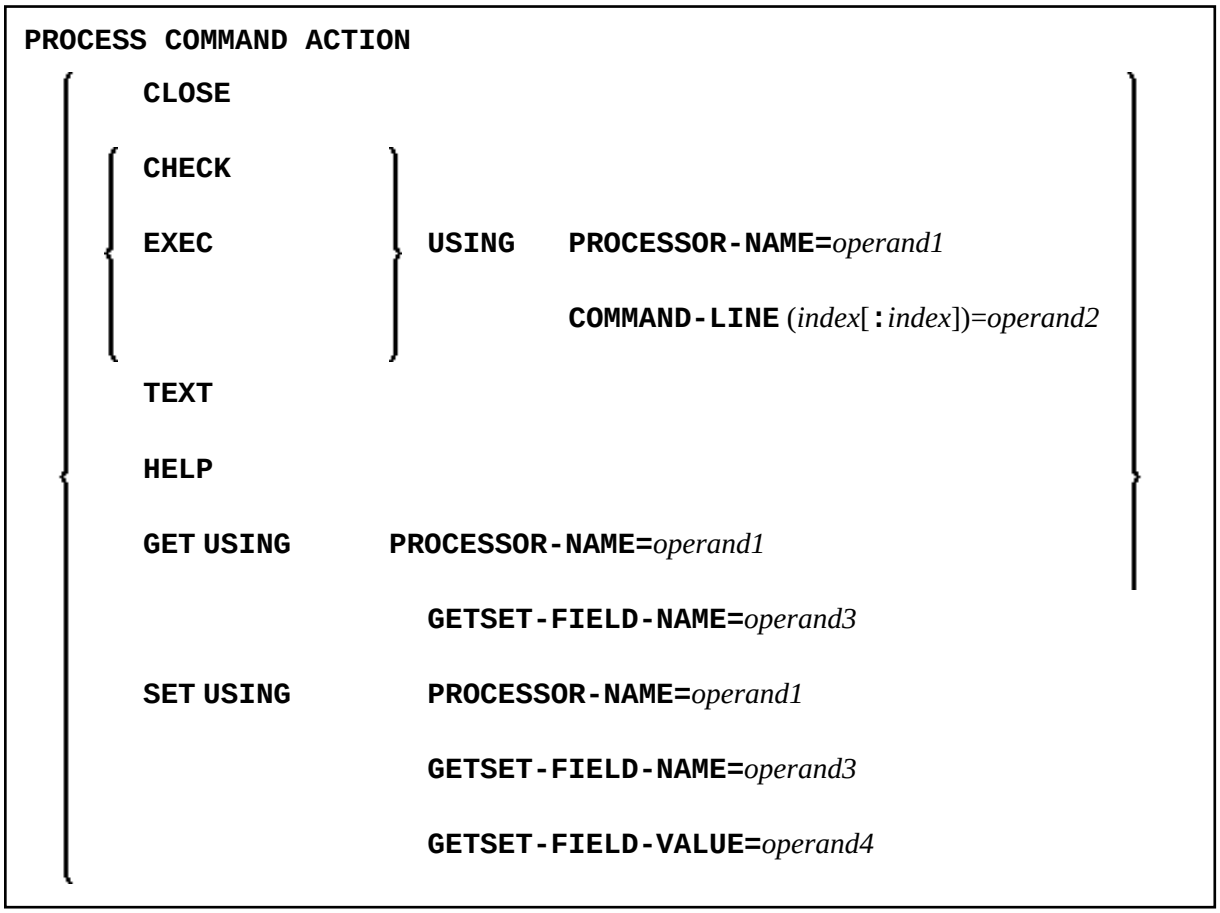
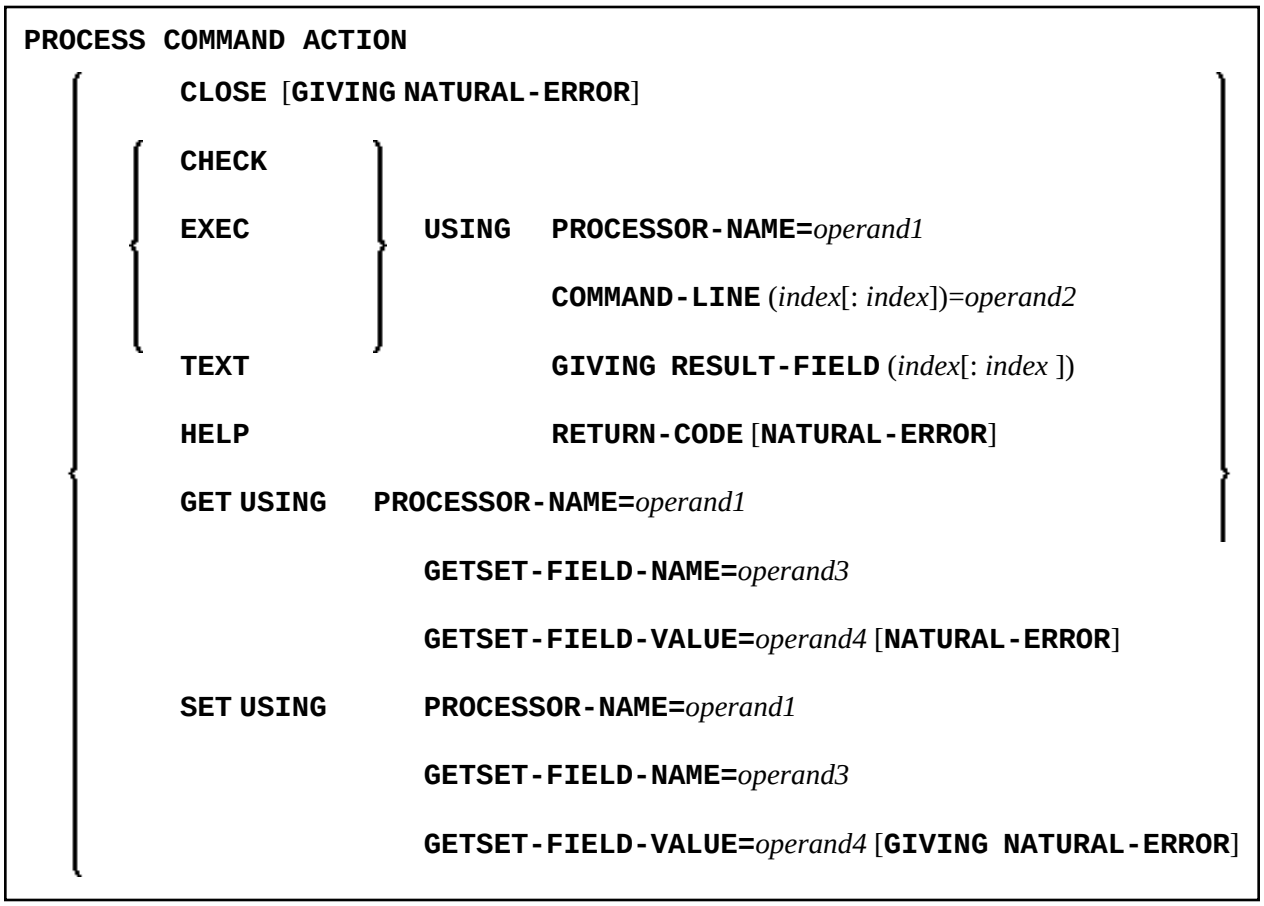
PROCESS COMMAND

Structured Mode Syntax



Reporting Mode Syntax



This chapter covers the following topics:

- Function
- Syntax Description
- DDM: COMMAND
- Examples

For an explanation of the symbols used in the syntax diagram, see *Syntax Symbols*.

Belongs to Function Group: *Invoking Programs and Routines*

Function

Once a Command Processor has been created using the Natural utility SYSNCP, it can be invoked from a Natural program using the PROCESS COMMAND statement.

For details on how to create a Natural Command Processor, please refer to the *SYSNCP Utility* documentation.

Note:
The word COMMAND in the PROCESS COMMAND statement is in fact the name of a view. The name of the view that is used need not necessarily be COMMAND; however, we recommend the use of COMMAND

because there exists a DDM with the same name. This DDM must be referenced within the `DEFINE DATA` statement, for example`COMMAND VIEW OF COMMAND`.

Security Considerations

With Natural Security, it is possible to restrict the usage of certain keywords and/or functions which are defined in a Command Processor. Keywords and/or functions can be allowed/disallowed for a specific user or group of users. See the *Natural Security* documentation for details.

Syntax Description

Operand Definition Table:

Operand	Possible Structure					Possible Formats											Referencing Permitted	Dynamic Definition
operand1	C	S				A											no	no
operand2	C	S	A	G		A		N									no	no
operand3	C	S				A		N									no	no
operand4	C	S				A		N	P	I							no	no

Syntax Element Description:

Syntax Element	Description
`CLOSE`	`CLOSE` terminates the use of the command processor and releases the command processor buffer. When the command processor is used during a session and is not released with `CLOSE`, then there exists a buffer named `NCPWORK` in your thread. The runtime part of the command processor requires this buffer; it can be released using the statement `PROCESS COMMAND ACTION CLOSE`. If any `PROCESS COMMAND` statement follows this statement, then the command processor buffer will be opened again. See also *Example 1 - PROCESS COMMAND ACTION CLOSE*.

Syntax Element	Description
CHECK	CHECK is used as a precautionary measure to determine if a command is executable with the statement PROCESS COMMAND EXEC. It works as follows: for the given processor name, a runtime check is performed in two steps: • It is checked whether the processor exists in the current library or one of its steplibs; • The content of the command line COMMAND-LINE (1) is analyzed to determine whether it is acceptable. In addition, the runtime action definitions R, M and 1-9 are written into RESULT-FIELD (1:9). If the field NATURAL-ERROR is specified in the view or in the GIVING clause, it returns the error code. If this field is not available and the command analysis fails, a Natural system error occurs. **Note:** As the function of the CHECK option is also performed as part of the EXEC option, it is not necessary to use CHECK before every EXEC.
EXEC	EXEC works exactly the same as CHECK with the addition that the runtime actions are executed as specified in the runtime action editor. Only COMMAND-LINE (1) is needed. You can use up to 9 occurrences of RESULT-FIELD (however, for optimum performance, you should not use more occurrences than you really need). **Note:** EXEC is the only option which can be used to leave the currently active program. This is the case when the runtime action definition contains a FETCH or STOP statement. See also *Example 2 - PROCESS COMMAND ACTION EXEC*.

Syntax Element	Description
HELP	HELP returns a list of all valid keywords, synonyms, and functions for the purpose of, for example, the creation of online help windows. This list is contained in the field(s) of RESULT-FIELD. The type of help returned is dependent on the content of the command lines. • COMMAND-LINE (1) must contain the search criteria. • COMMAND-LINE (2), if specified, must contain the start value or a search value. • COMMAND-LINE (3), if specified, must contain a start value. For further information, see the following sections: • *HELP for Keywords* • *HELP for Synonyms* • *HELP for Global Functions* • *HELP for Local Functions* • *HELP for IKN* • *HELP for IFN* **Note:** For optimum performance, the number of occurrences of the field RESULT-FIELD should not exceed the number of lines to be displayed on the screen. At least one occurrence must be used.
TEXT	The TEXT option is used to deliver general information about the processor and text associated with a keyword or function. This text is the same as that entered in the keyword editor or action editor of the SYSNCP utilityduring command processor definition. For further information, see the following sections: • *TEXT for General Information* • *TEXT for Keyword Information* • *TEXT for Function Information* **Note:** To access texts for keywords and functions, you must have specified Y in the field Catalog user texts on the Processor Header Maintenance 3 screen of the SYSNCP utility, see the section *Miscellaneous Options - Header 3*.

HELP for Keywords

This option returns an alphabetically sorted list of keywords and/or synonyms with their internal keyword numbers (IKN).

Command Line	Contents	
1	Must begin with indicator K.	
	The types of keywords to be returned:	
	*	Keywords of all types
	1	Keywords with type 1
	2	Keywords with type 2
	3	Keywords with type 3
	P	Keywords with type P (parameter)
	Options:	
	I	Return IKN in addition to keywords.
	T	Show keyword partially in upper case (to show possible abbreviation).
	S	Return synonyms in addition to keywords.
	X	Return only synonyms of specified keywords.
	A	Internal keywords are also returned.
	+	Search does not include start value.
2	Start value for the keyword search (optional). By default, the search begins with the start value. However, if you specify the plus (+) option, the search does not include the start value itself, but begins with the next higher value.	

The field `RESULT-FIELD (1:n)` returns the specified list.

Examples:

```
Command Line 1:    K*X     Returns all synonyms of all keyword types.

Command Line 1:    K123S   Returns all keywords of type 1, 2 and 3 including synonyms.
```

HELP for Synonyms

For a given IKN, this option returns the original keyword and all synonyms.

Command Line	Contents	
1	Must begin with the indicator S.	
	Option:	
	T	Shows keyword partially in upper case (to show possible abbreviation).
2	Internal Keyword Number (IKN) of the keyword in format N4.	

The field `RESULT-FIELD (1)` returns the original keyword. The fields `RESULT-FIELD (2:`*n*`)` return associated synonyms for this keyword.

Example:

Input:	Output:

`Command Line 1:  S`
`Command Line 2:  1003`
	`Result-Field 1:  Edit`
	`Result-Field 2:  Maintain`
	`Result-Field 3:  Modify`

HELP for Global Functions

This option returns a list of all global functions.

Command Line	Contents	
1	Must begin with the indicator `G`.	
	Options:	
	`I`	Internal Function Number (IFN) is also returned.
	`T`	Shows keyword partially in upper case (to show possible abbreviation).
	`S`	The keywords returned in `RESULT-FIELD` will be aligned in columns.
	`A`	Internal keywords are also returned.
	`1`	Only functions containing the given keyword of type 1 are to be returned.
	`2`	Only functions containing the given keyword of type 2 are to be returned.
	`3`	Only functions containing the given keyword of type 3 are to be returned.
	`+`	Search does not include start value.
2	Start value for global function search. Keywords must be given in sequence `123`. By default, the search begins with the start value. However, if you specify the plus (+) option, the search does not include the start value itself, but begins with the next higher value.	
3	Must be blank.	
4	To search only for global functions with a specific keyword, you specify the keyword here. If you specify a keyword, you also have to specify the keyword type (1, 2 or 3) as option (see above).	

The field `RESULT-FIELD (1:`*n*`)` returns the specified list.

Example:

Input:	Output:

`Command Line 1:  G`
`Command Line 2:  ADD`
	`Result-Field 1:  ADD CUSTOMER`
	`Result-Field 2:  ADD FILE`
	`Result-Field 3:  ADD USER`

HELP for Local Functions

This option returns a list of all local functions for a specified location.

Command Line	Contents	
1	Must begin with the indicator `L`.	
	Options:	
	`I`	Internal Function Number (IFN) is also returned.
	`T`	Shows keyword partially in upper case (to show possible abbreviation).
	`S`	The keywords returned in `RESULT-FIELD` will be aligned in columns.
	`A`	Internal keywords are also returned.
	`1`	Only functions containing given keyword of type 1 are to be returned.
	`2`	Only functions containing given keyword of type 2 are to be returned.
	`3`	Only functions containing given keyword of type 3 are to be returned.
	`C`	Only those functions are returned which are defined for the current location (command line 3 is ignored).
	`F`	Invoke "recursive" listing of local functions; that is, all local commands that lead to the current/specified location will be returned.
2	Start value for local function search (optional). Keywords must be given in sequence `123`.	
3	The location for which the list is to be returned. Keywords must be given in sequence `123`. If no location is specified, the current location of the command processor will be used.	
4	Keyword restriction (optional): If you specify a keyword, or an IKN with the format N4, only functions with this keyword will be returned.	

The field `RESULT-FIELD (1:`*n*`)` returns the specified list.

HELP for IKN

For any given internal keyword numbers (IKN), this option returns the original keyword.

Command Line	Contents	
1	Must start with `IKN`.	
	Options:	
	`A`	The internal keyword will be shown.
	`T`	Shows keyword partially in upper case (to show possible abbreviation).
2	The IKN to be translated, in format N4.	

The field RESULT-FIELD (1) returns the keyword.

Example:

Input:	Output:
Command Line 1: IKN Command Line 2: 0000002002	Result-Field 1: CUSTOMER

HELP for IFN

For any given internal function numbers (IFN), this option returns the keywords of a function.

Command Line	Contents	
1	Must start with IFN.	
	Option:	
	A	Functions with internal keywords will not be suppressed.
2	The IFN to be translated, in format N10.	
3	**Further options:**	
	S	Keywords belonging to the IFN will be returned in RESULT-FIELD (1:3).
	T	Shows keywords partially in upper case (to show possible abbreviations).
	L	IFN will be returned if IFN is used as a location.
	C	IFN will be returned if IFN is used as a command.

The field RESULT-FIELD(1) returns the function; if option S is used, the function is returned in RESULT-FIELD (1:3).

Example:

Input:	Output:
Command Line 1: IFN Command Line 2: 0001048578	Result-Field 1: DISPLAY INVOICE

TEXT for General Information

For general information, COMMAND-LINE (*); that is, all command lines, must be blank. Up to nine fields of RESULT-FIELD are returned containing the following information:

RESULT-FIELD	Contents	Format
1	Header 1 for User Text	Text (A40)
2	Header 2 for User Text	Text (A40)
3	"First Entry used as" text	Text (A16)
4	"Second Entry used as" text	Text (A16)
5	"Third Entry used as" text	Text (A16)
6	Number of Entry 1 Keywords	Numeric (N3)
7	Number of Entry 2 Keywords	Numeric (N3)
8	Number of Entry 3 Keywords	Numeric (N3)
9	Number of Cataloged Functions	Numeric (N7)

TEXT for Keyword Information

For keyword information, `COMMAND-LINE (1)` must contain the corresponding keyword; `COMMAND-LINE (2)` can optionally contain the keyword type (1, 2, 3 or P); `COMMAND-LINE (3:6)` must be empty.

RESULT-FIELD	Contents	Format
1	Keyword comment text	Text (A40)
2	Keyword in full length	Text (A16)
3	Keyword in unique short form	Text (A16)
4	"Keyword used as" entry	Text (A16)
5	Internal keyword number (IKN)	Numeric (N4)
6	Minimum length of keyword	Numeric (N2)
7	Maximum length of keyword	Numeric (N2)
8	Keyword type (1, 2, 3, 1S, 2S, 3S, P)	Text (A2)

TEXT for Function Information

For function information, `COMMAND-LINE (1:3)` must contain the keywords which specify the wanted location. `COMMAND-LINE (4:6)` contains the keywords which specify the wanted function. For example, if information about the global command `ADD USER` is to be returned, the command lines 1, 2, 3, and 6 must be blank; the command line 4 must contain `ADD`, and the command line 5 must contain `USER`.

RESULT-FIELD	Contents	Format
1	Text as defined with the option T in runtime action definition.	Text (A40)
2	Internal function number (IFN) of the specified location.	Numeric (N10)
3	Internal function number (IFN) of the specified function.	Numeric (N10)

GET Option

The GET option is used to read internal command processor information and current command processor settings from the dynamically allocated buffer NCPWORK. The following fields are used:

Field Name	Contents
GETSET-FIELD-NAME (A32)	The name of the variable to be read.
GETSET-FIELD-VALUE (A32)	The value of the specified variable after PROCESS COMMAND ACTION GET is performed.

For a list of possible values for GETSET-FIELD-NAME, see below.

SET Option

The SET option is used to modify internal command processor settings in the buffer NCPWORK.

Field Name	Contents
GETSET-FIELD-NAME (A32)	The name of the variable to be modified.
GETSET-FIELD-VALUE (A32)	The value which is to written to the specified variable.

Possible values for GETSET-FIELD-NAME:

Field Name	Format	G/S*	Content
NAME	A8	G	Name of current processor.
LIBRARY	A8	G	Loaded from library.
FNR	N10	G	Loaded from file.
DBID	N10	G	Loaded from database.
TIMESTMP	A8	G	Time stamp of the current processor.
COUNTER	N10	G	Access counter.
BUFFER-LENGTH	N10	G	Bytes allocated for NCPWORK.
C-DELIMITER	A1	G/S	Multiple command delimiter.
DATA-DELIMITER	A1	G	Delimiter to precede data.
PF-KEY	A1	G/S	PF key may be command (Y/N).
UPPER-CASE	A1	G	Keywords in upper case (Y/N).

Field Name	Format	G/S*	Content
UQ-KEYWORDS	A1	G	Keywords unique (Y/N).
IMPLICIT-KEYWORD	A1	G/S	Identifier for implicit keyword entry.
MIN-LEN	N10	G	Minimum length of keywords.
MAX-LEN	N10	G	Maximum length of keywords.
KEYWORD-SEQ	A8	G/S	Keyword sequence.
ALT-KEYWORD-SEQ	A8	G/S	Alternative keyword sequence.
USER-SEQUENCE	A1	G	User may override KEYWORD-SEQ (Y/N).
CURR-LOCATION	N10	G/S	Current location (IFN).
CURR-IKN1	N10	G/S	IKN1 of current location.
CURR-IKN2	N10	G/S	IKN2 of current location.
CURR-IKN3	N10	G/S	IKN3 of current location.
CHECK-LOCATION	N10	G	Last checked location (IFN).
CHECK-IKN1	N10	G	IKN1 of CHECK-LOCATION.
CHECK-IKN2	N10	G	IKN2 of CHECK-LOCATION.
CHECK-IKN3	N10	G	IKN3 of CHECK-LOCATION.
TOP-IKN1	N10	G	IKN1 of topmost keyword.
TOP-IKN2	N10	G	IKN2 of topmost keyword.
TOP-IKN3	N10	G	IKN3 of topmost keyword.
KEY1-TOTAL	N10	G	Number of keywords of type 1.
KEY2-TOTAL	N10	G	Number of keywords of type 2.
KEY3-TOTAL	N10	G	Number of keywords of type 3.
FUNCTIONS-TOTAL	N10	G	Number of cataloged functions.
LOCAL-GLOBAL-SEQ	A8	G/S	Local/global function validation.
ERROR-HANDLER	A8	G/S	General error program.
SECURITY	A1	G	Natural Security installed (Y/N).
SEC-PREFETCH	A1	G	Natural Security data are to be read (Y/N) or have been read (D = done).
PREFIX1	A1	G	Corresponds to the field Prefix Character 1 on the Processor Header Maintenance 2 screen of the SYSNCP utility, see the section *Keyword Editor Options - Header 2*.
PREFIX2	A1	G	Corresponds to the field Prefix Character 2 on the Processor Header Maintenance 2 screen.
HEX1	A1	G	Corresponds to the field Hex. Replacement 1 on the Processor Header Maintenance 2 screen.

Field Name	Format	G/S*	Content
HEX2	A1	G	Corresponds to the field Hex. Replacement 2 on the Processor Header Maintenance 2 screen.
DYNAMIC	A32	G	Dynamic part (:*n*:) of last error message.
LAST	-	G	Last command placed on top of stack as data.
LAST-ALL	-	G	Last commands placed on top of stack as data.
LAST-COM	-	G	Last command moved to *COM.
MULTI	-	G	Places the last of multiple commands as data on top of the stack.
MULTI-COM	-	G	Places the last of multiple commands in the system variable *COM.

* G = Can be used with the GET Option.

* S = Can be used with the SET Option.

USING Clause

The contents of the fields in the USING clause specify, for example, the processor name and the command line.

Specified in the USING clause are fields to be sent to the command processor.

Option	Field Name			
	PROCESSOR-NAME	COMMAND-LINE	GETSET-FIELD-NAME	GETSET-FIELD-VALUE
CLOSE				
CHECK	mandatory	mandatory		
EXEC	mandatory	mandatory		
TEXT	mandatory	mandatory		
HELP	mandatory	mandatory		
GET	mandatory		mandatory	
SET	mandatory		mandatory	mandatory

GIVING Clause

Note:
This clause can only be used in reporting mode.

Specified in the GIVING clause are fields to be filled by the command processor as a result of the processing of any option.

Option	Field Name			
	NATURAL-ERROR	RETURN-CODE	RESULT-FIELD	GETSET-FIELD-VALUE
CLOSE	recommended			
CHECK	recommended	mandatory	mandatory	
EXEC	recommended	mandatory	mandatory	
TEXT	recommended	mandatory	mandatory	
HELP	recommended	mandatory	mandatory	
GET	recommended			mandatory
SET	recommended			

Note:
The GIVING clause is not available in structured mode, because there exists an implicit GIVING clause made up of all fields specified in the DEFINE DATA statement, which are usually referenced in the GIVING clause for reporting mode. This means that in structured mode all fields that are marked as "mandatory" in the table above must be defined in the DEFINE DATA statement.

DDM: COMMAND

The data definition module (DDM) COMMAND has been created specifically for use in conjunction with the PROCESS COMMAND statement:

```
DB:    1 File:    1 - COMMAND                          Default Sequence: ?

  TYL  DB  NAME                            F LENG  S D   REMARKS
  ---  --  ------------------------------- - ----  - -  ------------------------
    1  AA  PROCESSOR-NAME                  A    8  N D DE    USING
  M 1  AB  COMMAND-LINE                   A   80  N D MU/DE USING
    1  AF  GETSET-FIELD-NAME               A   32  N D DE    USING
    1  BA  NATURAL-ERROR                   N  4.0  N         GIVING
    1  BB  RETURN-CODE                     A    4  N         GIVING
  M 1  BC  RESULT-FIELD                    A   80  N   MU    GIVING
    1  BD  GETSET-FIELD-VALUE              A   32  N D       USING; GIVING
  ***** DDM OUTPUT TERMINATED ******
```

Note:
To avoid possible compilation or runtime errors, please make sure that the DDM named COMMAND is cataloged as type C (field DDM Type on the SYSDDM Menu) before you use it. (If you re-catalog the DDM, any DBID/FNR specification in the SYSDDM utility will be ignored.)

The DDM COMMAND contains the following fields:

DDM Field	Explanation
PROCESSOR-NAME	The name of the command processor for which the PROCESS COMMAND statement is issued. The command processor specified must be cataloged.
COMMAND-LINE	The command line to be processed by the command processor (options CHECK, EXEC), or the keyword/command for which user text or help text is to be returned to the program (options TEXT, HELP). Note that this field may extend beyond one line.
GETSET-FIELD-NAME	This field is used with the options GET and SET and is used to specify the name of a constant or variable which is to be read (GET) or written (SET).
RETURN-CODE	This field contains the return code of an action resulting from the option EXEC or CHECK as specified within a Runtime Actions definition (see the Natural SYSNCP utility).
NATURAL-ERROR	This field is used in conjunction with all options. When the field is used in DEFINE DATA, then it returns the Natural error code for the command processor. When the field is absent, Natural runtime error processing is triggered if an error occurs.
RESULT-FIELD	This field contains information resulting from the use of various options as specified within a runtime action definition (see Runtime Actions in the Natural SYSNCP utility). Please note that this field may be more than one line.
GETSET-FIELD-VALUE	This field is used with the options GET and SET and contains the value of the constant or variable which is specified in the field GETSET-FIELD-NAME.

Examples

- Example 1 - PROCESS COMMAND ACTION CLOSE
- Example - PROCESS COMMAND ACTION EXEC2

Example 1 - PROCESS COMMAND ACTION CLOSE

```
/* EXAM-CLS - Example for PROCESS COMMAND ACTION CLOSE (Structured Mode)
/*********************************************************************
DEFINE DATA LOCAL
  01 COMMAND VIEW OF COMMAND
END-DEFINE
/*
PROCESS COMMAND ACTION CLOSE
/*
DEFINE WINDOW CLS
INPUT WINDOW = 'CLS'
  'NCPWORK has just been released.'
/*
END
```

Example - PROCESS COMMAND ACTION EXEC2

```
/* EXAM-EXS - Example for PROCESS COMMAND ACTION EXEC (Structured Mode)
/*********************************************************************
DEFINE DATA LOCAL
  01 COMMAND VIEW OF COMMAND
     02 PROCESSOR-NAME
     02 COMMAND-LINE (1)
     02 NATURAL-ERROR
     02 RETURN-CODE
     02 RESULT-FIELD (1)
  01 MSG (A65) INIT <'Please enter a command.'>
END-DEFINE
/*
REPEAT
  INPUT (AD=MIT' ' IP=OFF) WITH TEXT MSG
    'Example for PROCESS COMMAND ACTION EXEC (Structured Mode)' (I)
  / 'Command ==>' COMMAND-LINE (1) (AL=64)
  /*******
 PROCESS COMMAND ACTION EXEC
    USING
      PROCESSOR-NAME = 'DEMO'
      COMMAND-LINE (1) = COMMAND-LINE (1)
  /*******
  COMPRESS 'NATURAL-ERROR =' NATURAL-ERROR TO MSG
END-REPEAT
END
```

Note:
You will find other example programs in the library SYSNCP. These programs all begin with EXAM.