OPSUPP` | `PATH` | `PSIGNF` | `RETRY` | `RLS` | `ROLLSIZ` | `SFILE` | `TAFE` | `TAFN` | `TIMEOUT` | `TSAE` | `TVS` | `TVSD` | `UPDL`

BTSUPP - Support of BACKOUT TRANSACTION Statement

`BTSUPP=`*`value`* specifies whether `BACKOUT TRANSACTION` statements are executed or not.

Value	Explanation
`ON`	Each `BACKOUT TRANSACTION` is executed and translated into an appropriate `ROLLBACK` command. This is the default value.
`OFF`	`BACKOUT TRANSACTION` statements are ignored.

Note: This subparameter is applicable only in TP and DFSMStvs environments where VSAM logging is supported.

CLSUPP - Support of CLOSE Call at Session Termination

`CLSUPP=`*`value`* specifies whether or not a `CLOSE` call is executed at session termination.

Value	Explanation
`ON`	Each `CLOSE` call is executed and translated into an appropriate `SYNCPOINT` command. This is the default value.
`OFF`	Each `CLOSE` call is ignored.

Note: If a `CLOSE` is executed, Natural for VSAM forces an `END TRANSACTION` only in TP and DFSMStvs environments where VSAM logging is supported.

DDMCHK - Support of DDM Integrity

`DDMCHK=`*`value`* checks whether the file layout and, in consequence, the DDM has changed.

Value	Explanation
`ON`	DDM check enabled.
`OFF`	DDM check disabled. This is the default value.

Note: The check is performed after each program termination at the `NEXT` level, through the Natural buffer pool. The `DDMCHK` subparameter is only relevant for development environments where DDMs are modified. In production environments, disable this feature to improve performance.

DDSWITE - Maximum Entries in DD/DLBL Name Switch Buffer

`DDSWITE=`*`value`* specifies the maximum number of entries in the DD/DLBL name switch buffer.

Value	Explanation
`0` up to `TAFE=`*`value`*	Maximum number of entries; that is `0` or a value in the range given by the maximum value specified in the `TAFE` subparameter.
`0`	This is the default value.

Note: For details on switching DD names, see the application programming interface `USR1047N` in the *SYSEXT Utility* documentation.

DFBE - Number of Decoded Format Buffer Entries

`DFBE=`*`value`* specifies the initial number of entries in the table of decoded format buffers.

Value	Explanation
`1 - 1000`	Average number of fields.
`10`	This is the default value.

Notes:

1. For each active Natural I/O statement (`FIND`, `READ`, `UPDATE`, `STORE`) one entry is allocated in this table.
2. When increasing `DFBE` or `DFBN`, take into consideration that the allocated storage area size is obtained by multiplying these values and not by adding them.

DFBN - Number of Fields in Entry of Decoded Format Buffer

`DFBN=value` specifies the average number of fields contained in an entry of the decoded format buffer table.

Value	Explanation
`1 - 1000`	Average number of fields.
`50`	This is the default value.

Notes:

1. One entry is built for each Natural I/O statement (`FIND`, `READ`, `UPDATE`, `STORE`).
2. When increasing `DFBE` or `DFBN`, take into consideration that the allocated storage area size is obtained by multiplying these values and not by adding them.

ENADIS - Enabling Disabled Files

`ENADIS=value` is used to enable disabled files.

Value	Explanation
`ON`	For all disabled files accessed during the session, an `EXEC CICS SET ENABLED` command is executed.
`OFF`	All disabled files remain disabled. This is the default value.

Notes:

1. This subparameter only applies to CICS environments and is only honored by the first file access performed in the current Natural session.
2. If this subparameter is set to `OFF` and the file has not been enabled, the NAT3516 error message must follow the first file access.

ENAUNE - Enabling Unenabled Files

`ENAUNE=value` is used to enable disabled (“unenabled”) files.

Value	Explanation
`ON`	For all “unenabled” files accessed during the session, an `EXEC CICS SET ENABLED` command is executed.
`OFF`	All “unenabled” files remain unenabled. This is the default value.

Notes:

1. This subparameter only applies to CICS environments and is only honored by the first file access performed in the current Natural session.
2. If this subparameter is set to `OFF` and the file has not been enabled, the NAT3539 error message must follow the first file access.

ETSUPP - Support of END TRANSACTION Statement

`ETSUPP=`*`value`* specifies whether `END TRANSACTION` statements are executed or not.

Value	Explanation
`ON`	Each `END TRANSACTION` is executed and translated into an appropriate `SYNCPOINT` command. This is the default value.
`OFF`	`END TRANSACTION` statements are ignored.

Note: `ETSUPP` is applicable only in TP and DFSMStvs environments where VSAM logging is supported.

EXIT - File User Exits

Natural for VSAM provides the facility to define one or more user exits. For each VSAM file to be accessed, one user exit can be defined. The definition of a user exit is done by using by the subparameter `EXIT` of profile parameter `VSAM` or the `NTVEXIT` macro in the Natural parameter module.

The subparameter `EXIT` is specified as follows:

```
EXIT=(dd-name,exit-name,workarea-size)
```

The macro `NTVEXIT` is specified as follows:

```
        NTVEXIT dd-name,exit-name,workarea-size
```

Where:

Value	Explanation
dd-name	DD/DLBL/FCT name of the VSAM file to be accessed. **Note:** There is no default value.
exit-name	Name of the user exit. **Note:** There is no default value.
workarea-size	Optionally, the size of the user exit work area (in bytes) can be specified. **Note:** 1. A minimum size of 72 bytes is required, which corresponds to the size of the IBM standard register saved area; that is, 18 full words. 2. The maximum value is 1024 bytes. 3. The default value is 72.

Note: All user exits must be either linked to the Natural parameter module or must be defined by means of CSTATIC or RCA techniques. For each file user exit, a separate definition with subparameter `EXIT` or macro `NTVEXIT` is required.

User Exit Linkage Conventions

When passing control to and from the user exit, standard IBM linkage conventions and standard linkage register notations are used.

Register	Usage
R1	Address pointer to the parameter address list. The parameter address list provides you with the addresses of the record, of LRECL, of the current function and of the work area.
R3	Address pointer to the VSAM control area (VCA).
R12	Address pointer to the Natural basic control block (BB).
R13	Address of 18-word save area.
R14	Return address.
R15	Entry address/return code. A return code of `0` indicates a normal return of control. In all other cases, a Natural error message is returned.

The current function (see Register R1 above) indicates the way control has been passed to the user exit. Control can be passed either before or after a Natural call for VSAM (see also the `DCRREQCD` field in the `NVMDCR` macro delivered):

- With the `STORE` and `UPDATE` statements, control is passed before the call.
- With the `FIND`, `GET` and `READ` statements, control is passed after the call.

Sample User Exit

A sample user exit `NVSEX01` is provided on the installation tape.

FORMAT - Support of Record Formatting for STORE and UPDATE Statements

Note: This section describes the new (extended) syntax of the `FORMAT` subparameter and, in addition, the **old syntax**, which is still supported for compatibility reasons.

New (Extended) Subparameter Syntax:

`FORMAT=(`*`value1`*`,`*`value2`*`)` supports the formatting of VSAM records referenced in a `STORE` or `UPDATE` statement. Record fields that are not referenced, and therefore contain binary zeros, are converted into a format that corresponds to the field type and record length defined in the relevant DDM.

Value		Explanation
value1	`ON`	VSAM records are formatted in accordance with the corresponding DDM definitions. This is the default.
	`OFF`	VSAM records are not formatted and fields that are not referenced contain binary zeros.
value2	`KEYS`	All VSAM keys are formatted in accordance with the DDM field type. This is the default.
	`NOKEYS`	All VSAM keys are not formatted.

Note: Natural for VSAM system file records are always formatted; this cannot be changed.

The following value combinations are reasonable:

`FORMAT=(ON,NOKEYS)`	All fields are formatted, excepting the keys.
`FORMAT=(OFF,KEYS)`	Only the keys are formatted.

Old Subparameter Syntax:

The old subparameter syntax is still supported for compatibility reasons.

`FORMAT=`*`value`* supports the formatting of VSAM records referenced in a `STORE` or `UPDATE` statement.

Value	Explanation
`ON`	VSAM records are formatted in accordance with the corresponding DDM definitions. This is the default value.
`OFF`	VSAM records are not formatted and fields that are not referenced contain binary zeros.

Notes:

1. Record fields that are not referenced, and therefore contain binary zeros, are converted into a format that corresponds to the field type and record length defined in the relevant DDM.
2. Natural for VSAM system file records are always formatted; this cannot be changed.

KEYLGH - Length of VSAM Keys used in I/O Statements

`KEYLGH=`*`value`* specifies the length of VSAM keys used in Natural I/O statements.

Value	Explanation
`1 - 255`	Length of VSAM keys (in bytes) used in Natural I/O statements.
`32`	This is the default value.

Notes:

1. The maximum key length for a VSAM file is 255 bytes.
2. The value of this subparameter is used to calculate the size of the TSA table (Table of Sequential Access).
3. If you use VSAM system files, specify at least: 87 bytes for the `FNAT`, `FUSER`, `FDIC` and `FSPOOL` files, and 126 bytes for the `FSEC` and Natural ISPF system files.

LSR - Local Shared Resources Subpools

This subparameter is only required if VSAM files are used as local shared resources.

Note: This method results in a substantial increase of the performance of TSO and batch runs, and, at the same time, decrease the VSAM I/O rate. The definition of the usage of a local shared resources subpool per file is done by using by the subparameter `LSR` of profile parameter `VSAM` or the `NTVLSR` macro in the Natural parameter module.

The subparameter `LSR` is specified as follows:

`LSR=(dd-name,subpool-number)`

The macro `NTVLSR` is specified as follows:

```
        NTVLSR dd-name,subpool-number
```

Value	Explanation
dd-name	DD/DLBL/FCT name of the VSAM file to be accessed. There is no default value.
subpool-number	Subpool number (ID) between `0` and `15` for z/VSE or between `0` and `255` for z/OS; see also the relevant IBM VSAM documentation. There is no default value.

Up to 200 logical files are possible. For each file a separate definition with subparameter `LSR` or macro `NTVLSR` is required.

If `ERROR=YES` is set in `NVSMISC`, all files defined with subparameter `LSR` or macro `NTVLSR` must be defined via JCL at runtime; otherwise, an appropriate Natural initialization error message is returned.

If you have defined base clusters with subparameter `LSR` or macro `NTVLSR` which contain path entries, all paths must also be defined with subparameter `LSR` or macro `NTVLSR`.

For non-path environments the following applies: If the upgrade option is active in the VSAM catalog and if a VSAM file is defined with subparameter `LSR` or macro `NTVLSR` and contains references to an alternate index (AIX), all AIX files must also be defined with subparameter `LSR` or macro `NTVLSR`.

Natural for VSAM automatically calculates the optimum pool size by using the corresponding VSAM catalog information on the files involved, and then creates separate subpools for data and index components.

In batch mode under z/OS, Natural for VSAM allocates the pools as ESO Hiperspace if the following conditions are met:

- All sizes in the VSAM catalog are at least specified as 4 KB or a multiple of this value (this is valid for both data and index components).
- The library from which Natural for VSAM was loaded is an APF-authorized library.

  This condition is necessary to define the address space as *non-swappable*, which is a prerequisite for ESO Hiperspaces.

OPSUPP - Support of Dynamic Open Calls

`OPSUPP=value` enables or disables the support of multiple different open calls within one session.

Value	Explanation
`ON`	Multiple different open calls are supported by calling the application programming interface `USR2008N`.
`OFF`	Multiple different open calls are not supported within one session. This is the default value.

Note: For further information on application programming interfaces, see the *SYSEXT Utility* documentation.

PATH - Support of Path Processing

`PATH=value` is used to handle a secondary key as a path or as a native AIX file.

Value	Explanation
`ON`	All secondary keys defined in a DDM are handled as paths for AIX files.
`OFF`	All secondary keys are handled as AIX files.
`CHECK`	Natural for VSAM checks whether the secondary keys are defined as paths or as AIXs in the VSAM catalog. This is the default value.

Notes:

1. If you use the VSAM system files `FSEC` and/or `FSPOOL`, you must not specify `PATH=ON`; specify either `PATH=OFF` or `PATH=CHECK`.
2. If `PATH=CHECK` is set under CICS and/or Com-plete in a z/VSE environment, the startup JCL job must contain the corresponding DLBL card(s).

PSIGNF - Support of Compiler Option PSIGNF

`PSIGNF=value` is used to handle the internal representation of positive signs of packed numbers.

Value	Explanation
`ON`	Natural for VSAM supports the compiler option `PSIGN` for a Natural object, the corresponding DDM description in the field `ZONES` is ignored.
`OFF`	Natural for VSAM uses the DDM description in field `ZONES`. This is the default value.

RETRY - Support of RETRY Statement for an ON ERROR Clause

`RETRY=(`*`value1`*`,`*`value2`*`)` is used to support the `RETRY` statement for the following Natural for VSAM error messages:

- `NAT3541     File :1:, control interval/record held by another user`
- `NAT3520     Held VSAM record modified by another user`

Where:

Value	Explanation
`(`*`value1`*`,`*`value2`*`)`	*`value1`* applies to NAT3541, *`value2`* applies to NAT3520. Each value can be either `ON` or `OFF`.
`(OFF,OFF)`	This is the default value.

RLS - Support of Record-Level Sharing

`RLS=`*`value`* is used to enable, disable or check for the support VSAM record-level sharing (RLS) under z/OS, DFSMS Version 1.6 or higher.

Value	Explanation
`ON`	All files are opened in RLS mode.
`OFF`	All files are opened in non-RLS mode (NSR, LSR). This is the default value.
`CHECK`	All files are checked whether they are defined as SMS-managed data sets with RLS options; if they are, the file is opened in RLS mode, if not in non-RLS mode.

Notes:

1. This subparameter applies to z/OS only.
2. If `TVS=ON` is set (see subparameter `TVS` below) and no VSAM file has been defined in the `NTVTVSD` macro (see above), set `RLS=CHECK` to verify that the corresponding VSAM file has been defined as recoverable data set.

ROLLSIZ - Size of Area for Session Status Information

`ROLLSIZ=`*`value`* specifies the size of the area used by Natural to save internal session status information when a Natural transaction is terminated due to the end of a TP-monitor task.

Value	Explanation
`0` or `1 - 10000`	Size of the area in bytes.
`550`	This is the default value.

Note: This subparameter is applicable in a thread environment only (CICS, Com-plete, Natural as a Server).

SFILE - Support of VSAM System Files

`SFILE=`*`value`* is used to enable, disable or check for the support VSAM system files.

Value	Explanation
`ON`	Support of VSAM system files.
`OFF`	No support of VSAM system files. This is the default value.
`CHECK`	Checks whether the Natural system files `FNAT`, `FUSER` and `FDIC` are defined as Natural for VSAM Version 8.2 VSAM system files with the required key length of 87.

Note: If `SFILE=CHECK` is set under CICS and/or Com-plete in a z/VSE environment, the startup JCL job must contain the corresponding DLBL card(s).

TAFE - Maximum Number of DDMs per Natural Session

`TAFE=`*`value`* specifies the maximum number of DDMs per Natural session.

Value	Explanation
`0` or `1 - 1000`	Maximum number of DDMs.
`10`	This is the default value.

Notes:

1. Since it is possible to define several descriptors in one DDM, the `TAFE` subparameter has impact on the sizes of the FCT, FWA, OPV and TAF buffers (see *Buffers for Memory Management*) in the *Natural for VSAM* documentation.
2. When increasing `TAFE` or `TAFN`, take into consideration that the allocated storage area size is obtained by multiplying these values and not by adding them.

TAFN - Average Number of DDM Fields

`TAFN=value` specifies the average number of DDM fields contained in each entry in the table of accessed VSAM files.

Value	Explanation
`0` or `1 - 1000`	Maximum number of DDM fields.
`50`	This is the default value.

Note: When increasing `TAFE` or `TAFN`, take into consideration that the allocated storage area size is obtained by multiplying these values and not by adding them.

TIMEOUT - Timeout in Seconds for an RLS Request

`TIMEOUT=value` is used to support an RLS/non-RLS-file mixed environment under z/OS CICS Version 5.3 or higher in a Natural for VSAM session.

Value	Explanation
`0` or `1 - 30`	Timeout period in seconds.
`0`	This is the default value.

Notes:

1. This subparameter only applies to z/OS CICS Version 5.3 or higher.
2. Natural and Natural for VSAM Version 6.2 are plex-enabled; that is, after a terminal I/O the Natural session can be continued by the workload manager on a different z/OS in a different CICS 5.3, provided the resources are plex-enabled. Since this is not the case with non-RLS files, the session must be run in conversational mode as soon as a VSAM file is opened in non-RLS mode. With the `TIMEOUT` subparameter, you can determine that non-RLS files are to be deleted from the Natural for VSAM FCT queue. When there are no further non-RLS FCT entries for the particular Natural for VSAM session, Natural for VSAM switches to non-conversational mode, which means that z/OS Parallel Sysplex processing is possible again.

TSAE - Maximum Number of Nested READ and FIND Statements

`TSAE=value` is used to set the maximum number of all nested `READ` and `FIND` statements.

Value	Explanation
`0` or `1 - 100`	Maximum number of all nested `READ` and `FIND` statements.
`10`	This is the default value.

TVS - Support of DFSMStvs

`TVS=value` is used to support DFSMS Transactional VSAM Services (DFSMStvs).

Value	Explanation
`ON`	Support of DFSMStvs.
`OFF`	No support of DFSMStvs. This is the default value.

Notes:

1. This subparameter applies to z/OS only.
2. If `TVS` is set to `ON`, the subparameters `BTSUPP` and `ETSUPP` are forced to `ON`. The subparameter `RLS` is only forced to `ON` if `RLS` has been set to `OFF` (`RLS=CHECK` is not forced to `ON`).

TVSD – Activate DFSMS Transactional VSAM Services

DFSMS Transactional VSAM Services (DFSMStvs) is activated by setting either the ACB parameter `RLSREAD` or the JCL parameter `RLS`. In general, Natural for VSAM opens all VSAM files for output by default.

This subparameter activates DFSMStvs by specifying the read integrity value of the ACB parameter `RLSREAD`. If specifying `RSLREAD` in subparameter `TVSD` or macro `NTVTVSD`, you do not have to adapt the JCL to activate DFSMStvs.

If you only set `VSAM` subparameter `TVS=ON` without specifying the corresponding VSAM file definition with `TVSD` or macro `NTVTVSD`, to activate DFSMStvs, you need to modify the JCL as described below. In this case, you must specify `VSAM` subparameter `RLS=CHECK`.

The subparameter `TVSD` is specified as follows:

```
TVSD=(dd-name,option)
```

The macro `NTVTVSD` is specified as follows:

```
NTVTVSD dd-name,option
```

Where:

Value	Explanation
`dd-name`	DD/DLBL/FCT name of the VSAM file to be accessed. There is no default value.
`option`	`NRI` - No read integrity (dirty read). `CR` - Consistent read. `CRE` - Consistent read explicit. There is no default value.

UPDL - Size of Update Table

`UPDL=`*`value`* specifies the size of the table used by the Natural interface to VSAM to save the fields of records read for subsequent updating.

Value	Explanation
`0` or `1 - 500000`	Size of table in bytes.
`8192`	Or `32768` if `SFILE=ON` is set. This is the default value.

Note: Because these records are not read with hold by Natural to avoid deadlock conditions, the content of the `UPDL` table is used to check if any changes have been made before the update request by another user.

Examples of NTVSAM Macro

```
NTVSAM  RLS=ON,PATH=ON,KEYLGH=66
NTVEXIT FILE1,EXIT1,400
NTVEXIT FILE2,EXIT2
```

Examples of VSAM Parameter

```
VSAM=(RLS=ON,PATH=ON,KEYLGH=66,EXIT=(FILE1,EXIT1,400),EXIT=(FILE2,EXIT2))
```

PARM - Alternative Natural Parameter Module for VSAM

If you want to use an alternative Natural parameter module for Natural for VSAM, specify the name of this module with the `PARM` profile parameter and link the Natural I/O module for VSAM to this module. In addition, link all Natural for VSAM user exits defined in the `NTVEXIT` macro to this Natural parameter module.

278

VSEP - Parameters for z/VSE Batch

This Natural profile parameter is used to specify the parameters for z/VSE batch. It corresponds to the `NTVSEP` macro in the Natural parameter module.

Possible settings	See *VSEP Parameter Syntax*.	
Default settings	See default values of *keyword subparameters*.	
Dynamic specification	yes	
Specification within session	no	

VSEP Parameter Syntax

The `VSEP` parameter is specified as follows:

VSEP=(*keyword-subparameter*=*value*,*keyword-subparameter*=*value*,...)

The `VSEP` parameter covers a subset of the subparameters which are available with the `NTVSEP` macro. The following keyword subparameters are available with `VSEP`:

`CANCEL` | `FILMNGR` | `FILSCAN` | `FLUSH` | `MAXABND` | `RCSIZE` | `RJEUSER` | `SEGMENT` | `TIOBSZ` | `USERID`

For subparameter descriptions, see *Keyword Subparameters*.

NTVSEP Macro Syntax

The `NTVSEP` macro is specified as follows:

```
         NTVSEP CANCEL=value,                                          *
               FILEID=value,                                           *
               FILMNGR=value,                                          *
               FILSCAN=value,                                          *
               FLUSH=value,                                            *
               LIBRID=value,                                           *
               MAXABND=value,                                          *
               RCSIZE=value,                                           *
               RJEUSER=value,                                          *
               SEGMENT=value,                                          *
               SPOOLID=value,                                          *
               TIOBSZ=value,                                           *
               USERID=value,                                           *
               WAITIME=value
```

For subparameter descriptions, see *Keyword Subparameters*.

Keyword Subparameters

`CANCEL` | `FILEID` | `FILMNGR` | `FILSCAN` | `FLUSH` | `LIBRID` | `MAXABND` | `RCSIZE` | `RJEUSER` | `SEGMENT` | `SPOOLID` | `TIOBSZ` | `USERID` | `WAITIME`

Note: The `VSEP` parameter covers only a subset of the subparameters which are available with the `NTVSEP` macro. For details, see ***VSEP Parameter Syntax***.

CANCEL - Cancel Action for Natural Job at Session Termination

`CANCEL=`*`value`* specifies how the Natural z/VSE interface proceeds at session termination.

Value	Explanation
`ON`	The Natural z/VSE interface cancels the Natural batch job if an error has occurred during the session. The job is not cancelled in the case of a NAT9995 termination message or when a user terminates the session with a Natural `TERMINATE` statement.
`OFF`	The Natural z/VSE interface always terminates with a return code. This is the default value.

FILEID - String to Ignore VSE Label Information

`FILEID=`*`value`* specifies a string of characters which is checked against the start of a `DLBL` or `TLBL` file ID.

Value	Explanation
String of up to 8 characters.	Any character string, which must be enclosed in apostrophes if it contains special characters.
`IGNORE`	This is the default value.

Notes:

1. This subparameter can be specified only in the `NTVSEP` macro. It cannot be specified dynamically using the profile parameter `VSEP`.
2. If the string matches, the label information is ignored. This is particularly helpful when `DLBL` or `TLBL` statements for `CMWKF`*nn** and/or `CMPRT`*nn** are supplied in the (partition) standard labels, but should not be used.
3. If, for example, a `// DLBL CMPRT01,'...'` statement is found, it is not possible to direct a `WRITE(1)` output to a printer `SPOOL`. To do so, use the JCS statement `// DLBL CMPRT01,'IGNORE'` and a suitable printer assignment of the relevant `SYS`*nnn*.

FILMNGR - Management of Print or Work File in Natural

`FILMNGR=value` specifies how a print or a work file is to be managed in Natural.

Value	Explanation
`ON`	The fact that there is label information for a print or a work file and the fact that `LABEL=ON` or `LABEL=OFF` is specified for an unlabeled work file indicates to Natural that this file is available. In particular, this is relevant if the Natural print and work files are to be managed by a file management system.
`OFF`	The logical unit number of the Natural print or work file must be assigned to the appropriate device type. This is the default value.

FILSCAN - Scanning of Print or Work Files

`FILSCAN=value` specifies whether print or work files are to be scanned.

Value	Explanation
`ON`	The Natural z/VSE interface scans the z/VSE label area for all Natural print and work files for which no specific file access method has been defined via Natural profile parameters, as this may cause overhead. This is the default value.
`OFF`	Access to all Natural print and work files must be specified explicitly via Natural profile parameters in order to be “available”. This concentrates all file access efforts on the defined files.

FLUSH - Flush Card Input Files until EOF

`FLUSH=value` specifies how the Natural z/VSE interface is to proceed at session termination with the `CMSYNIN/CMOBJIN` card input files.

Value	Explanation
`ON`	At session termination, the Natural z/VSE interface will read the `SYSIN/SYSRDR/SYSIPT` card input file until `EOF`, unless `EOF` had been encountered by Natural; this means that when driving the batch mode Natural session completely with `STACK` data, an extra “/*” has to be provided in the JCL for a `CMSYNIN/CMOBJIN` null file. This is the default value.
`OFF`	No extra `SYSIN/SYSRDR/SYSIPT` card input (null) file is required, if the batch mode Natural session is completely driven with `STACK` data; the `SYSIN/SYSRDR/SYSIPT` card input file is then left as is, thus potentially resulting in `INVALID STATEMENT` operator prompts or job cancellation due to `INVALID STATEMENT`, when the Natural `CMSYNIN/CMOBJIN` had not been retrieved completely.

LIBRID - String to Trigger z/VSE Library Access

`LIBRID=`*`value`* specifies a string of up to 8 characters which is checked against the start of a DLBL file ID. If it matches, the remaining portion of that file ID is scanned for information specifying a library member in a z/VSE library or library chain.

Value	Explanation
String of up to 8 characters.	Any character string, which must be enclosed in apostrophes if it contains special characters.
`LIBR:`	This is the default value.

Note: This subparameter can be specified only in the `NTVSEP` macro. It cannot be specified dynamically using the profile parameter `VSEP`.

MAXABND - Maximum Number of Abends

`MAXABND=`*`value`* specifies the maximum number of abends which `NATVSE` tolerates (that is, `NATVSE` intercepts the abend and invokes the Natural abend handler) until it assumes an unrecoverable abend situation or abend loop and terminates the Natural session abnormally by itself.

Value	Explanation
Numeric.	Maximum number of abends.
`16`	This is the default value.

RCSIZE - Default Roll Cache Size for a Server Environment

`RCSIZE=`*`value`* specifies the default roll cache size for a server environment for the case that the roll cache size is *not* passed with the Initialize Environment request.

Value	Explanation
Numeric.	Default roll cache size in KB.
`0`	This is the default value.

Note: This subparameter can be specified only in the `NTVSEP` macro. It cannot be specified dynamically using the profile parameter `VSEP`.

RJEUSER - User ID for Submission via XPCC Macro Requests

`RJEUSER=value` defines which user ID is to be set for submission via `XPCC` macro requests.

Value	Explanation

`ON`
or
`(ON,VSE)`
	The system variable `*INIT-USER` is used as the mandatory submission user ID.
	`RJEUSER=ON` is the default value.

`(ON,NAT)`	The system variable `*USER` is used as the mandatory submission user ID.
`OFF`	The user ID `R000` is used.

SEGMENT - Behavior at Output Spool File Close

`SEGMENT=value` specifies how the Natural z/VSE interface is to behave at `CLOSE` of an output `SPOOL` file (print or punch).

Value	Explanation
`ON`	A file close is accompanied by a `POWER` segment close unless `CLOSE=FIN` is in effect for that file.
`OFF`	The `SPOOL` file is closed without closing the `POWER` segment. This is the default value.

SPOOLID - String to Trigger Direct POWER SPOOL Access

`SPOOLID=value` specifies a string which is checked against the start of a DLBL file ID or the file ID of a `DEFINE PRINTER` or `DEFINE WORK FILE` statement.

Value	Explanation
String of up to 8 characters.	Any character string, which must be enclosed in apostrophes if it contains special characters. If it matches, the output will be directly spooled to POWER using POWER SPOOL access services.
`SPL2PWR`	This is the default value.

Note: This subparameter can be specified only in the `NTVSEP` macro. It cannot be specified dynamically using the profile parameter `VSEP`.

TIOBSZ - Size of Natural I/O Buffer

`TIOBSZ=`*`value`* specifies the size of the Natural I/O buffer which is used for all input and output operations.

Value	Explanation
Minimum: `8`	Size of the Natural I/O buffer in KB.
`8`	This is the default value.

USERID - Content of System Variable *INIT-USER

`USERID=`*`value`* specifies the content of the system variable `*INIT-USER`.

Value	Explanation
`ON`	The following logic applies: ■ if a z/VSE user ID is specified in JCL (`// ID USER=`*`xxx`*), this user ID is taken; ■ otherwise, if a `POWER from-user` is specified in JCL (`* $$ JOB FROM=`*`xxx`*), this user ID is taken; ■ otherwise, the VSE job name is taken for the Natural user ID.
`OFF`	The VSE job name is taken for the Natural user ID. This is the default value.

WAITIME - Time Limit for Session Roll-Out

`WAITIME=`*`value`* specifies a time limit for session roll-out.

Value	Explanation
Numeric.	Time limit in milliseconds.
`1000`	This is the default value.

Notes:

1. This subparameter applies to the `CMROLL` call in a Natural server environment: if the time interval passed in the `CMROLL` call is not less than the `WAITIME` interval, the session is rolled-out and the its thread is released, while the session is waiting.
2. This subparameter can be specified only in the `NTVSEP` macro. It cannot be specified dynamically using the profile parameter `VSEP`.

Example of VSEP Parameter

```
VSEP=(FILMNGR=OFF,FILSCAN=OFF,FLUSH=ON,RJEUSER=(ON,NAT),SEGMENT=ON,USERID=ON)
```

Example of NTVSEP Macro

```
         NTVSEP FLUSH=ON,                                                 *
               FILEID=IGNORE,                                             *
               FILMNGR=OFF,                                               *
               FILSCAN=ON,                                                *
               LIBRID='LIBR:',                                            *
               MAXABND=16,                                                *
               RCSIZE=0,                                                  *
               RCSIZE=0,                                                  *
               RJEUSER=ON,                                                *
               SEGMENT=OFF,                                               *
               SPOOLID='SPL2PWR',                                         *
               TIOBSZ=8,                                                  *
               USERID=OFF,                                                *
               WAITIME=1000
```

279 VSIZE - Size of Buffer Area for Natural for VSAM

This Natural profile parameter sets the maximum size of the buffer area required by Natural for VSAM. If set to `0` or if the requested space is not available, Natural for VSAM cannot be used.

Possible settings	`1 - 512`	Buffer size in KB.
	`0`	With `VSIZE=0`, Natural for VSAM cannot be used.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter applies only if Natural for VSAM is installed.
2. If Natural for VSAM is installed, the corresponding Natural buffers are requested at the initialization of the Natural session.
3. If the requested space is not available, Natural for VSAM cannot be used. An appropriate error message at the initialization of Natural for VSAM tells you which buffer specified in the `NTVSAM` macro in the Natural parameter module does not fit into the `VSIZE` area; you can then increase the size of the `VSIZE` area.
4. If you do not need VSAM support during a Natural session, it is recommended that you invoke Natural with `VSIZE=0` to avoid overhead caused by handling of unused buffers.

280 WEBIO - Web I/O Interface Screen Rendering

This Natural profile parameter can be used to individually enable or disable the rendering of certain features of the Natural Web I/O Interface display on the basis of a style sheet. It corresponds to the `NTWEBIO` macro in the Natural parameter module.

Possible settings	See *WEBIO Parameter Syntax*.	
Default setting	See default values of **keyword subparameters**.	
Dynamic specification	yes	The parameter `WEBIO` can only be specified dynamically. In the Natural parameter module, use the macro `NTWEBIO`.
Specification within session	no	

Notes:

1. By default, the style sheet based rendering of the message line, PF key buttons and Natural window objects is disabled.
2. For further information, see the corresponding sections in *Using Style Sheets* in the *Natural Web I/O Interface* documentation.

The following topics are covered below:

WEBIO Parameter Syntax

The `WEBIO` parameter is specified as follows:

WEBIO=(*keyword-subparameter*=*value*,*keyword-subparameter*=*value*,...)

For information on subparameter names and values, see *Keyword Subparameters*.

NTWEBIO Macro Syntax

The `NTWEBIO` macro is specified as follows:

```
         NTWEBIO ML=value,                                      *
               KEYS=value,                                      *
               WIN=value
```

See *Keyword Subparameters*.

Keyword Subparameters

`ML` | `KEYS` | `WIN`

ML - Message Line

`ML=`*`value`* enables/disables the style sheet based rendering of the message line.

Value	Explanation
`ON`	The style sheet based rendering of the message line is enabled.
`OFF`	The style sheet based rendering of the message line is disabled.

Note: See also *Modifying the Message Line* in the *Natural Web I/O Interface* documentation.

KEYS - PF Keys

`KEYS=`*`value`* enables/disables the style sheet based rendering of the PF key buttons.

Value	Explanation
`ON`	The style sheet based rendering of the PF key buttons is enabled.
`OFF`	The style sheet based rendering of the PF key buttons is disabled.

Note: See also *Modifying the Style of the PF Key Buttons* in the *Natural Web I/O Interface* documentation.

WIN - Window Objects

`WIN=`*`value`* enables/disables the style sheet based rendering of Natural window objects.

Value	Explanation
`ON`	The style sheet based rendering of Natural window objects is enabled.
`OFF`	The style sheet based rendering of Natural window objects is disabled.

Note: See also *Modifying the Natural Windows* in the *Natural Web I/O Interface* documentation.

Example of WEBIO Parameter

```
WEBIO=(KEYS=ON,ML=ON)
```

Example of NTWEBIO Macro

```
         NTWEBIO KEYS=ON,ML=ON
```

281 WH - Wait for Record in Hold Status

This Natural profile and session parameter specifies the action to be taken if a required record is not available for processing, because it has been placed in hold status by another user.

Possible settings	`ON`	The user is placed in wait status until either the requested record becomes available, or an error message is issued due to Adabas exceeding a time limit or other limit while attempting to place the record in hold status.
	`OFF`	An error message is returned if any of these records cannot be placed in hold status.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. This Natural profile and session parameter applies to Adabas databases only.
2. Within a Natural session, the profile parameter `WH` can be overridden by the session parameter `WH`.
3. When a Natural statement is executed which results in Adabas records being read and an update/delete operation could follow, Natural requests that Adabas places these records in hold status. See the Adabas *Command Reference* documentation for further information on hold processing.
4. Under Natural Security, the setting of this parameter can be overridden by the Session Parameters option of the *Library Profile*.

5. For a `READ` or `FIND` statement using the `SKIP RECORDS IN HOLD` option, database access is always executed as if `WH=OFF` is set. If a user attempts to read a record that was placed into hold by another user, this record is skipped and processing continues with the next record in the read sequence. An error message is not returned in this case.

282 WORK - Work-File Assignments

This Natural profile parameter specifies the maximum number of work files to be used during the session. It corresponds to the `NTWORK` macro in the Natural parameter module.

Possible settings	See ***WORK Parameter Syntax***.	
Default setting	See default values of the keyword subparameters described below. Depending on the access method and the environment, there may be different default settings.	
Dynamic specification	yes	The parameter `WORK` can only be specified dynamically. In the Natural parameter module, the macro `NTWORK` must be used.
Specification within session	no	

Notes:

1. Within a session, up to 32 logical work files (numbered 1 to 32) can be used.
2. To provide different work file definitions, `WORK` or `NTWORK` can be specified multiple times.
3. The software components for accessing work files in different environments are called access methods. For the duration of a Natural session, each logical work file can be assigned to one access method only. The access method for a work file is determined by the keyword subparameter `AM`.
4. In z/OS under TSO and in batch mode, work files need not be predefined in the JCL. Provided they are defined by subparameter `AM=STD`, they can be allocated dynamically during the session by a Natural program using the `DEFINE WORK FILE` statement or the application programming interface `USR2021` (in library `SYSEXT`).
5. See also *Print and Work File Handling with External Data Sets in a Server Environment* in the *Operations* documentation.

This document covers the following topics:

WORK Parameter Syntax

With the `WORK` parameter, you first specify one or more logical work file numbers and then several keyword subparameters, which define the characteristics for these work files:

WORK=((*work-file-numbers*),*keyword-subparameters*,...)

Where:

Syntax Element	Description
work-file-numbers	The file numbers must be specified first and enclosed in parentheses: - The numbers can be from 1 to 32. - They can be specified in any sequence. - Multiple numbers must be separated from one another by commas or blanks. - To specify a range of numbers, you can use a hyphen (-).
keyword-subparameters	The keyword subparameters (for the different environments) are described below. If any previous definition (or default) for the same work file exists, only the values for the specified keyword subparameters are overwritten, all other values remain unchanged.

Note: To provide different work file definitions, `WORK` can be specified multiple times.

Examples:

```
WORK=((2,12,18),AM=STD,DEST='WORK**')
WORK=((1,3,6-11,15),AM=COMP,OPEN=INITOBJ,CLOSE=CMD)
```

NTWORK Macro Syntax

With an `NTWORK` macro, you first specify one or more logical work file numbers, and then several keyword subparameters, which define the characteristics for these work files:

```
         NTWORK (work-file-numbers),keyword-subparameters,...
```

Where:

Syntax Element	Description
work-file-numbers	The file numbers must be specified first and enclosed in parentheses: - The numbers can be from 1 to 32. - They can be specified in any sequence. - Multiple numbers must be separated from one another by commas or blanks. - To specify a range of numbers, you can use a hyphen (-).

Syntax Element	Description
keyword-subparameters	The keyword subparameters (for the different environments) are described below. If any previous definition (or default) for the same work file exists, only the values for the specified keyword subparameters are overwritten, all other values remain unchanged.

Note: To provide different work file definitions, `NTWORK` can be specified multiple times.

Examples:

```
        NTWORK (2,12,18),AM=STD,DEST='WORK**'
        NTWORK (1,3,6-11,15),AM=COMP,OPEN=INITOBJ,CLOSE=CMD
```

Keyword Subparameters for All Environments

The following keyword subparameters are available for all environments:

`AM` | `CLOSE` | `DEST` | `LRECL` | `OPEN` | `PAD` | `PADCHRI` | `PADCHRO` | `TRUNC`

AM - Type of Access Method

`AM=`*`value`* specifies the type of access method to be used.

Value	Access Method
`STD`	Standard sequential files (batch, TSO, TIAM).
`COMP`	Com-plete work files.
`SMARTS`	SMARTS work files. Work file on a SMARTS Portable File System (PFS).
`CICS`	CICS transient data or temporary storage.
`PC`	Entire Connection.
`USER`	Third-party vendor work-file interface.
`OFF`	Unassigned. No automatic assignments if `FAMSTD=OFF` is set. **Note:** `WORK=OFF` is equivalent to: `WORK=((1-32)),AM=OFF)`. It does not affect any of the other keyword subparameter specifications.
`0`	Unassigned. Automatic assignments if `FAMSTD=OFF` is set. This is the default value.

Notes:

1. For an online session, all work files to be used have to be assigned to a specific access method.
2. For a batch session, any work files not assigned to a specific access method will be automatically detected and assigned by the standard batch access method (`AM=STD`), provided that they have been predefined in the JCL. See also *FAMSTD - Overwriting of Print and Work File Access Method Assignments*.

CLOSE - Time of File Closure

`CLOSE=`*`value`* specifies when the file is to be closed:

Value	Explanation: The file is closed ...
`OBJ`	when processing of the object in which it was first accessed is finished, or when command mode, `NEXT` mode or `MAINMENU` is reached.
`CMD`	when command mode, `NEXT` mode or `MAINMENU` is reached. This is the default value.
`FIN`	at session end. **Note:** With `CLOSE=FIN`, a `DEFINE WORK FILE` statement causes an error if the work file was opened already. A `CLOSE WORK FILE` statement for the work file is ignored. When the end-of-file condition occurs during the `READ WORK FILE` statement, Natural closes the work file immediately.
`USER`	This value specifies that a work file is closed only if the file is open and one of the following conditions is true: - a `CLOSE WORK FILE` statement is issued, - a `DEFINE WORK FILE` statement is issued, - session termination.

DEST - External Data Set Name

`DEST=`*`value`* specifies the external data set name.

Value	Explanation
1 - 8 characters or 1 - 7 characters, depending on access method and environment.	Name of the external data set. **Note:** The `DEST` subparameter corresponds to *`operand1`* of the `DEFINE WORK FILE` statement, and the subparameter value can be overwritten by a `DEFINE WORK FILE` specification.

The meaning of the subparameter `DEST` depends on the access method specified with the `AM` subparameter:

Access Method	Meaning of Keyword Subparameter DEST
`AM=STD`	`DEST` is the logical data set name (`DDNAME`, `LINK` name, DTF name). **Note:** 1. If the destination is to be for multiple files, two asterisks (**) have to be specified for the file number. These will be replaced by the corresponding logical file number for each work file. A `DEST` value including two asterisks must be enclosed in apostrophes ('), when it is used as a dynamic parameter. 2. The default value is `DEST='CMWKF**'` for z/OS, z/VSE, and `DEST='W**'` for BS2000 environments. 3. Under z/VSE, only 7-character names are supported.
`AM=CICS`	There is no default value for work files under CICS. Here, the `DEST` subparameter is mandatory; that is, CICS work files defined without a valid `DEST` specification are ignored. **Note:** The Natural CICS interface also supports a variable (see the `TERMVAR=&TID` default parameter setting in the `NTCICSP` macro as part of the `DEST` value which, when being specified, is replaced by the actual CICS terminal ID; see also *Natural Print and Work Files under CICS* in the *TP Monitor Interfaces* documentation).
`AM=COMP`	`DEST` defines the name of the Com-plete SD-file. The length is restricted to a maximum of 8 characters. **Note:** 1. If the file is defined with `TYPE=TID`, the `DEST` value is appended by the Com-plete stack level. The length is restricted accordingly to a maximum of 7 characters. 2. SD-file names starting with `'&&'` are treated as temporary files which are deleted automatically after Natural termination.

LRECL - Default and Maximum Record Length of Data Set

`LRECL=`*`value`* specifies the record length of the data set.

Value	Explanation

`0`
or
`5 - 32767`
	Record length of the data set (in bytes).

`0`	This is the default value.

Note: This subparameter is used particularly to check for truncation and padding. For more information on `AM=STD`, see the keyword subparameter `LRECL` in the section ***Keyword Subparameters for AM=STD in All Environments***.

OPEN - Time of File Opening

`OPEN=`*`value`* specifies when the file is to be opened:

Value	Explanation: The file is opened ...
`INIT`	for output at session initialization.
`OBF`	according to the default `OPEN` value for the different environments (batch, CICS, Com-plete, TSO).
`OBJ`	when the execution of the first object which accesses the file starts. This is the default value.
`INITOBF`	for output at session initialization. Any subsequent re-opening of the file sets the default `OPEN` value for the different environments (batch, CICS, Com-plete, TSO).
`OBJ1`	when the execution of the first object on level 1 which accesses the file starts. Otherwise, it is opened when it is first accessed.
`ACC`	when it is first accessed by a statement.
`INITOBJ`	for output at session initialization. Any subsequent re-opening of the file will be performed when the execution of the first object which accesses the file starts.
`INITOBJ1`	when the execution of the first object on level 1 which accesses the file starts. Otherwise, it is opened when it is first accessed.
`INITACC`	for output at session initialization. Any subsequent re-opening of the file will be performed when it is first accessed by a statement.

PAD - Padding of Output Records

`PAD=`*`value`* specifies whether the output records are padded or not (applies only to data sets of fixed record length).

Value	Explanation
`ON`	Output records that are shorter than the record length (`LRECL`) of the data set will be padded with padding characters defined by keyword subparameter `PADCHRO`. This is the default value.
`OFF`	Error NAT1510 will be issued if an output record is shorter than the data set record length.

PADCHRI - Padding Character of Input Records

`PADCHRI=value` specifies the character which is used for padding of input records.

Value	Explanation

`'x'`
or
`x'xx'`
	One character *x* within single quotes
	or
	one hex character *xx*.

`x'40'`	Blank. This is the default value.

PADCHRO - Padding Character of Output Records

`PADCHRO=value` specifies the character which is used for padding of output records if `PAD=ON` is defined for the work file.

Value	Explanation

`'x'`
or
`x'xx'`
	One character *x* within single quotes
	or
	one hex character *xx*.

`x'00'`	This is the default value.

TRUNC - Truncation of Output Records

`TRUNC=value` specifies whether the output records are truncated or not.

Possible values:

Value	Explanation
`ON`	Output records that are longer than the record length (`LRECL`) of the data set will be truncated.
`OFF`	Error NAT1512 will be issued if an output record is longer than the data set record length. This is the default value.

Keyword Subparameters for AM=STD in All Environments

The following keyword subparameters are available for `AM=STD` in all environments:

`BLKSIZE` | `LRECL` | `RECFM`

BLKSIZE - Default Block Size of Data Set

`BLKSIZE=`*`value`* specifies the default block size of the data set.

Value	Explanation

`0`
or
`8 - 32767`
	Default block size of the data set (in bytes).

`4628`	This is the default value.

Note: The `BLKSIZE` specification only applies if no block size is predefined in the JCL or (under z/OS only) in the data set DCB.

LRECL - Default and Maximum Record Length of Data Set

`LRECL=`*`value`* specifies the record length of the data set.

Value	Explanation

`0`
or
`5 - 32767`
	Record length of the data set (in bytes).

`0`	This is the default value.

Notes:

1. This subparameter is used particularly to check for truncation and padding.
2. For `RECFM=V(B)` the `LRECL` value includes a 4-byte record descriptor word.
3. If `LRECL=0` is defined, the following applies: With `RECFM=V(B)`, `LRECL` defaults to `BLKSIZE-4`. With `RECFM=U`, `LRECL` defaults to `BLKSIZE`. With `RECFM=F(B)`, the maximum record length in the Natural program being executed is taken when the file is opened. If no record length from a program is available when the file is opened, for example with `OPEN=INIT`, this leads to an error.
4. For a work file defined as DUMMY output data set (z/OS only) and with `OPEN=INIT`, a sufficient `LRECL` value must be defined in the JCL or with the `LRECL` subparameter defined for the file.
5. The `LRECL` specification only applies if no record length is predefined in the JCL or (z/OS only) in the DCB data set.

RECFM - Default Record Format of Data Set

`RECFM=value` specifies the default record format of the data set.

Supported formats:

Value	Format
`F`	Fixed
`V`	Variable
`U`	Undefined
`B`	Blocked
`S`	Spanned
`A`	ASA
`M`	Machine control characters

Possible values or combinations of values:

Value	Explanation
`F, FA, FM, FB, FBA, FBM, V, VA, VM, VB, VBA, VBM, VBS, VBSA, VBSM, U, UA, UM`	These values or combinations of values that can be specified.
`VB`	Variable blocked. This is the default value.

Note: The `RECFM` specification only applies if no record format is predefined in the JCL or (under z/OS only) in the data set DCB.

Keyword Subparameters for AM=STD in z/OS Environments

The following keyword subparameters are available for `AM=STD` in z/OS environments:

`BUFNO` | `DISP` | `FREE` | `REREAD` | `VMAX`

BUFNO - Default Number of z/OS I/O Buffers of Data Set

`BUFNO=value` specifies the default number of z/OS I/O buffers of the data set.

Value	Explanation
`0` or `1 - 255`	Default number of z/OS I/O buffers of the data set.
`0`	With `BUFNO=0`, z/OS allocates five I/O buffers per default. This is the default value.

Notes:

1. The number of I/O buffers can improve the performance of work file access dramatically. Note that the storage for I/O buffers is allocated below the 16 MB line.
2. The `BUFNO` specification applies only if the `BUFNO` parameter is not specified in the JCL for the data set.

DISP - Open Work File for Modification

`DISP=value` specifies that the work file is opened for modification.

Value	Explanation
`MOD`	New records are added at the end of the file. **Note:** This corresponds to the JCL DD statement subparameter `DISP=MOD`.
`NOMOD`	The work file is rewritten from the start. This is the default value.

FREE - Data Set De-allocation at File Closure

`FREE=value` specifies whether the data set is de-allocated when the file is closed.

Value	Explanation
`ON`	The `FREE` option is set for the CLOSE SVC, which means that the data set is de-allocated when it is closed (and not at step termination).
`OFF`	The `FREE` option is not set for the CLOSE SVC. This is the default value.

REREAD - Closing of Tape File Data Sets

`REREAD=value` specifies the `REREAD` option for the closing of the tape file.

Value	Explanation
`ON`	The `REREAD` option is set for the `CLOSE SVC`. This causes the volume to be repositioned to reprocess the data set. This is the default value.
`OFF`	The `REREAD` option is not set for the `CLOSE SVC`.

VMAX - Control LRECL for Variable Record Format

`VMAX=value` controls the `LRECL` setting for an output file with variable record format (`RECFM=V`).

Value	Explanation
`ON`	Providing a non-zero `BLKSIZE` value exists for the file, `VMAX=ON` sets `LRECL=BLKSIZE-4` for variable record format, regardless of the `LRECL` setting in the DCB or the `LRECL` subparameter.
`NAT`	`LRECL` is set to the length +4 of the largest record in the application program if this value is less than `LRECL` in the DCB for the data set.
`OFF`	`LRECL` from the DCB for the data set is used. This is the default value.

Keyword Subparameters for AM=STD in z/VSE Environments

The following keyword subparameters are available for `AM=STD` in z/VSE environments:

`BLOCKS` | `DISP` | `LABEL` | `REWIND` | `SYSNR`

BLOCKS - Number of Storage Blocks

`BLOCKS=value` specifies the number of file blocks or file tracks to be allocated for a dynamic `NATVSE` work file.

Value	Explanation
`1 - 9999`	Number of file blocks or file tracks to be allocated.
`20`	This is the default value.

Note: See *NATVSE Dynamic Work File Allocation (DYNALLOC) Support* in the *Operations* documentation.

DISP - Work File Disposition for VSAM/SAM

`DISP=(`*`value1`*`,`*`value2`*`)` specifies the disposition of a dynamic `NATVSE` work file controlled by VSAM/SAM.

Value Pair	Explanation
`(NEW,KEEP)`	File is to be reset at open and to be kept at close. This is the default value.
`(NEW,DELETE)`	File is to be reset at open and to be made inaccessible at close.
`(OLD,DELETE)`	File is not to be reset at open and to be made inaccessible at close.
`(OLD,KEEP)`	File is not to be reset at open and to be kept at close.

Note: See *NATVSE Dynamic Work File Allocation (DYNALLOC) Support* in the *Operations* documentation.

LABEL - Tape Label Processing

`LABEL=`*`value`* specifies the tape label processing.

Value	Explanation
`ON`	The tape is in standard label format. This is the default value.
`OFF`	The tape is unlabeled with front tape mark.
`NOTM`	The tape is unlabeled without front tape mark.

REWIND - Action at File Closure

`REWIND=`*`value`* specifies the action to be taken when a tape file is closed.

Value	Explanation
`ON`	The tape is rewound when the file is closed. This is the default value.
`OFF`	The tape is not rewound when the file is closed.
`UNLOAD`	The tape is unloaded when the file is closed.

SYSNR - Logical VSE SYS Number

`SYSNR=value` specifies the logical VSE SYS number.

Value	Explanation
`1 - 99`	Logical VSE SYS number. By default, the SYS number is identical to the work file number.

Keyword Subparameters for AM=STD in BS2000 Environments

The following keyword subparameter is available for `AM=STD` in BS2000 environments:

`DISP`

DISP - File Open Mode

`DISP=value` specifies the open mode of the file.

Value	Explanation
`EXT`	The open mode is set to `EXTEND`.
`NOEXT`	The open mode is set to the default value `OUTPUT`. This is the default value.

Keyword Subparameters for AM=CICS

The following keyword subparameters are available for `AM=CICS`:

`DISP` | `TYPE`

DISP - CICS Temporary Storage Queue Disposition

`DISP=(value1,value2)` specifies the CICS temporary storage queue disposition.

Value Pairs	Explanation
`(NEW,KEEP)`	The storage queue is deleted when the file is opened. This is the default value.
`(NEW,DELETE)`	The storage queue is deleted when the file is opened and when it is closed.
`(OLD,DELETE)`	The storage queue is deleted when the file is closed.
`(OLD,KEEP)`	The storage queue is not deleted.

Note: The `DISP` specification does not apply to CICS extra-partition transient data queues.

TYPE - Type of CICS Storage Medium

`TYPE=`*`value`* specifies the type of CICS storage medium to be used.

Value	Explanation
`MAIN`	Temporary main storage.
`AUX`	Temporary auxiliary storage.
`TD`	Transient data.

Note: The default value used depends on the setting of the subparameter `DEST`. If the `DEST` subparameter value matches a valid CICS transient data queue, the `TYPE` subparameter defaults to `TD`, otherwise `MAIN` will be taken as default value.

Keyword Subparameters for AM=COMP

The following keyword subparameters are available for `AM=COMP`:

`BLKSIZE` | `BLOCKS` | `TYPE`

BLKSIZE - Size of Storage Blocks

`BLKSIZE=`*`value`* specifies the default block size of the data set.

Value	Explanation

`0`
or
`8 - 32767`
	Default block size of the data set (in bytes).

`4628`	This is the default value.

BLOCKS - Number of Storage Blocks

`BLOCKS=value` specifies the number of storage blocks to be allocated.

Value	Explanation
`1 - 9999`	Number of storage blocks to be allocated.
`20`	This is the default value.

TYPE - Type of Storage Access

`TYPE=value` specifies the type of storage access to be used.

Value	Explanation
`SHR`	Shared access; that is, the work file is accessible by all users.
`TID`	The work file is only available to the current Com-plete terminal ID.
`DYN`	The work file is only available to the current terminal stack level.

Keyword Subparameters for AM=SMARTS

The following keyword subparameters are available for `AM=SMARTS`:

`DEST` | `DISP` | `TYPE`

DEST - Work File Name

`DEST=value` specifies the work file name.

Value	Explanation
1 - 8 characters.	Work file name.

Notes:

1. Since the `DEST` clause is restricted to an 8 character maximum, it is useless to define a file with absolute PFS path specification.
2. The name specified in the `DEST` clause is relative to the work file root directory. The work file root directory is specified with the environment variable `NAT_WORK_ROOT`.
3. To specify a file with absolute path definition, use the `DEFINE WORK FILE` statement.

DISP - File Open Mode

`DISP=(value1,value2,value3)` specifies the mode of the work file.

Value	Explanation	
value1	*value1* specifies whether an existing file should be deleted or new data should be appended to the file.	
	`NEW`	An existing file will be deleted if the file is opened for writing. This is the default value.
	`OLD` or `MOD`	New data written are appended at the end of the file.
value2	*value2* specifies whether a file should be kept or removed after access.	
	`KEEP`	Permanent file that will be kept after close. This is the default value.
	`DELETE`	Temporary file that will be removed after close.
value3	*value3* specifies whether a user has exclusive access to the file or not.	
	`SHR`	Shared access; that is, the work file is accessible by all users. This is the default value.
	`OWN`	Exclusive access, the work file is accessible to the current Comp-lete user ID. Files with exclusive access are located in an additional directory which has the name of the current user ID.

Example 1:

```
DISP=(NEW,KEEP,SHR)
```

Example 2:

If you specify only the first value (with or without parentheses), the other values will assume their default settings:

```
DISP=(MOD)
```

or

```
DISP=MOD
```

Both specifications correspond to:

```
DISP=(MOD,KEEP,SHR)
```

TYPE - Type of Storage Access

`TYPE=`*`value`* specifies the type of storage access to be used.

Value	Explanation
`BIN`	Each line is written to the work file without terminating end-of-line character. This is the default value.
`TXT`	Each line is written to the work file with a terminating end-of-line character (`x'15'`).

283 WPSIZE - Sizes of Natural Work Pools

This Natural profile parameter specifies the sizes of the Natural work pools below and above the 16 MB line for one Natural session.

Possible settings	See *Syntax Description*.	
Default setting	`(32,128,2097151,2097151)`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. Natural uses work pools below and above the 16 MB line. In these work pools, all temporary buffers physical storage requests are satisfied.
2. Natural uses physical storage in special situations only, for example, for passing parameter areas outside the thread (while the thread is released) during the execution of the `CALL` statement with the "call by value option" indicated by a `SET CONTROL 'P=V'` statement under CICS.
3. The advantage of work pools is that, if there are many requests for physical storage, Natural can satisfy these requests by itself rather than by passing it to the operating system.

Syntax Description

The `WPSIZE` parameter is specified as follows:

WPSIZE=(*size-below*,*size-above*,*maximum-below*,*maximum-above*)

Where:

Syntax Element	Explanation
size-below	*size-below* (`0-1024`) is the size of one work pool in KB below the 16 MB line. The value `0` means that no work pool is allocated, i.e. all requests for physical storage below 16 MB are passed directly to the operating system.
size-above	*size-above* (`0-16384`) is the size of one work pool in KB above the 16 MB line. The value `0` means that no work pool is allocated, that is, all requests for physical storage above 16 MB are passed directly to the operating system.
maximum-below	*maximum-below* (`0-2097151`) limits the total physical storage in KB which can be allocated below the 16 MB line. The value `0` means no physical storage can be allocated below the 16 MB line.
maximum-above	*maximum-above* (`0-2097151`) limits the total physical storage in KB which can be allocated above the 16 MB line. The value `0` means no physical storage can be allocated above the 16 MB line.

Notes:

1. If a work pool is exhausted, another work pool of the specified work pool size is allocated.
2. If the size of the requested physical storage is larger than the specified work pool size, a `GETMAIN` request for that larger size is made.
3. Subparameters not to be changed can be omitted; for example, you can specify `WPSIZE=(,1000)` if you want to set the work pool size only above 16 MB to 1000 KB.
4. Natural allocates the work pools outside the Natural storage thread according to the specified settings. A work pool is allocated during the first request for physical storage and is released during the next terminal I/O.
5. For non-thread environments (for example, batch, TSO), the recommended setting is `WPSIZE=(0,0)`. This may save virtual storage. Exception: This recommendation does not apply if Natural Batch for zIIP, Natural for CICS for zIIP, Natural for Com-plete for zIIP or Natural for IMS for zIIP is installed and active (see also *Natural for zIIP*).
6. If Natural Batch for zIIP is installed and active (z/OS batch and TSO only), an appropriate setting of `WPSIZE` can reduce the number of switches into TCB mode, because of the reduced number of physical `GETMAIN`s. The same applies if the profile parameter `THSIZE` is used.

284 WSISIZE - Buffer for Natural Workstation Interface

This Natural profile parameter only applies if Natural Workstation Interface is installed.

Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the buffer size.

Possible settings	`10 - 256`	Size of buffer area in KB. If the required space is not available, the Natural Workstation Interface cannot be used.
	`0`	The Natural Workstation Interface cannot be used.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

285 XML - Activate PARSE XML and REQUEST DOCUMENT Statements

This Natural profile parameter is used to activate/deactivate the statements `REQUEST DOCUMENT` and `PARSE XML`. It corresponds to the `NTXML` macro in the Natural parameter module.

Possible settings	See *XML Parameter Syntax*.	
Default setting	`XML=(OFF)`	
Dynamic specification	yes	The parameter `XML` can only be specified dynamically. In the Natural parameter module, use the macro `NTXML`.
Specification within session	no	

Notes:

1. As a prerequisite for using the `XML` profile parameter, the profile parameter `CFICU` must be set to `CFICU=ON`.
2. See also *Statements for Internet and XML Access* in the *Programming Guide*.

The following topics are covered below:

XML Parameter Syntax

The `XML` parameter is specified as follows:

XML=(ON,*keyword-subparameter*=*value*,*keyword-subparameter*=*value*,...)

Or:

XML=(OFF,*keyword-subparameter*=*value*,*keyword-subparameter*=*value*,...)

Where:

Syntax Element / Value	Explanation
`ON`	Enable XML support according to the **keyword subparameter** settings. For the `PARSE XML` and `REQUEST DOCUMENT` statement usage, the subparameters `RDOC` and `PARSE` must be set to `ON` as well.
`OFF`	Disable XML support. Any subparameter settings are ignored. This is the default value.
keyword-subparameter	Names and values of possible keyword subparameters; see *Keyword Subparameters*.

NTXML Macro Syntax

The `NTXML` macro is specified as follows:

```
         NTXML  ON                                                         *
                PARSE=value,                                               *
                RDCP=value,                                                *
                RDIPV6=value,                                              *
                RDNOP=value,                                               *
                RDP=value,                                                 *
                RDOC=value,                                                *
                RDPPORT=value,                                             *
                RDPPOV6=value,                                             *
                RDPS=value,                                                *
                RDPSV6=value,                                              *
                RDPV6=value,                                               *
                RDSPORT=value,                                             *
                RDSPOV6=value,                                             *
                RDTOUT=value,                                              *
                RDV4MAP=value
```

Or:

```
         NTXML  OFF
```

For a description of the syntax elements, see ***XML Parameter Syntax***.

Keyword Subparameters

`PARSE` | `RDCP` | `RDIPV6` | `RDNOP` | `RDOC` | `RDP` | `RDPPORT` | `RDPPOV6` | `RDPS` | `RDPSV6` | `RDPV6` | `RDSPORT` | `RDSPOV6` | `RDTOUT` | `RDV4MAP` |

Note: The keyword subparameters `RDPS`, `RDPSV6`, `RDSPORT` and `RDSPOV6` are currently for z/OS only.

PARSE - Support of PARSE XML Statement

`PARSE=`*`value`* enables/disables the support of the `PARSE XML` statement.

Value	Explanation
`ON`	Use of the `PARSE XML` statement is supported.
`OFF`	Use of the `PARSE XML` statement is not supported. This is the default value.

RDCP - Name of the Default HTML/XML Code Page

`RDCP=`*`value`* specifies the default code page which is assumed if *`code-page-in`* in the `REQUEST DOCUMENT` statement contains only spaces.

Value	Explanation
`code-page-name`	Name of the default code page.
`ISO 8859-1:1987`	This is the default value.

RDIPV6 - Support for IPv6 for the REQUEST DOCUMENT Statement

`RDIPV6=`*`value`* enables/disables the support for IPv6 for the `REQUEST DOCUMENT` statement.

Value	Explanation
`ON`	Use of the IPv6 protocol is enabled, if available on the local host.
`OFF`	Use of the IPv6 protocol is disabled. This is the default value.

RDNOP - Name of Local Domain

`RDNOP=`*`value`* specifies local domain(s) which are to be addressed directly, not via the proxy.

Value	Explanation
`domain-name(s)`	Name(s) of local domains. **Note:** 1. Blanks are not allowed. 2. Wildcard notation for prefixes can only be used in the form `*.`*`xxx`* and not in the form `.`*`xxx`*. 3. Multiple name entries must be separated by a semicolon.

Value	Explanation
`OFF`	`RDNOP=OFF` means that no URL is defined. This is the default value.

Note: Specification of IPv6 address or IPv6 address prefixes is possible.

RDOC - Support of REQUEST DOCUMENT Statement

`RDOC=`*`value`* enables/disables the support of the `REQUEST DOCUMENT` statement.

Value	Explanation
`ON`	Use of the `REQUEST DOCUMENT` statement is supported.
`OFF`	Use of the `REQUEST DOCUMENT` statement is not supported. This is the default value.

RDP - URL of Proxy Server

`RDP=`*`value`* specifies the URL of the proxy server through which all internet (not intranet) HTTP requests have to be routed.

Value	Explanation
`url`	URL of the proxy server. Blanks are not allowed.
`OFF`	`RDP=OFF` means that no URL is defined. This is the default value.

RDPPORT - Proxy Port Number

`RDPPORT=`*`value`* specifies the port number of the proxy, if any is set.

Value	Explanation
`0` or `1 - 65535`	Port number.
`80`	This is the default value.

RDPPOV6 - IPv6 Proxy Port Number

`RDPPOV6=`*`value`* specifies the port number of the IPv6 proxy, if any is set.

Value	Explanation
`0` or `1 - 65535`	Port number.
`80`	This is the default value.

RDPS - URL of SSL Proxy Server

`RDPS=`*`value`* specifies the URL of the SSL proxy server through which all internet (not intranet) HTTPS requests have to be routed.

Value	Explanation
`url`	URL of the SSL proxy server. Blanks are not allowed.
`OFF`	`RDPS=OFF` means that no URL is defined. This is the default value.

Note: This keyword subparameter is currently for z/OS only.

RDPSV6 - URL of IPv6 SSL Proxy Server

`RDPSV6=`*`value`* specifies the URL of the IPv6 SSL proxy server through which all internet (not intranet) HTTPS requests have to be routed.

Value	Explanation
`url`	URL of the IPv6 SSL proxy server. Blanks are not allowed.
`OFF`	`RDPSV6=OFF` means that no URL is defined. This is the default value.

Notes:

1. This keyword subparameter is currently for z/OS only.
2. Specification of an IPv6 address is possible.

RDPV6 - URL of IPv6 Proxy Server

`RDPV6=`*`value`* specifies the URL of the IPv6 proxy server through which all internet (not intranet) HTTP requests have to be routed.

Value	Explanation
`url`	URL of the IPv6 proxy server. Blanks are not allowed.
`OFF`	`RDPV6=OFF` means that no URL is defined. This is the default value.

Note: Specification of an IPv6 address is possible.

RDSPORT – SSL Proxy Port Number

`RDSPORT=`*`value`* specifies the port number of the SSL proxy, if any is set.

Value	Explanation
`0` or `1 - 65535`	Port number of the SSL proxy.
`443`	This is the default value.

Note: This keyword subparameter is currently for z/OS only.

RDSPOV6 – IPv6 SSL Proxy Port Number

`RDSPOV6=`*`value`* specifies the port number of the IPv6 SSL proxy, if any is set.

Value	Explanation
`0` or `1 - 65535`	Port number of the IPv6 SSL proxy.
`443`	This is the default value.

Note: This keyword subparameter is currently for z/OS only.

RDTOUT - Timeout Value for Ongoing HTTP Requests

`RDTOUT=value` specifies the timeout (in seconds) for HTTP requests that are in progress.

This keyword subparameter is not supported by the IPv4-only load modules `NAT2TCP4` (z/OS) and `NCFIP482` (z/VSE).

Value	Explanation
`0`	No timeout control provided by Natural, the default settings apply. This is the default value.
`1-32767`	Time in seconds after which a timeout error is issued if one of the following socket functions cannot be completed within the specified interval: connect, send or receive.

RDV4MAP - Support for IPv4-Mapped Addresses under IPv6

`RDV4MAP=value` enables or disables the use of IPv4-mapped IPv6 addresses for URLs in symbolic notation. Numeric IPv4-mapped IPv6 addresses are always enabled in IPv6 mode.

This keyword subparameter is not supported by the IPv4-only load modules `NAT2TCP4` (z/OS) and `NCFIP482` (z/VSE).

Value	Explanation
`ON`	Enabled: IPv4-mapped addresses are allowed within the IPv6 protocol.
`OFF`	Disabled: IPv4-mapped addresses are not allowed within the IPv6 protocol. This is the default value.

For Security Reasons:

Use IPv4-mapped addresses only in IPv6 environments where no IPv4 TCP/IP stack (or dual stacking mode) is available to access IPv4-based HTTP servers. For detailed information on IPv4-mapped IPv6 addresses, refer to the appropriate IPv6 documentation.

Examples of XML Parameter

```
XML=(ON,RDP='HTTPPROXY.MYCOMPANY.COM',RDPPORT=8080,RDPS='SSLPROXY.MYCOMPANY.COM',RDSPORT=443,RDNOP='*.MYCOMPANY.COM',RDOC=ON,PARSE=ON)
```

```
XML=(ON,RDP='HTTPPROXY.MYCOMPANY.COM',RDPPORT=8080,RDPS='SSLPROXY.MYCOMPANY.COM',RDSPORT=443,RDNOP='*.MYCOMPANY.COM;2AE0:4899:200:1E00:',
     RDOC=ON,PARSE=ON,RDIPV6=ON,RDPV6='V6HTTPPROXY.MYCOMPANY.COM',RDPPOV6=888)
```

Note: The keyword subparameters `RDSPORT`, `RDSPOV6`, `RDPSV6` and `RDPS` are currently for z/OS only.

Examples of NTXML Macro

```
         NTXML ON,RDP=HTTPPROXY.MYCOMPANY.COM,                         *
               RDPPORT=8080,                                           *
               RDPS=SSLPROXY.MYCOMPANY.COM,                            *
               RDSPORT=443,                                            *
               RDNOP=*.MYCOMPANY.COM,                                  *
               RDOC=ON,                                                *
               PARSE=ON
```

```
         NTXML ON,RDP=HTTPPROXY.MYCOMPANY.COM,                         *
               RDPPORT=8080,                                           *
               RDPS=SSLPROXY.MYCOMPANY.COM,                            *
               RDSPORT=443,                                            *
               RDNOP=*.MYCOMPANY.COM,                                  *
               RDOC=ON,                                                *
               RDIPV6=ON,                                              *
               RDPV6=V6HTTPPROXY.MYCOMPANY.COM,                        *
               RDPSV6=V6SSLPROXY.MYCOMPANY.COM,                        *
               RDSPOV6=8443
```

Note: The keyword subparameters `RDSPORT`, `RDSPOV6`, `RDPSV6` and `RDPS` are currently for z/OS only.

286 XREF - Creation of XRef Data for Natural

This Natural profile parameter is used to enable/disable the creation of XRef data for Natural. This parameter also determines how XRef data are treated when Natural members are processed with the Natural utilities `SYSMAIN` or `INPL` or with the Object Handler.

Possible settings	`ON`	XRef data are generated in the cases described above. Documentation premise is not checked.
	`OFF`	XRef data are not generated. Documentation premise is not checked.
	`FORCE`	A Natural object can only be cataloged if a documentation object already exists for this implementation object. XRef data are generated in the cases described above.
	`DOC`	A Natural object can only be cataloged if a documentation object already exists for this object. XRef data are not generated.
Default setting	`OFF`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	none	
Applicable commands	`XREF`	

The following topics are covered below:

Possibilities of Setting the XREF Parameter

There are different ways to set the Natural `XREF` parameter:

- In the Natural parameter module.
- As a dynamic parameter when starting a Natural session.
- In Natural Security. If Natural Security has been used to set the `XREF` parameter, the `XREF` command may only be used to enforce this setting (by changing from `ON` to `FORCE`, from `OFF` to `ON` or `FORCE`).
- With the Natural `XREF` command. If Natural Security is not installed, the `XREF` parameter is usually set with the Natural `XREF` command. The Natural command `XREF ?` displays the current setting of the `XREF` parameter.

XRef Data Generation

XRef data is generated in two cases:

- The Natural compiler writes XRef data for Natural programs and data areas when these are cataloged (provided that the `XREF` parameter has been set to either `ON` or `FORCE`, see below).
- Natural Security writes XRef data for programs that are used as Startup, Restart or Error-Transaction in an application or as a special link if the `XREF` parameter is set to `ON` or `FORCE` in the application's Natural Security definition and a user system file is defined for the application.

The `XREF` parameter controls the compilation in two aspects:

- generation of XRef data in the cases described above and
- fulfilment of premise to document implementation objects. The adherence to this premise can be ensured by allowing the completion of the catalog operation only for objects that are documented in the Predict `FDIC` system file or in the development server file used in Natural Single Point of Development (SPoD).

Extended XRef Data Generation (For Internal Use Only)

The extended `XREF` parameter is reserved for internal use by Natural.

287 YD - Year Differential

This Natural profile parameter can be used to adjust the current machine date (as read by using the internal machine time) by adding/subtracting a number of years to/from it. This may be useful for countries that use different calendars.

Possible settings	`-499` to `499`	The parameter is specified as `YD=+`*`nnn`* or `YD=-`*`nnn`* where *`nnn`* is the number of years. **The following considerations apply:** 1. If the current year is a leap year, but the year resulting from the `YD` setting is not, the 1st March will be used instead of the 29th February. 2. The date resulting from the sum of the profile parameters `TD`, `DD` and `YD` must be in the range of January 1, 1582 through December 31, 2699, if the `MAXYEAR` profile parameter is set to its default setting of `2699`. 3. If the profile parameter `MAXYEAR` is set to `9999`, the maximum `YD` value is calculated as follows: `9999 - current machine date year = maximum YD value` This means, that the maximum `YD` value decreases with each increase of the current machine date year. It is never a fix value. For example, if `MAXYEAR` is set to `9999` and the current machine date year is 2017, then the calculated maximum `YD` value is `7982`.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

288 YSLW - Year Sliding or Fixed Window

This Natural profile parameter specifies the range of years covered by the "year sliding window" or "year fixed window".

Note: The sliding-window or "year fixed window" mechanism assumes a date with a 2-digit year to be within a "window" of 100 years. Within these 100 years, every 2-digit year setting is uniquely related to a specific century, so that there is no confusion about which century is meant.

Possible settings	Normal Setting	`0`	When you set the parameter to `0`, the current century is assumed. No sliding or fixed-window mechanism is used.
	Sliding Window	`1 - 99`	By setting the parameter to a value between `1-99`, you determine when the 100-year range begins in the past. The `YSLW` setting is subtracted from the current year to determine the first year of the window range. See *Example of a Sliding Window*.
	Fixed Window	`1582-2600`	By setting the parameter to a value between `1582-2600`, you determine the first year of a 100-year range. The upper boundary of the 100-year range is evaluated by adding `99` to the value specified. See *Example of a Fixed Window*.
Default setting	`0`		No sliding or fixed-window mechanism is used.
Dynamic specification	yes		
Specification within session	no		

The `YSLW` parameter is evaluated at runtime when an alphanumeric date setting with a 2-digit year component is moved into a date variable. This applies to date settings which are:

- used with the mathematical function `VAL`;
- used with the `IS(D)` option in a logical condition;
- read from the stack as input data;
- or entered in a map as input data.

See also the section *Processing of Date Information* in the *Programming Guide*.

Examples of YSLW Parameter

Example of a Sliding Window

If the current year is 2014 and you specify `YSLW=40`, the sliding window will cover the years 1974 to 2073. A 2-digit year setting *nn* from `74` to `99` is then interpreted accordingly as `19`*nn*, while a 2-digit year setting *nn* from `00` to `73` is interpreted as `20`*nn*.

See also the examples under *Year Sliding Window - YSLW Parameter* and *Combinations of DFSTACK and YSLW* in the *Programming Guide*.

Example of a Fixed Window

If you specify `YSLW=1985`, the fixed window will cover the years 1985 to 2084. A 2-digit year setting *nn* from 85 to 99 is then interpreted accordingly as `19`*nn*, while a 2-digit year setting *nn* from `00` to `84` is interpreted as `20`*nn*.

289 ZD - Zero-Division Check

This Natural profile and session parameter specifies the action to be taken when an attempt is made to perform a division operation in which the divisor is `0`.

Possible settings	`ON`	Natural issues an error message if a division by `0` is attempted.
	`OFF`	Natural returns a result of `0` for any division operation in which the divisor is `0`.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	
Applicable statements	`SET GLOBALS`	
Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. Within a Natural session, the profile parameter `ZD` can be overridden by the session parameter `ZD`.
2. Under Natural Security, the setting of this parameter can be overridden by the *Session Parameters* option of the Library Profile.

290

ZIIP - zIIP Processing (z/OS Only)

This Natural profile parameter can be used to configure zIIP processing under Natural. It corresponds to the `NTZIIP` macro in the Natural parameter module.

Note: The `ZIIP` setting takes effect only if Natural Batch for zIIP, Natural for CICS for zIIP, Natural for Com-plete for zIIP or Natural for IMS for zIIP has been installed in your z/OS environment. For further information, refer to *Installing Natural for zIIP* in the *Installation for z/OS* documentation.

Possible settings	See ***ZIIP Parameter Syntax***.	
Default setting	`AUTO`	See ***ZIIP Parameter Syntax*** and ***Keyword Subparameters***.
Dynamic specification	yes	The parameter `ZIIP` can be specified dynamically only. In the Natural parameter module, use the macro `NTZIIP`.
Specification within session	yes	By means of the system command `ZIIP`, the zIIP component switch statistics can be controlled and information about zIIP processing can be displayed.

The following topics are covered below:

ZIIP Parameter Syntax

The `ZIIP` parameter is specified as follows:

```
ZIIP=(state,keyword-subparameter=value,keyword-subparameter=value,...)
```

Or:

```
ZIIP=state
```

Where *`state`* can be `ON`, `OFF` or `AUTO`.

Value	Explanation
`ON`	zIIP support will be activated.
`OFF`	zIIP support will not be activated.
`AUTO`	zIIP support will be activated only if there is at least one zIIP online, or if the z/OS parameter `PROJECTCPU=YES` is set in the `SYS1.PARMLIB` member `IEAOPT`. This is the default value.
`keyword-subparameter`	See ***Keyword Subparameters***.

Note: If specified as a list within brackets, `ON`, `OFF` or `AUTO` must be the first value.

NTZIIP Macro Syntax

The `NTZIIP` macro is specified as follows:

```
         NTZIIP state,                                                *
               IMSG=value,                                            *
               IOCSIZE=value,                                         *
               PNR=value,                                             *
               PRINT=value,                                           *
               PWCSIZE=value,                                         *
               PWCXAM=value,                                          *
               STAT=value,                                            *
               TMSG=value
```

See *Keyword Subparameters*.

For a description of `state`, see ***ZIIP Parameter Syntax***.

Keyword Subparameters

`IMSG` | `IOCSIZE` | `PNR` | `PRINT` | `PWCSIZE` | `PWCXAM` | `STAT` | `TMSG`

IMSG – Natural zIIP Support Message

This keyword subparameter determines whether Natural issues a NAT7070 system message after Natural zIIP support has been successfully enabled during session initialization.

Value	Explanation
`ON`	The zIIP support message is issued. This is the default value.
`OFF`	The zIIP support message is not issued.

IOCSIZE - Cache Size for Primary Batch I/O Files

This keyword subparameter can be used to determine the cache sizes for processing the primary batch I/O files CMSYNIN, CMOBJIN and CMPRINT. This can be useful when running Natural in zIIP mode because it reduces the overhead caused by switching from zIIP to GCP (general central processor) and vice versa. There is one cache for each file, the CMSYNIN input file, the CMOBJIN input file and the CMPRINT output file.

The advantage of the cache is that all I/O data is collected in the cache instead of accessing a file immediately. When the cache for the CMPRINT file fills up, the cache is flushed, that is, all records are written at once. For a CMSYNIN or CMOBJIN input file, respectively, all records for the given file are read and stored in the input cache for this file. Subsequently, Natural reads the input records directly from the input cache.

Caution: The usage of `IOCSIZE` can have a negative impact on your application flow. For example, an application can receive sent records at a later time since the records are not written immediately to CMPRINT.

IOCSIZE Parameter Syntax

The subparameter `IOCSIZE` is specified as follows:

IOCSIZE=(*`CMSYNIN-cache-size`*, *`CMOBJIN-cache-size`*, *`CMPRINT-cache-size`*)

Where:

Syntax Element	Value	Explanation
`CMSYNIN-cache-size`	`1 - 2097151`	The *`CMSYNIN-cache-size`* in KB is used to allocate the cache for the CMSYNIN input file.
	`0`	No cache for the CMSYNIN input file is allocated. This is the default value.
`CMOBJIN-cache-size`	`1 - 2097151`	The *`CMOBJIN-cache-size`* in KB is used to allocate the cache for the CMOBJIN input file.
	`0`	No cache for the CMOBJIN input file is allocated. This is the default value.
`CMPRINT-cache-size`	`64 - 2097151`	The *`CMPRINT-cache-size`* in KB is used to allocate the cache for the CMPRINT output file.
	`0`	No cache for the CMPRINT output file is allocated. This is the default value.

Notes:

1. The default setting is `IOCSIZE=(0,0,0)`. If you want to change a default value, you only need to specify the size to be changed, for example, `IOCSIZE=(,200)` to allocate 200 KB for the CMOBJIN cache file only.

2. You cannot change the `IOCSIZE` setting during a session.

PNR – Print File for zIIP Processing Information (in Batch Sessions only)

This keyword subparameter can be used to determine the file where the information about zIIP processing (see the keyword subparameter `PRINT`) is printed at the end of a batch session. `PNR` is ignored in server and TSO sessions.

Value	Explanation
`0`	The information about zIIP processing is routed to the standard printer (CMPRINT). This is the default value.
`1 - 31`	The information about zIIP processing is routed to the specified print file. If the specified print file is unavailable during session termination, Natural returns an error message. For more information about the definition of Natural print files, see the profile parameter `PRINT`.

PRINT – Print zIIP Processing Information (in Batch Sessions only)

This keyword subparameter can be used to get information printed about zIIP processing automatically at the end of a batch session. It is ignored in server and TSO sessions.

Value	Explanation
`INFO`	Print zIIP processing general information only.
`STAT`	Print zIIP switch component statistics only.
`ALL`	Print both: zIIP processing general information and zIIP switch component statistics.
`OFF`	No print of zIIP processing information is generated at the end of the session. This is the default value.

PWCSIZE - Size of the Cache for Print and Work Files

This keyword subparameter can be used to determine the cache sizes for print and work file I/O processing. This may be useful when running Natural in zIIP mode, because it reduces the overhead caused by switching from zIIP to GCP (general central processor) and vice versa. There is one cache for print files, one for output work files, and one for each input work file.

The advantage of the cache is that all I/O data is collected in the cache instead of accessing the file immediately. When the cache for print files or output work files fills up, the cache is flushed, that is, all records are written to their respective files at once. In case of work file input, all records for a given file are read and stored in the input cache for this file. Subsequently, Natural reads the input records directly from the input cache.

Caution: The usage of `PWCSIZE` can have a negative impact on your application flow. For example, an application may receive sent records at a later point of time since the records are not written immediately to the desired media. Moreover, the error behavior can be different. Example: If the error “NAT1507 - The work/print file is full.” occurs, it is not displayed immediately. Instead, the error message “NAT1532 - Error(s) during flush of print/work file cache.” is displayed later during the cache flush. The error that actually occurred can then only be displayed with the system command `LASTMSG`. This means that an `ON ERROR` clause for NAT1507 no longer works.

PWCSIZE Parameter Syntax

The subparameter `PWCSIZE` is specified as follows:

PWCSIZE=(*print-size*,*work-input-size*,*work-output-size*)

Where:

Syntax Element	Value	Explanation
print-cache-size	`1 - 2097151`	The *print-cache-size* in KB is used to allocate the cache for all print files.
	`0`	No cache for all print files is allocated. This is the default value.
work-input-cache-size	`1 - 2097151`	The *work-input-cache-size* in KB is used to allocate a cache for each input work file.
	`0`	No cache for each input work file is allocated. This is the default value.
work-output-cache-size	`1 - 2097151`	The *work-output-cache-size* in KB is used to allocate the cache for all output work files.
	`0`	No cache for all output work files is allocated. This is the default value.

Notes:

1. The default setting is `PWCSIZE=(0,0,0)`. If you want to change a default value, you only need to specify the size to be changed, for example, `PWCSIZE=(,200)` to allocate 200 KB for the input cache only.
2. You cannot change the `PWCSIZE` setting during a session.

PWCXAM - Disable Caching for Print and Work Files

This keyword subparameter can be used to disable the caching for print and work files (defined with the `PWCSIZE` keyword subparameter) for single or multiple access methods defined for the files.

Disabling caching for file access methods can be useful if you want to avoid negative impact on the logical flow of your application (see also the ***Warning*** for the `PWCSIZE` subparameter).

PWCXAM Parameter Syntax

The subparameter `PWCXAM` is specified as follows:

For a single access method:

```
PWCXAM=access-method
```

For multiple access methods:

```
PWCXAM=(access-method1,access-method2,...)
```

The syntax above only applies to the `NTZIIP` macro in the Natural parameter module. If you want to define more than one value dynamically, specify `PWCXAM` multiple times.

`access-method` defines the type of access method to be used. The values you can specify correspond to the values of the **`AM` subparameter** of the `PRINT` profile parameter and the **`AM` subparameter** of the `WORK` profile parameter, respectively.

STAT – zIIP Switch Component Statistics

This keyword subparameter controls the collection of component statistics for the switches into TCB-mode. The statistics can be displayed by the system command `ZIIP` during the session, or the report can be triggered by the subparameter `PRINT` automatically at the end of the session (batch only). Moreover, it is possible to control the statistics by means of the `ZIIP` command.

Value	Explanation
`ON`	Activate zIIP switch component statistics.
`OFF`	Deactivate zIIP switch component statistics. This is the default value.

Note: The setting of `STAT` can be overridden by the `STAT` option of the `ZIIP` system command and the application programming interface `USR8204N` which performs `ZIIP` command functions. See the relevant sections in the *System Commands* documentation.

TMSG – Destination for Natural zIIP Support Initialization Messages

This keyword subparameter determines whether Natural sends zIIP support initialization messages to the job log or the Natural standard output device, for example, a user terminal for online environments or CMPRINT for batch. This can be helpful for testing purposes.

Value	Explanation
`ON`	All zIIP support initialization messages are sent to the Natural standard output device.
`OFF`	All zIIP support initialization messages are sent to the job log. This is the default value.

Example of ZIIP Parameter

```
ZIIP=(ON,STAT=ON,PWCSIZE=(200,,300),IOCSIZE=(100,100,500))
```

Example of NTZIIP Macro

```
         NTZIIP AUTO,                                                  *
               STAT=ON,                                                *
               PWCSIZE=(500,),                                         *
               IOCSIZE=(,,2000),                                       *
               PRINT=ALL
```

291 ZP - Zero Printing

This Natural profile and session parameter specifies how a field which contains a setting of all zeros is to be output.

Possible settings	`ON`	Each field value which consists of all zeros is output as one zero, right justified (for numeric fields) or all zeros (for time fields).
	`OFF`	Each field value which consists of all zeros is suppressed.
Default setting	`ON`	
Dynamic specification	yes	
Specification within session	yes	

Applicable statements	`DISPLAY`
	`FORMAT`
	`INPUT`
	`PRINT`
	`REINPUT`
	`SET GLOBALS`
	`WRITE`	

Applicable command	`GLOBALS`	
Application programming interface	`USR1005N`	See *SYSEXT - Natural Application Programming Interfaces* in the *Utilities* documentation.

Notes:

1. This Natural profile and session parameter is used to suppress the display of a numeric field (format N, I, P or F) or time field (format T) which contains a value of all zeros.
2. Within a Natural session, the profile parameter `ZP` can be overridden by the session parameter `ZP`.
3. See also *Parameters to Influence the Output of Fields* in the *Programming Guide*.

292 ZSIZE - Size of Entire DB Buffer Area

This Natural profile parameter specifies the size of the buffer area required by Entire DB.

Possible settings	`1-64`	Size of the buffer area in KB.
	`0`	If `ZSIZE=0` or if the required space is not available, the Entire DB Interface cannot be used.
Default setting	`0`	
Dynamic specification	yes	
Specification within session	no	

Notes:

1. This Natural profile parameter only applies to Entire DB.
2. Alternatively, you can use the equivalent Natural profile parameter `DS` or macro `NTDS` to specify the size of the buffer.

Index