

## Natural for Windows

システムコマンド

バージョン 6.3.3

October 2008

This document applies to Natural バージョン 6.3.3 and to all subsequent releases.

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

Copyright © Software AG 1992-2008. All rights reserved.

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

# 目次

1 システムコマンド .....	1
2 システムコマンドの発行 .....	3
コマンド入力 .....	4
コマンド行 .....	4
NEXT プロンプト .....	4
MORE プロンプト .....	5
3 システムコマンド構文 .....	7
構文要素 .....	8
コマンド構文の例 .....	9
4 機能別システムコマンド .....	11
Natural での操作 .....	12
環境設定 .....	12
プログラミングオブジェクトの編集および保存 .....	13
プログラムの実行 .....	14
メンテナンスユーティリティ .....	14
プログラミングオブジェクトの転送 .....	14
モニタリングおよびデバッグ .....	14
NaturalX で使用するコマンド .....	15
その他のコマンド .....	15
5 CATALL .....	17
対話式モードでの CATALL .....	18
バッチモードでの CATALL .....	19
6 CATALOG .....	21
7 CHECK .....	23
8 CLEAR .....	25
9 COMPOPT .....	27
構文説明 .....	28
コンパイラオプション .....	28
コンパイラパラメータの指定 .....	29
リモートメインフレーム環境での COMPOPT .....	30
コンパイラキーワードパラメータの指定 (リモートメインフレーム環境) .....	30
一般的なコンパイラオプション (リモートメインフレーム環境) .....	31
バージョン互換を維持するためのコンパイラオプション (リモートメインフレーム環境) .....	40
10 DEBUG .....	45
11 EDIT .....	47
構文 1 .....	48
構文 2 .....	50
構文 3 .....	50
12 EXECUTE .....	53
構文説明 .....	54
EXECUTE コマンドの例 .....	55
13 FIN .....	57

14 GLOBALS .....	59
構文説明 .....	60
パラメータのリスト .....	60
SET GLOBALS とその他のステートメントとの相互作用 .....	62
15 HELP .....	63
16 INPL .....	65
17 LAST .....	67
18 LASTMSG .....	69
19 LIST .....	71
構文の概要 .....	72
個々のソースの表示 .....	74
オブジェクトのリストの表示 .....	74
ディレクトリ情報の表示 .....	75
ビューの表示 .....	75
リソースオブジェクトのファイル情報の表示 .....	76
エラーメッセージコンテナのファイル情報の表示 .....	76
20 LIST COUNT .....	77
21 LIST XREF .....	79
22 LOGOFF .....	81
23 LOGON .....	83
24 MAIL .....	85
25 MAP .....	87
Natural 開発サーバー環境への接続の確立 .....	88
Natural アプリケーションへの接続の確立 .....	89
26 PROFILE .....	91
27 PURGE .....	93
28 READ .....	95
29 REGISTER .....	97
30 RENAME .....	99
31 RENUMBER .....	101
32 RETURN .....	103
33 RUN .....	105
34 SAVE .....	107
35 SCAN .....	109
36 SCRATCH .....	111
37 SETUP .....	113
構文説明 .....	114
例：SETUP/RETURN .....	115
38 STOW .....	117
39 STRUCT .....	119
ソースコード行のインデント .....	120
40 SYSAPI .....	123
41 SYSCP .....	125
42 SYSERR .....	127
43 SYSEXT .....	129

44 SYSEXV .....	131
45 SYSFILE .....	133
リモートメインフレーム環境での SYSFILE .....	134
46 SYSINST .....	135
47 SYSMAIN .....	137
48 SYSMN .....	139
49 SYSNCP .....	141
50 SYSOBJH .....	143
51 SYSPROD .....	145
52 SYSPROF .....	147
53 SYSRPC .....	149
54 SYSWIZDB .....	151
55 SYSWIZDW .....	153
56 TECH .....	155
57 UNCATALOG .....	157
58 UNLOCK .....	159
Natural オブジェクトのロック解除 .....	160
ドキュメンテーションオブジェクトのロック解除 .....	161
パラメータの説明 .....	161
パラメータ処理と見つかったオブジェクトの表示 .....	163
59 UNMAP .....	165
現在アクティブな環境／アプリケーションのアンマップ .....	166
Natural 開発サーバー環境のアンマップ .....	166
Natural アプリケーションのアンマップ .....	166
60 UNREGISTER .....	167
61 UPDATE .....	169
62 XREF .....	171
索引 .....	173

---

# 1 システムコマンド

---

このドキュメントでは Natural システムコマンドについて説明します。

Natural システムコマンドは、Natural プログラミングオブジェクトの作成、管理、または実行に必要な機能を実行します。また、Natural システムコマンドを使用して、Natural 環境をモニタしたり管理したりすることもできます。

このドキュメントは次の項目で構成されています。

 システムコマンドの発行	Natural システムコマンドの入力時に適用される一般的なルールについて説明します。
 システムコマンド構文	Natural システムコマンドの構文記述に使用される記号について説明します。
 機能別システムコマンド	機能に応じて分類された Natural システムコマンドの概要について説明します。
 システムコマンド一覧（アルファベット順）	システムコマンド（アルファベット順）の説明です。



## 2 システムコマンドの発行

---

▪ コマンド入力 .....	4
▪ コマンド行 .....	4
▪ NEXT プロンプト .....	4
▪ MORE プロンプト .....	5

このchapterでは、次のトピックについて説明します。

## コマンド入力

---

システムコマンドを実行するには、次のいずれかの方法でコマンドを入力します。

- **コマンド行**で入力します。
- Natural **NEXT** または **MORE** プロンプトで入力します。

次の規則が適用されます。

- コマンド入力では、大文字と小文字は区別されません。
- コマンドはコンテキスト依存です。
- 一部のNaturalコマンドは、現在アクティブなオブジェクト以外のオブジェクトにも影響を与えます。

構文の記述で使用される記号の詳細については、「[システムコマンド構文](#)」を参照してください。

## コマンド行

---

システムコマンドの機能は、さまざまなメニューから使用できます。また、システムコマンドは、コマンド行に入力することもできます。『*Natural* スタジオの使用』の「[コマンド行でのコマンドの発行](#)」を参照してください。

## NEXT プロンプト

---

NEXT プロンプトは、保留になっている出力が残っていない場合に Natural アプリケーションやプログラムで表示されます。

## MORE プロンプト

---

MORE プロンプトは、保留になっている出力がまだあることを示すために出力画面の下段に表示されます。MORE プロンプトに応答してシステムコマンドを入力すると、プログラムの実行は中断されて、システムコマンドが実行されます。

---

# 3 システムコマンド構文

---

- 構文要素 ..... 8
- コマンド構文の例 ..... 9

このchapterでは、次のトピックについて説明します。

### 構文要素

システムコマンドの構文の記述には、次の記号を使用します。

要素	説明
ABCDEF	大文字の用語は Natural キーワードまたは Natural 予約語のいずれかで、示されたとおりに入力する必要があります。
<u>ABCDEF</u>	大文字のオプション用語全体に下線が付いている（ハイパーリンクではない）場合は、その用語がデフォルト値であることを示します。用語を省略すると、下線の付いた値が適用されます。
<u>ABCDEF</u>	大文字の用語の一部に下線が付いている（ハイパーリンクではない）場合は、下線部分はその用語の入力可能な省略形です。
<i>abcdef</i>	斜体の文字は、変数情報を表します。この用語を指定する場合は、有効な値を入力する必要があります。
[ ]	角カッコ内の要素はオプションです。  角カッコに複数の行が入れ子になって含まれている場合は、各行がオプションの選択肢です。選択肢の中から1つ選択できます。
{ }	中カッコに複数の行が入れ子になって含まれている場合は、各行が選択肢です。選択肢の中から1つのみ選択する必要があります。
	選択肢は縦棒で区切られます。
...	省略記号の前の用語は任意に繰り返すことができます。省略記号の後の数値は用語の繰り返し回数を示します。  省略記号の前の用語が大カッコまたは中カッコで囲まれている場合、省略記号はカッコ全体に適用されます。
,...	コンマと省略記号の前の用語は任意に繰り返すことができます。繰り返す場合は、繰り返しをコンマで区切る必要があります。コンマと省略記号の後の数値は用語の繰り返し回数を示します。  コンマと省略記号の前の用語が大カッコまたは中カッコで囲まれている場合、コンマと省略記号はカッコ全体に適用されます。
:...	コロンと省略記号の前の用語は任意に繰り返すことができます。繰り返す場合は、繰り返しをコロンので区切る必要があります。コロンと省略記号の後の数値は用語の繰り返し回数を示します。  コロンと省略記号の前の用語が大カッコまたは中カッコで囲まれている場合、コロンと省略記号はカッコ全体に適用されます。
その他の記号 ([ ] { }   ... , ... : ... を除く)	他のすべての記号は、この表で定義しているものを除いて、示されたとおりに入力する必要があります。

要素	説明
	例外： SQL スカラ連結演算子は2つの縦線で表現され、構文定義に記述されているとおりに完全に入力する必要があります。

## コマンド構文の例

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

- CATALOG は Natural キーワードであり、指定する場合は正しく入力する必要があります。下線部分 (CAT) は省略形として入力できることを示しています。
- *object-name* および *library-id* はユーザーが設定するオペランドであり、処理するプログラムの名前とそのプログラムが存在するライブラリ ID を指定します。
- 大カッコは、*object-name* および *library-id* がオプションであることを示します。必要に応じて指定してください。カッコによるグループ化は、CATALOG だけを指定することも、あるいは、CATALOG の後にプログラム名だけ、またはプログラム名とライブラリ ID を指定することも可能であることを示しています。ただし、プログラム名を指定せずにライブラリ ID を指定することはできません。



# 4 機能別システムコマンド

---

▪ Natural での操作 .....	12
▪ 環境設定 .....	12
▪ プログラミングオブジェクトの編集および保存 .....	13
▪ プログラムの実行 .....	14
▪ メンテナンスユーティリティ .....	14
▪ プログラミングオブジェクトの転送 .....	14
▪ モニタリングおよびデバッグ .....	14
▪ NaturalX で使用するコマンド .....	15
▪ その他のコマンド .....	15

このchapterでは、機能に応じて分類された Natural システムコマンドの概要について説明します。

### Natural での操作

---

コマンド	機能
FIN	Natural セッションを終了します。
LOGOFF	ライブラリ ID に SYSTEM を設定し、Adabas パスワードに空白を設定します。ソースプログラムワークエリアの内容は、このコマンドの影響を受けません。
LOGON	ユーザーにライブラリ ID を設定します。セッション中に保存した全ソースまたはオブジェクトプログラムが、指定したライブラリに保存されます (SAVE、CATALOG、または STOW コマンドに別のライブラリ ID を明示的に指定する場合を除く)。
RETURN	SETUP コマンドで設定した戻り位置に戻ります。
SETUP	RETURN コマンドを使用して制御を戻すことができる戻り位置を設定します。これにより、Natural セッション中にアプリケーション間を容易に移動できます。

### 環境設定

---

コマンド	機能
COMPOPT	Natural プログラミングオブジェクトをコンパイルするときの各種コンパイルオプションを設定します。
GLOBAL S	Natural セッションパラメータの設定を変更します。
MAP	リモート開発サーバーへの接続を設定します。
PROFILE	Natural Security をインストールしている場合にのみ使用できます。 現在有効なセキュリティプロファイルを表示します。このプロファイルは、現在の Natural 環境の実際の使用条件を知らせます。
SYSPROD	インストールされている製品のリストとその製品に関する情報を表示します。
SYSPROF	Natural システムファイルの現在の定義を表示します。
UNMAP	現在アクティブなりモート環境を切断します。

## プログラミングオブジェクトの編集および保存

コマンド	機能
CATALL	現在のライブラリのすべてのオブジェクトまたは選択したオブジェクトをカタログします。
CATALOG	現在、エディタのソースワークエリアにある Natural プログラミングオブジェクトをコンパイルし、構文が正しければ、作成されたオブジェクトモジュールを Natural システムファイルに保存します。
CHECK	プログラミングオブジェクトのソースコードに構文エラーが含まれていないことをチェックします。チェック処理はチェックするオブジェクトのタイプによって変わります。  構文チェックは、システムコマンド RUN、CATALL、CATALOG、および STOW の一部としても実行されます。
CLEAR	現在アクティブなオブジェクトを閉じて、内容も名前もない新規のエディタウィンドウを開きます。エディタのタイプは、現在アクティブなオブジェクトに対応します。現在アクティブなオブジェクトを変更し、まだ保存していない場合は、変更の保存を促すプロンプトが表示されます。
EDIT	プログラミングオブジェクトのソースを編集します。
LIST	現在のライブラリに含まれている 1 つ以上のオブジェクトをリストします。
READ	Natural システムファイルからソースワークエリアに、ソース形式のオブジェクトを読み込みます。
RENUMBER	現在ソースワークエリアにあるソースコードの番号を変更します。
SAVE	現在エディタのソースワークエリアにあるプログラミングオブジェクトを、ソース形式で Natural システムファイルに保存します。
SCAN	オブジェクト内の文字列を検索します。その際に、オプションとして文字列を置換することもできます。
STOW	Natural システムファイルにオブジェクトを（ソース形式とオブジェクト形式で）保存します。
SYSWIZDW	Natural Dialog Wizard を呼び出します。これは、特定用途のためのダイアログを作成するツールです。定義済みダイアログボックスに、目的に適合する複数のレイアウトを設定できます。

## プログラムの実行

コマンド	機能
EXECUTE	コンパイル後にオブジェクト形式で保存されたプログラムを実行します。コンパイルされた形式で保存されたプログラムだけを実行 (EXECUTE) できます。
RUN	現在エディタのワークエリアにあるソースプログラムを、コンパイルして実行します。

## メンテナンスユーティリティ

コマンド	機能
SYSERR	Natural アプリケーションでユーザーに対して表示するメッセージを作成およびメンテナンスします。
SYSNCP	Natural アプリケーションで使用するコマンドプロセッサを作成およびメンテナンスします。
SYSRPC	リモートプロシージャコールを作成およびメンテナンスします。つまり、リモートサーバーにあるサブプログラムを実行するために必要な設定を提供します。

## プログラミングオブジェクトの転送

コマンド	機能
SYSMAIN	Natural システムファイル内のオブジェクトを、ライブラリ間で転送します。
SYSOBJH	Natural 環境での分散のために、Natural と Natural 以外のオブジェクトを処理します。

## モニタリングおよびデバッグ

コマンド	機能
RPCERR	リモートプロシージャコール (RPC) に関連している場合は、最後の Natural エラー番号とメッセージを表します。また、最後の Broker 理由コードと関連メッセージを表示します。
TECH	Natural セッションの技術的な情報およびその他の情報を表示します。

## NaturalX で使用するコマンド

コマンド	機能
REGISTER	Natural クラスを登録します。それらは、Natural を起動したサーバー ID に登録されます。
UNREGISTER	Natural クラスの登録を解除します。

## その他のコマンド

コマンド	機能
HELP	Natural ヘルプシステムを呼び出します。
INPL	INPL ユーティリティを呼び出します。Software AG インストールデータセットをシステムファイルにロードするためだけに使用します。
LAST	最後に実行されたシステムコマンドを表示し、再度実行することができます。
LASTMSG	最後に発生したエラー状況に関する追加情報を表示します。
MAIL	Natural Security をインストールしている場合にのみ使用できます。  メールボックスを呼び出し、その内容と満了日のいずれかまたは両方を変更します。メールボックスは、Natural ユーザーにメッセージを送る掲示板です。
SYSEXT	SYSEXT ライブラリを呼び出します。このライブラリには、Natural ユーザーアプリケーションインターフェイスが含まれています。
SYSEXV	現在の Natural バージョンで採用された新機能の例が含まれている SYSEXV アプリケーションを呼び出します。
SYSFILE	SYSFILE ユーティリティの Natural 出力/ワークファイル機能を呼び出します。使用可能なワークファイルおよび出力ファイルに関する情報が表示されます。
SYSWIZDB	Natural データブラウザを呼び出します。このブラウザは、Natural スタジオ内の開発ツールウィザードです。これにより、ファイル構造を表示、出力、または保存することができます。
UNLOCK	ロックされているオブジェクトを表示し、必要に応じてアンロックできるようにします。
UPDATE	プログラムによるデータベース更新の実行を阻止します。
XREF	Predict がインストールされている場合にのみ使用できます。  Predict の「アクティブクロスリファレンス」機能の使用を制御します。プログラムまたはデータエリアが参照しているオブジェクトについてのドキュメントを、Predict に自動で作成します。



# 5 CATALL

---

- 対話式モードでの CATALL ..... 18
- バッチモードでの CATALL ..... 19

このコマンドは、現在のライブラリ内のすべてのオブジェクトまたは選択したオブジェクトをカタログ、チェック、保存、または格納するために使用します。

次のトピックについて説明します。

## 対話式モードでの CATALL

### CATALL

このコマンドを発行すると、[カタログオブジェクト (ライブラリ: ...)] ダイアログボックスが表示されます。このダイアログボックスで、処理するオブジェクトのタイプを指定します。オブジェクトは、ダイアログボックスにリストされているオブジェクトタイプの順番で処理されます。さらに、実行するアクションと処理対象のオブジェクトを選択できます。

このダイアログボックスで指定できる項目は、次のとおりです。

開始する名前	現在のライブラリで選択されているタイプのすべてのオブジェクトを処理する場合は、アスタリスク (*) を入力します。  オブジェクトの番号を特定の範囲に制限する場合は、名前にアスタリスク表記を使用できます。
存在するモジュールのみに適応	このオプションをマークすると、現在のライブラリにオブジェクトモジュールがすでに存在するオブジェクトのみが再カタログされます。ソース形式のみで存在するオブジェクトは処理されません。
全てのソースに適応	このオプションをマークすると、選択したすべてのオブジェクトが処理されます。
アクション	ここでは、選択したオブジェクトに適用するアクションを次の中から 1 つ選択できます。  <ul style="list-style-type: none"> <li>■ CATALOG</li> <li>■ CHECK</li> <li>■ SAVE</li> <li>■ STOW</li> </ul> これらのアクションは、同名のシステムコマンドに対応します。  <b>注意:</b> Natural Security では、特定のアクションが許可されない場合があります。
行番号再設定	デフォルトでは、保存または格納されたソースのソースコード行の番号も再設定されます。  行番号の自動再設定を望まない場合は、このチェックボックスをオフにします。
オブジェクトタイプ	デフォルトでは、CATALL は現在のライブラリのあらゆるタイプのオブジェクトに適用されます。つまり、すべてのオブジェクトタイプが有効になっています。特定の

	<p>タイプのオブジェクトがCATALLで処理されないようにするには、該当するオプションを無効にします。</p> <p>また、すべてのオプションを選択したり、すべてのチェックボックスをオフにできるコマンドボタンがあります。</p> <p><b>注意:</b> リモートのメインフレーム開発環境でSPoDを使用して作業している場合は、<b>[DDM]</b> および <b>[マップソース変換 (Generate new map source)]</b> オプションは使用できず、グレー表示になります。</p>
マップソース変換	<p>Natural の以前のバージョンで作成されたマップは、Natural バージョン 3.1 以降と必ずしも互換性がありません。このオプションをマークすると、カタログ処理中にマップが確実に変換されます。このオプションによって、ソース形式とオブジェクト形式の両方のマップが変換され、保存されます。</p> <p>リモートのメインフレーム開発環境でSPoDを使用して作業している場合は、このオプションは使用できず、グレー表示になります。</p>

詳細については、『Natural スタジオの使用』の「ライブラリ内のオブジェクトのカタログ化」を参照してください。

## バッチモードでの CATALL

$\text{CATALL } object\text{-name} \left[ \begin{array}{c} \text{RECAT} \\ \text{ALL} \end{array} \right] [\text{TYPES } types] \left[ \begin{array}{c} \text{CATALOG} \\ \text{CHECK} \\ \text{SAVE} \\ \text{STOW} \end{array} \right] [options \dots]$
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

[カタログオブジェクト (ライブラリ: ...)] ダイアログボックスで実行できるさまざまな指定に対応し、システムコマンド CATALL で直接指定できる次のようなオプションもあります。

<i>object-name</i>	<p>カタログされるオブジェクトの名前。</p> <p>現在のライブラリで指定されているタイプのすべてのオブジェクトをカタログする場合は、アスタリスク (*) を入力します。</p> <p>オブジェクトの番号を特定の範囲に制限する場合は、名前にアスタリスク表記を使用できます。</p>
RECAT/ALL	<p>[カタログオブジェクト (ライブラリ: ...)] ダイアログボックスの [存在するモジュールのみに適応 (Apply action only to existing modules)] オプションまたは [全てのソースに適応 (Apply action to all sources)] オプションに対応します。RECAT はデフォルトです。</p>

<p><b>TYPES</b> <i>types</i></p>	<p>[カタログオブジェクト (ライブラリ：...)] ダイアログボックスでマークされるオブジェクトタイプに対応します。可能な <i>types</i> :</p> <p>G グローバルデータエリア  A パラメータデータエリア  L ローカルデータエリア  D DDM  S サブルーチン  N サブプログラム  H ヘルプルーチン  M マップ  P プログラム  3 ダイアログ  4 クラス  7 ファンクション  8 アダプタ  * すべてのタイプ (デフォルト)</p> <p><i>types</i> は、1つの文字列 (ローカルに対する LAG など)、パラメータ、およびグローバルデータエリアで指定する必要があります。デフォルトでは、CATALL は現在のライブラリ内にあるすべてのタイプのオブジェクトに適用されます。</p>	
<p><b>CATALOG/CHECK/SAVE/STOW</b></p>	<p>[カタログオブジェクト (ライブラリ：...)] ダイアログボックスの同名のアクションに対応します。CATALOG はデフォルトです。</p>	
<p><i>options</i></p>	<p>NOREN</p>	<p>ソースオブジェクトのソースコード行の自動番号再設定は行われません。</p>

 **Note:** 個々のコマンドコンポーネントは、空白または INPUT 区切り文字 (セッションパラメータ ID で定義) で区切る必要があります。

## 6 CATALOG

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

関連コマンド： [SAVE](#) | [STOW](#) | [UNCATALOG](#)

このコマンドは、現在、エディタのソースワークエリアにある Natural プログラミングオブジェクトをコンパイルし、（構文が正しければ）作成されたオブジェクトモジュールを Natural システムファイルに保存するために使用します。

『Natural スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

 **Important:** プロファイルパラメータ RECAT が ON に設定されている場合は、CATALOG コマンドを使用できません。この場合は [STOW](#) コマンドを使用して、オブジェクトをコンパイルおよび保存してください。

<b>CATALOG</b>	<i>object-name</i> を指定しない場合、オブジェクトは、EDIT または READ などを使用してソースワークエリアに最後に読み込まれたオブジェクトの名前でライブラリにカタログされます。
<b>CATALOG</b> <i>object-name</i>	新しいオブジェクトが作成されます。 <i>object-name</i> には、新しいオブジェクトがカタログされる時の名前を指定します。オブジェクトは現在のライブラリに保存されます。オブジェクトが存在する場合、コマンドは拒否されます。
<b>CATALOG</b> <i>object-name</i> <i>library-id</i>	新しいオブジェクトを別のライブラリにカタログするには、そのライブラリの <i>library-id</i> を指定する必要があります。オブジェクトが存在する場合、コマンドは拒否されます。

 **Note:** 有効ではないパラメータファイルに FDIC システムファイルが指定された場合は、CATALOG コマンドが発行されたときに、Natural によって適切なエラーメッセージが表示されます。

『Natural スタジオの使用』ドキュメントの「オブジェクトのカタログ化」も参照してください。



# 7 CHECK

---

## CHECK

このコマンドは、現在エディタワークエリアにあるソースコードの構文にエラーが含まれていないかどうかをチェックするために使用します。

構文エラーが検出されると、構文チェックは中断され、エラーが含まれる行が表示されます。ここでは、行を修正する（その後整合性チェックが継続します）か、または表示されている行を修正しないでEnterキーを押すことができます。この場合は整合性チェック手順が停止し、エディタが開きます。

 **Note:** 構文チェックは、[RUN](#)、[STOW](#)、[CATALOG](#)、および [CATALL](#) の各コマンドの一部としても実行されます。

『*Natural* スタジオの使用』ドキュメントの「オブジェクトのチェック」も参照してください。



# 8

## CLEAR

---

CLEAR

このコマンドは、現在アクティブなオブジェクトを閉じて、内容や名前のない新しいエディタウィンドウを開くために使用します。エディタのタイプは、現在アクティブなオブジェクトに対応します。

現在アクティブなオブジェクトを変更し、まだ保存していない場合は、変更の保存を促すプロンプトが表示されます。

『*Natural* スタジオの使用』ドキュメントの「エディタウィンドウのクリア」も参照してください。



## 9 COMPOPT

---

▪ 構文説明 .....	28
▪ コンパイラオプション .....	28
▪ コンパイラパラメータの指定 .....	29
▪ リモートメインフレーム環境での COMPOPT .....	30
▪ コンパイラキーワードパラメータの指定 (リモートメインフレーム環境) .....	30
▪ 一般的なコンパイラオプション (リモートメインフレーム環境) .....	31
▪ バージョン互換を維持するためのコンパイラオプション (リモートメインフレーム環境) .....	40

COMPOPT [*option=value ...*]

このシステムコマンドは、各種のコンパイラオプションを設定するために使用します。設定したオプションは、Natural プログラミングオブジェクトのコンパイル時に評価されます。

オプションなしで COMPOPT コマンドを入力すると、画面が表示されます。この画面では、以降で説明するオプションを有効または無効にできます。

個々のオプションのデフォルト設定は、Naturalパラメータファイル内の対応するプロファイルパラメータを使用して設定されます。

このchapterでは、次のトピックについて説明します。

## 構文説明

COMPOPT	<p>オプションなしで COMPOPT システムコマンドを発行すると、ダイアログボックスが表示されます。ここで使用可能なキーワードについては、次で説明します。</p> <p>『Natural スタジオの使用』ドキュメントの「コンパイラオプション」も参照してください。</p>
COMPOPT <i>option=value</i>	<p>ダイアログボックスでオプションを変更する代わりに、COMPOPT コマンドを使用して直接指定することもできます。</p> <p>例：</p> <pre>COMPOPT DBSHORT=ON</pre>

## コンパイラオプション

次のコンパイラオプションを使用できます。各オプションの目的および可能な設定の詳細については、対応する Natural プロファイルパラメータの解説を参照してください。

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

## コンパイラパラメータの指定

コンパイラパラメータは、次のような異なるレベルで指定できます。

### 1. デフォルト設定として指定

個々のコンパイラパラメータのデフォルト設定は、コンフィグレーションユーティリティの [コンパイラオプション] カテゴリを使用して指定し、Naturalパラメータファイル NATPARM に保存されます。

### 2. セッション開始時に指定

セッション開始時に、対応するプロファイルパラメータを指定することによって、コンパイラオプション設定を上書きできます。

### 3. アクティブな **Natural** セッション中に指定

アクティブな **Natural** セッション中に COMPOPT システムコマンドを使用してコンパイラパラメータの値を変更するには、次の2つの方法があります。コマンド割り当て (COMPOPT *optionvalue*) を使用して直接行う方法、およびオプションなしで COMPOPT コマンドを発行して= [コンパイラオプション] ダイアログボックスを表示する方法です。コンパイラオプションに割り当てられた設定は、次の LOGON コマンドを別のライブラリに発行するまで有効です。別のライブラリへの LOGON 時には、デフォルト設定 (上記の項目 1 を参照) に戻ります。次に例を示します。

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

### 4. **Natural** プログラミングオブジェクトに指定

プログラムやサブプログラムなどの **Natural** プログラミングオブジェクトに、OPTIONS ステートメントを使用してコンパイラパラメータを設定できます。次に例を示します。

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

OPTIONS ステートメントで定義したコンパイラオプションは、このプログラミングオブジェクトのコンパイルにのみ影響し、COMPOPT コマンドで設定した一連の設定を更新することはありません。

## リモートメインフレーム環境での COMPOPT

次に示すトピックは、COMPOPT コマンドをリモートメインフレーム環境で使用する場合にのみ適用されます。

### コンパイラキーワードパラメータの指定（リモートメインフレーム環境）

コンパイラキーワードパラメータは、次のような異なるレベルで指定できます。

1. 個々のキーワードパラメータのデフォルト設定は、Natural パラメータモジュール NATPARM の NTCMPO マクロに指定します。
2. セッション開始時に、プロファイルパラメータ CMPO でコンパイラキーワードパラメータを上書きできます。
3. アクティブな Natural セッション中に COMPOPT システムコマンドを使用してコンパイラキーワードパラメータを変更するには2つの方法があります。コマンド割り当て (COMPOPT=*optionvalue*) を使用して直接行う方法、およびキーワードパラメータなしで COMPOPT コマンドを発行して [Compilation Options] 画面を表示する方法です。コンパイラオプションに割り当てられた設定は、次の LOGON コマンドを別のライブラリに発行するまで有効です。LOGON 時には、NTCMPO マクロおよびプロファイルパラメータ CMPO（上記参照）のいずれかあるいはその両方で設定されたデフォルト設定に戻ります。次に例を示します。

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

4. プログラムやサブプログラムなどの Natural プログラミングオブジェクトに、OPTIONS ステートメントを使用してコンパイラパラメータ（オプション）を設定できます。次に例を示します。

```

OPTIONS KCHECK=ON
WRITE 'Hello World'
END

```

OPTIONS ステートメントで定義したコンパイラオプションは、このプログラミングオブジェクトのコンパイルにのみ影響し、COMPOPT コマンドで設定した一連の設定を更新することはありません。

## 一般的なコンパイラオプション（リモートメインフレーム環境）

次のオプションを使用できます。

- KCHECK - キーワードチェック
- PCHECK - CALLNAT ステートメントのパラメータチェック
- DBSHORT - データベースショートフィールド名の解釈
- PSIGNF - パック十進数の正記号の内部表現
- TSENABL - TS プロファイルパラメータの適用性
- GFID - グローバルフォーマット ID の生成
- LOWSRCE - 小文字ソースの許可
- TQMARK - 引用符の変換
- THSEP - ダイナミック千桁単位セパレータ
- CPAGE - 英数字定数に対するコードページのサポート
- DB2ARRY - SQL ステートメント SELECT および INSERT での DB2 配列のサポート

これらのオプションは、CMPPO プロファイルパラメータと NTCMPO パラメータマクロのいずれかまたは両方のキーワードサブパラメータに対応します。

### KCHECK - キーワードチェック

<b>ON</b>	プログラミングオブジェクトでのフィールド宣言が、重要な Natural キーワードの集合に対してチェックされます。定義された変数名がこれらのキーワードのいずれかに一致する場合、プログラミングオブジェクトがチェックまたはカタログされたときに構文エラーが報告されます。
<b>OFF</b>	キーワードチェックは実行されません。これはデフォルト値です。

『プログラミングガイド』の「Natural 予約キーワードのチェックの実行」セクションには、KCHECK オプションでチェックされるキーワードのリストが記載されています。

『プログラミングガイド』の「Natural 予約キーワードのアルファベット順リスト」セクションには、Natural のすべてのキーワードおよび予約語の概要が記載されています。

## PCHECK - CALLNAT ステートメントのパラメータチェック

<b>ON</b>	<p>コンパイラが、CALLNAT ステートメントで指定されたパラメータの数値、フォーマット、長さ、および配列インデックスの限度をチェックします。また、DEFINE DATA PARAMETER ステートメントの OPTIONAL 機能もパラメータチェックで考慮されます。</p> <p>パラメータチェックは、CALLNAT パラメータと、呼び出されたサブプログラムの DEFINE DATA PARAMETER 定義との比較に基づいて行われます。</p> <p>このチェックを行うには、次の条件が必須です。</p> <ul style="list-style-type: none"> <li>■ 呼び出されるサブプログラム名が、英数字の変数としてではなく英数字の定数として定義されている。</li> <li>■ 呼び出されるサブプログラムが、カタログ化オブジェクトとして利用可能である。</li> </ul> <p>これに該当しない場合、PCHECK=ON には何の効力もありません。</p> <p><b>CATALL コマンドを PCHECK=ON で使用する場合の問題点</b></p> <p>CATALL コマンドを PCHECK=ON に指定して使用するときは、次の点を考慮する必要があります。</p> <p>CATALL プロセスが開始すると、プログラミングオブジェクトがコンパイルされる順序は、まずオブジェクトのタイプが優先され、次にオブジェクトのアルファベット名で決定されます。使用されるオブジェクトタイプの順序は、GDA、LDA、PDA、外部サブルーチン、サブプログラム、ヘルプルーチン、マップ、クラス、そして最後にプログラムとなります。同じタイプのオブジェクトでは、名前のアルファベット順によって、オブジェクトがカタログされる順序が決定されます。</p> <p>上述したように、CALLNAT ステートメントは、呼び出されたサブプログラムのコンパイル後の形に対してチェックされます。呼び出し側のオブジェクト（現在はコンパイルが済み、CALLNAT ステートメントが含まれるもの）がヘルプルーチン、マップ、またはプログラムである場合、問題は発生しません。CALLNAT ステートメントで参照されるサブプログラムがすでにカタログされているため、そのオブジェクト形式での最新のパラメータレイアウトで使用できるためです。</p> <p>ただし、呼び出し側のオブジェクトが外部サブルーチンであるか、または CALLNAT ステートメントで呼び出されたサブプログラムよりもアルファベット順で先になる名前を持つサブプログラムである場合に、CALLNAT ステートメントおよび呼び出されたサブプログラムのパラメータが変更されると、PCHECKの結果が正しくないものになる可能性があります。つまり、パラメータが適切に一致しているのに構文エラー NAT0648/NAT0651 が発生するか、またはパラメータが一致していないのに構文エラーが発生しない場合があります。この場合、呼び出されたサブプログラムの新しいオブジェクトイメージは、まだ CATALL によって作成されていません。カタログ化の順序が現在コンパイルされているオブジェクトよりも後になるためです。これにより、CALLNAT ステートメントの新しいパラメータレイアウトが、呼び出されたサブプログラムの DEFINE DATA PARAMETER ステートメントの古いパラメータレイアウトと比較されることとなります。</p> <p>解決策：</p> <ul style="list-style-type: none"> <li>■ コンパイラオプション PCHECK=OFF を設定します。</li> <li>■ 変更されたパラメータの数に応じて次のように処理します。 <ul style="list-style-type: none"> <li>■ 1つまたは少数のサブプログラムのパラメータが変更された場合は、これらのオブジェクトに対して別々にコンパイルを実行します。</li> </ul> </li> </ul>
-----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

	<ul style="list-style-type: none"> <li>■ 大量のサブプログラムのパラメータが変更された場合は、タイプが "サブプログラム" であるオブジェクトのみを選択して CATALL を実行します。</li> <li>■ コンパイラオプション PCHECK=ON を設定します。</li> <li>■ ライブラリ全体に対し、CATALL で全体コンパイルを実行します。</li> </ul>
OFF	パラメータチェックは実行されません。これはデフォルト値です。

## DBSHORT - データベースショートフィールド名の解釈

ON	<p>プログラミングオブジェクト内のデータベースフィールド名は、対応する DDM に定義されるようなロングネームとみなされますが、例外として、2文字のフィールド名は、基礎となるデータベースシステムで使用されるようなショートネームとみなされます。これはデフォルト値です。</p> <p>たとえ DBSHORT=ON に設定しても、次の場合には、データベースショートフィールド名は使用できません。</p> <ul style="list-style-type: none"> <li>■ プログラムに DEFINE DATA LOCAL が指定されている場合</li> <li>■ メインフレームプラットフォーム向けではない Natural の場合</li> <li>■ Natural Security がインストールされている場合</li> </ul>
OFF	プログラミングオブジェクト内のすべてのデータベースフィールド名は、長さに関係なくロングネームとみなされます。これにより、プログラムのデータベースフィールド名が誤って解釈される可能性を回避できます。この設定を使用して、ショートネームの使用を全体的に禁止することができます。

### 予備知識 (DBSHORT=ON)

Natural コンパイラによってデータベースフィールド (DDM に定義されているフィールド) と DBSHORT=ON が解決されるときは、フィールド名の長さを使用して、識別子が「db-short-name」と「db-long-name」のどちらを表すかが判断されます。フィールド名の長さが2文字のときは、「db-short-name」参照が想定されますが、それ以外のすべての識別子は「db-long-names」として処理されます。

### 常に適用される Natural 規則

一般的な Natural 規則に従うと、DEFINE DATA LOCAL が指定されたときに、プログラムで「db-short-names」を使用することはできません。ビュー定義や FIND ステートメントの検索式にフィールドリストを作成したり、READ ステートメントや HISTOGRAM ステートメントに読み込み順序制御フィールドを指定することもできません。これらの制限は、DBSHORT オプションの ON/OFF にかかわらず、Natural コンパイラによって制御されます。

DBSHORT の目的は、コンパイラの動作を次のように変更することです。

<b>DBSHORT=ON</b>	すべてが上記の「 <a href="#">予備知識</a> 」のとおり動作します。
<b>DBSHORT=OFF</b>	あらゆるデータベースフィールド識別子は、何文字で構成されているかにかかわらず「db-long-name」とみなされます。つまり、DDM のロングネーム列 (DDM 表示では "Name" と表記) のみが、参照フィールドを確認するために考慮され、DDM ショートネーム (DDM 表示では "DB" と表記) は完全に無視されます。

### DBSHORT=ON の目的

DBSHORT=ON の目的は、Natural プログラムで「db-long-name」を使用する代わりに「db-short-names」を使用して、データベースフィールドを参照できるようにすることです。

### DBSHORT=OFF の目的

DBSHORT=OFF を主に使用するのは、2 バイトの長さしかない識別子を持つロングネームが DDM に定義されている場合です。DDM の生成時に、この DDM でアクセスする、基礎となるデータベースが DB2 などの SQL の場合には、2 バイトのロングネームを作成するだけで済みます。これ以外のすべてのデータベースタイプについては、2 バイトの名前の長さでロングフィールドを定義しようとする、エラー SYSDDM4219 (SYSDDM ユーティリティ) が発生します。

ただし、DBSHORT=OFF が設定されているときは、DDM の「db-short-name」はコンパイラでチェックされません。したがって、プログラムで db-short-field が使用されると、構文エラー NAT0981 が発生します。

### PSIGNF - パック十進数の正記号の内部表現

<b>ON</b>	パック型数値の正の符号は、内部的に H'F' として表されます。これはデフォルト値です。
<b>OFF</b>	パック型数値の正の符号は、内部的に H'C' として表されます。

### TSENLB - TS プロファイルパラメータの適用性

このオプションは、プロファイルパラメータ TS (非標準の小文字使用が行われる地域のための出力変換) を、Natural システムライブラリ (SYSTEM を除いた "SYS" で始まる名前のライブラリ) のみに適用するか、またはすべてのユーザーライブラリにも適用するかを決定します。

TSENABLE=ON でカタログされた Natural オブジェクトでは、そのオブジェクトがシステムライブラリにない場合でも TS パラメータが決定されます。

<b>ON</b>	プロファイルパラメータ TS は、すべてのライブラリに適用されます。
<b>OFF</b>	プロファイルパラメータ TS は、Natural システムライブラリのみ適用されます。これはデフォルト値です。

## GFID - グローバルフォーマット ID の生成

このオプションを使用すると、Natural によるグローバルフォーマット ID の内部生成を制御して、フォーマットバッファ変換の再利用に関する Adabas のパフォーマンスに影響を与えることができます。

<b>ON</b>	グローバルフォーマット ID がすべてのビューに対して生成されます。これはデフォルト値です。
<b>VID</b>	グローバルフォーマット ID が、ローカル/グローバルデータエリアのビューに対してのみ生成されます。プログラム内で定義されたビューに対しては生成されません。
<b>OFF</b>	グローバルフォーマット ID は生成されません。

グローバルフォーマット ID の詳細については、Adabas のドキュメントを参照してください。

## Natural でグローバルフォーマット ID を生成するための規則

### ■ Natural ニュークリアスの内部システムファイルコールの場合

```
GFID=abccdde
```

要素	説明
<i>a</i>	x'F9'
<i>b</i>	x'22' または x'21' (DB ステートメントに依存)
<i>cc</i>	物理データベース番号 (2 バイト)
<i>dd</i>	物理ファイル番号 (2 バイト)
<i>ee</i>	ランタイムによって作成される番号 (2 バイト)

### ■ ユーザープログラムまたは Natural ユーティリティの場合

- GFID=abbbbbbc : ファイル番号が255以下で、Adabasバージョンが6.2より低い場合 (NTDBマクロ参照)

要素	説明
a	x'F8'、x'F7'、または x'F6'
bbbbbb	STOD 値の 1~6 バイト
c	物理ファイル番号

- GFID=axbbbbbc : ファイル番号が 255 よりも大きく、Adabasバージョンが6.2より低い場合

要素	説明
a	x'F8'、x'F7'、または x'F6'
x	物理ファイル番号 - 上位バイト
bbbbbb	STOD 値の 2~6 バイト
c	物理ファイル番号 - 下位バイト

- GFID=abbbbbbb : Adabasバージョン6.2以降の場合

要素	説明
a	x'F8'、x'F7'、または x'F6'  上記の意味は次のとおりです。  F6=UPDATE SAME F7=HISTOGRAM F8= 上記以外
bbbbbbb	STOD 値の 1~7 バイト



**Note:** STOD はストアクロック機械命令 (STCK) の戻り値です。

## LOWSRCE - 小文字ソースの許可

このオプションは、小文字または大文字と小文字が混在するプログラムソースをメインフレームプラットフォームで使用できるようにします。これにより、小文字または混在文字で記述されたプログラムを、他のプラットフォームからメインフレーム環境に移行しやすくなります。

<b>ON</b>	プログラムソースですべての小文字／大文字を許可します。
<b>OFF</b>	大文字モードのみを許可します。このためには、キーワード、変数名および識別子を大文字で定義する必要があります。これはデフォルト値です。

LOWSRCE=ON を指定して小文字を使用するときは、次の点を考慮してください。

- 変数名の構文規則では、2つ目以降の位置でも小文字が許可されます。したがって、小文字による変数と大文字による変数という2つの変数を定義できます。

例：

```
DEFINE DATA LOCAL
1 #Vari (A20)
1 #VARI (A20)
```

LOWSRCE=OFF を指定すると、上記の変数は異なるものとみなされます。

LOWSRCE=ON の場合、コンパイラでは大文字と小文字が区別されないため、小文字と大文字は同様に扱われます。同じ名前の変数を重複して定義することは許可されないため、これは構文エラーになります。

- I/O ステートメントまたは MOVE EDITED ステートメントでセッションパラメータ EM（編集マスク）を使用する場合は、変数に割り当てられたデータ設定のレイアウトに影響する文字（EM制御文字）およびデータ設定にテキストの部分挿入する文字があります。

例：

```
#VARI :='1234567890'
WRITE #VARI (EM=XXXXXXxXXXXX)
```

LOWSRCE=OFF の場合、英字形式の変数には大文字の X、H、およびシルコンプレクス (^) 記号しか使用できなくなるため、出力は "12345xx67890" になります。

LOWSRCE=ON の場合、x は大文字 X として扱われるため、このフィールドフォーマットの EM 制御文字として解釈され、出力は "1234567890" になります。この問題を回避するには、定数テキストの部分のアポストロフィ (') で囲んでください。

例：

```
WRITE #VARI(EM=XXXXX'xx'XXXXX)
```

このテキストの部分は、LOWSRCE の設定にかかわらず EM 制御文字とはみなされません。

- すべての変数名は LOWSRCE=ON によって大文字に変換されるので、I/O ステートメント (INPUT、WRITE、または DISPLAY) の変数名の表示は異なります。

例：

```
MOVE 'ABC' to #Vari
DISPLAY #Vari
```

LOWSRCE=OFF の場合、上記の例は次のよう出力されます。

```

#Vari
-----
ABC
```

LOWSRCE=ON の場合、上記の例は次のよう出力されます。

```

#VARI
-----
ABC
```

## TQMARK - 引用符の変換

<b>ON</b>	テキスト定数内の二重引用符は、シングルアポストロフィとして出力されます。これはデフォルト値です。
<b>OFF</b>	テキスト定数内の二重引用符は変換されず、二重引用符として出力されます。

例：

```
RESET A(A5)
A:= 'AB"CD'
WRITE '12"34' / A / A (EM=H(5))
END
```

TQMARK ON の場合、次のよう出力されます。

```
12'34
AB'CD
C1C27DC3C4
```

TQMARK OFF の場合、次のよう出力されます。

```
12"34
AB"CD
C1C27FC3C4
```

## THSEP - ダイナミック千桁単位セパレータ

このオプションは、コンパイル時に千桁単位セパレータの使用を有効または無効にするために使用できます。プロファイルパラメータ THSEP、プロファイル/セッションパラメータ THSEPCH、および『プログラミングガイド』の「セパレータ文字の表示方法のカスタマイズ」セクションを参照してください。

<b>ON</b>	千桁単位セパレータを使用します。文字列リテラルの一部ではないすべての千桁単位セパレータは、内部で制御文字と置き換えられます。
<b>OFF</b>	千桁単位セパレータを使用しません。つまり、コンパイラによって千桁単位セパレータ制御文字は生成されません。これが互換性のある設定です。

## CPAGE - 英数字定数に対するコードページのサポート

CPAGE オプションを使用して、変換ルーチンをアクティブにすることができます。変換ルーチンは、ランタイム時にオブジェクトが起動されると、すべての英数字定数を、コンパイル時にアクティブであったコードページからランタイム時にアクティブなコードページに変換します。

『Unicode およびコードページのサポート』ドキュメントの「プロファイルパラメータ」も参照してください。

<b>ON</b>	英字文字列に対するコードページのサポートは有効です。
<b>OFF</b>	英字文字列に対するコードページのサポートは無効です。これはデフォルト値です。

## DB2ARRY - SQL ステートメント SELECT および INSERT での DB2 配列のサポート

DB2ARRY オプションを使用すると、SQL ステートメントの SELECT または INSERT を 1 回実行することによって、複数行を DB2 から取得または DB2 に挿入できるようになります。これにより、配列を SQL SELECT ステートメントの受け取りフィールドおよび SQL INSERT ステートメントのソースフィールドとして指定できます。DB2ARRY を ON に指定すると、DB2 の VARCHAR/GRAPHIC 列に対して Natural の英数字配列を使用できなくなります。代わりに、Natural のロング英数字変数を使用する必要があります。

<b>ON</b>	DB2 配列のサポートが有効です。
<b>OFF</b>	DB2 配列のサポートが無効です。これはデフォルト値です。

## バージョン互換を維持するためのコンパイラオプション（リモートメインフレーム環境）

次のオプションを使用できます。

- FINDMUN - FIND ステートメントの比較ロジックの矛盾検出
- MASKCME - MASK の MOVE EDITED との互換性
- NMOVE22 - 同じ長さおよび精度の数値変数の割り当て
- V41COMP - 新バージョン 4.2 の構文の無効化

これらのオプションは、CMPO プロファイルパラメータと NTCMPO パラメータマクロのいずれかまたは両方のキーワードサブパラメータに対応します。

## FINDMUN - FIND ステートメントの比較ロジックの矛盾検出

Natural バージョン 2.3 では、FIND ステートメントの WITH 節のマルチプル設定フィールドに使用する比較ロジックが変更されています。これは、特定形式の FIND ステートメントが含まれるバージョン 2.2 プログラムがバージョン 3.1 でコンパイルされると、異なる結果が返されることを意味します。このオプションを使用して、バージョン 3.1 の拡張比較ロジックとは異なる方法でマルチプル設定フィールドが使用される WITH 節を含む FIND ステートメントを検索できます。

<b>ON</b>	コンパイル時に検出された、該当する形式のすべての FIND ステートメントに対し、エラー NAT0998 が返されます。
<b>OFF</b>	該当する形式の FIND ステートメントの検索は実行されません。これはデフォルト値です。

FIND ステートメントの WITH 節のマルチプルバリューフールドに対する比較ロジックが Natural バージョン 2.3 で変更されたのは、他のステートメント (IF など) の変換ロジックに合わせるためです。

FIND ステートメントの 4 種の形式を区別できます。次の例の MU は、マルチプルバリューフールドを表します。

### 1. FIND XYZ-VIEW WITH MU = 'A'

バージョン 2.2 以降では、MU の少なくとも 1 つ以上のオカレンスに "A" の値があるレコードが返されます。

### 2. FIND XYZ-VIEW WITH MU NOT EQUAL 'A'

バージョン 2.2 では、MU のどのオカレンスにも "A" の値がないレコードが返されます (4. と同じ)。バージョン 2.3 以降では、MU の少なくとも 1 つ以上のオカレンスに "A" の値がないレコードが返されます。

### 3. FIND XYZ-VIEW WITH NOT MU NOT EQUAL 'A'

バージョン 2.2 では、MU の少なくとも 1 つ以上のオカレンスに "A" の値があるレコードが返されます (1. と同じ)。バージョン 2.3 以降では、MU のすべてのオカレンスに "A" の値があるレコードが返されます。

### 4. FIND XYZ-VIEW WITH NOT MU = 'A'

バージョン 2.2 以降では、MU のどのオカレンスにも "A" の値がないレコードが返されます。これは、2. と 3. の形式の FIND ステートメントが含まれるバージョン 2.2 の既存のプログラムをバージョン 2.3 で新たにコンパイルした場合、異なる結果が返されることを意味します。

FINDMUN=ON を指定すると、コンパイル中に検出された 2. または 3. の形式のすべての FIND ステートメントに対してエラー NAT0998 が返されます。

このような場合でもバージョン 2.2 のときと同じ結果を取得するには、ステートメントを次のように修正する必要があります。

形式 2 の場合：

```
FIND XYZ-VIEW WITH MU NOT EQUAL 'A'
```

これを次のように修正します。

```
FIND XYZ-VIEW WITH NOT MU = 'A'
```

形式 3 の場合：

```
FIND XYZ-VIEW WITH NOT MU NOT EQUAL 'A'
```

これを次のように修正します。

```
FIND XYZ-VIEW WITH MU = 'A'
```

### MASKCME - MASK の MOVE EDITED との互換性

ON	YYYY マスク文字に一致する有効な年の値の範囲は 1582～2699 で、MASK オプションと MOVE EDITED とを互換します。プロファイルパラメータ MAXYEAR を 9999 に設定した場合は、有効な年の値の範囲は 1582～9999 です。
OFF	YYYY マスク文字に一致する有効な年の値の範囲は 0000～2699 です。これはデフォルト値です。プロファイルパラメータ MAXYEAR を 9999 に設定した場合は、有効な年の値の範囲は 0000～9999 です。

### NMOVE22 - 同じ長さおよび精度の数値変数の割り当て

ON	ソースとターゲットの長さや精度が同じである数値変数の割り当ては、Natural バージョン 2.2 の場合と同様に実行されます。
OFF	ソースとターゲットの長さや精度が同じである数値変数の割り当ては、Natural バージョン 2.3 以降の場合と同様に実行されます。つまり、ソースとターゲットの長さまたは精度が異なっているかのよう処理されます。これはデフォルト値です。

### V41COMP - 新バージョン 4.2 の構文の無効化

 **Important:** このコンパイラオプションは、Natural バージョン 4.2 で移行をスムーズにするためにのみ使用可能です。Natural バージョン 4.2 の後のリリースで再び削除されます。

Natural バージョン 4.2 で導入された多数の機能とプログラミング機能は、バージョン 4.2 で開発およびコンパイルされたプログラムをバージョン 4.1 環境でのオペレーションに使用するために再コンパイルするときに、問題を生じる可能性があります。該当する機能については、[下記のリスト](#)を参照してください。

V41COMP オプションによって、このような非互換性を検出し、再コンパイルが失敗した原因を示す理由コードを提供するエラーメッセージを表示することができます。可能な値は次のとおりです。

<b>ON</b>	バージョン 4.2 でのプログラムのコンパイル時に、バージョン 4.2 ではサポート対象であっても、バージョン 4.1 ではサポートされない構文構造が使用されようとするたびに拒否し、NAT0647 構文エラーと対応する理由コード（ <a href="#">下記参照</a> ）を出力します。
<b>OFF</b>	バージョン 4.1 の互換性のテストを実行しません。これはデフォルト値です。

## バージョン 4.2 と 4.1 間のコンパイル関連の相違点

次の表に、バージョン 4.2 と 4.1 におけるコンパイル関連の違いの概要、および互換性のない構文が検出されたときに表示される理由コードを示します。

機能	バージョン 4.2	バージョン 4.1	理由コード
新規フォーマット U (Unicode)	可能	不明	001
オカレンスの数が可変である配列  X-array の例：  <pre>DEFINE DATA LOCAL 1 #ARR (A10/1:*)</pre>	可能	不明	002
英字およびリテラル（定数）に可能な長さ	1 バイト～1 GB	1 バイト～253 バイト (NAT0264)	003
次の新規コンパイラパラメータ： THSEP            編集マスクでの千桁単位セパレータ文字 CPAGE            英数字定数のコードページ変換	可能	不明	004
次の新規ステートメント：  MOVE NORMALIZED MOVE ENCODED PARSE REQUEST DOCUMENT EXPAND/REDUCE/RESIZE ARRAY	可能	不明	005
SET GLOBALS ステートメント：  ■ セッションパラメータ CPCVERR=ON/OFF ■ ストラクチャードモード (SM=ON) 時の使用	可能	不明	006
次のシステム変数：	可能	不明	007

機能	バージョン 4.2	バージョン 4.1	理由コード
*PARSE-COL *PARSE-LEVEL *PARSE-NAMESPACE-URI *PARSE-ROW *PARSE-TYPE *CODEPAGE *LOCALE *TYPE *CURRENT-UNIT *UBOUND *LBOUND			
未使用	-	-	008
INCLUDE で提供されるソースパラメータの長さタイプ  例：  <pre>INCLUDE COPY01 'WRITE *LINE'                 'WRITE *PROGRAM'</pre>	フォーマット U (Unicode) で長さは指定なし	英字のみで長さは 80 バイト以下	009
データビューでの Adabas LA フィールドにおける次のいずれかの定義  ■ 253 バイトを超えるサイズ ■ タイプ DYNAMIC	可能	不明	010

# 10 DEBUG

---

`DEBUG object-name`

このコマンドは、Natural デバッガを呼び出すために使用します。コマンドには、デバッグするオブジェクトの名前を指定します。

デバッガの詳細については、『デバッガ』ドキュメントを参照してください。



**Note:** このコマンドは、バッチモードでは実行できません。



# 11 EDIT

---

▪ 構文 1 .....	48
▪ 構文 2 .....	50
▪ 構文 3 .....	50

このコマンドは、ソース形式の Natural プログラミングオブジェクトを編集する目的で Natural エディタを呼び出すために使用します。

コマンド構文には3つの形式があります。各形式は次の別個のセクションで説明されています。

関連コマンド：[READ](#)。

『Natural スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

『Natural スタジオの使用』ドキュメントの「エディタの呼び出し」も参照してください。

## 構文 1

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

*object-type*

次のオブジェクトタイプを編集できます。

```
{  
  { CLASS }  
  4  
  COPYCODE  
  { DIALOG }  
  3  
  7 (機能用)  
  GLOBAL  
  HELPROUTINE  
  LOCAL  
  MAP  
  PARAMETER  
  PROGRAM  
  { SUBPROGRAM }  
  N  
  SUBROUTINE  
  TEXT  
  VIEW  
}
```

呼び出されるエディタは、編集するオブジェクトタイプに応じて、次のように異なります。

- ローカルデータエリア、グローバルデータエリア、またはパラメータデータエリアは、データエリアエディタで編集します。
- マップはマップエディタで編集します。
- ダイアログはダイアログエディタで編集します。
- クラスはクラスビルダで編集します。
- EDIT VIEWは、現在のライブラリ内で *object-name* が指定されている場合にのみ機能します。表示するオブジェクトが DDM の場合は、DDM エディタが呼び出されます。
- その他すべてのオブジェクトタイプ（プログラム、サブプログラム、サブルーチン、7（機能用）、ヘルプルーチン、コピーコード、テキスト、説明）は、プログラムエディタで編集します。



**Note:** テキストオブジェクト「説明」は、メインフレームのみで使用可能です。説明とは、Predict データディクショナリに保存され、メンテナンスされているプログラム説明です。このタイプのオブジェクトは、Predict がインストールされている場合にのみ編集できます。

オブジェクトタイプについては、『プログラミングガイド』を参照してください。エディタについては、『エディタ』ドキュメントを参照してください。

編集するオブジェクトの名前を指定する場合、そのオブジェクトタイプを指定する必要はありません。

*object-name*

EDIT コマンドを使用して、編集するオブジェクトの名前を指定します。オブジェクト名の最大長は 8 文字です。



**Note:** DDM の場合、最大長は 32 文字です。

この後は、Natural によってオブジェクトが適切なエディタの編集ワークエリアにロードされ、後続の **SAVE**、**CATALOG**、または **STOW** コマンドに対してオブジェクト名が設定されます。

*object-name* を指定せず、ソースワークエリアにオブジェクトが存在しない場合は、空のプログラムエディタ画面が呼び出され、そこでプログラムを作成できます。ソースワークエリアが空ではない場合、オブジェクトは適切なエディタにロードされます。



**Note:** ソースワークエリアが空ではなく、エディタセッションで開かれているオブジェクトが存在する場合は、対応するエディタウィンドウが表示され、入力にフォーカスが置かれます。この間（Natural プログラムの実行など）にソースワークエリアに適用された変更内容は表示されません。

*library-id*

編集するオブジェクトが現在ログオンしているライブラリに存在しない場合は、編集するオブジェクトが存在するライブラリの *library-id* を指定する必要があります。

NaturalSecurity がアクティブも場合は、*library-id*を指定できません。これは、現在のライブラリ内にあるオブジェクトしか編集できないことを意味します。

## 構文 2

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

編集するオブジェクトの名前を覚えていない場合は、この形式の EDIT コマンドを使用してオブジェクトのリストを表示し、そのリストから希望するオブジェクトを選択できます。

EDIT *	現在のライブラリに存在するすべてのオブジェクトのリストを表示します。
EDIT <i>object-type</i> *	現在のライブラリに存在する指定タイプのすべてのオブジェクトのリストを表示します。

特定範囲のオブジェクト群からオブジェクトを1つ選択するには、システムコマンド LIST の記述と同じ方法で、*object-name* にアスタリスク表記とワイルドカード表記を使用できます。

## 構文 3

```
EDIT FUNCTION subroutine-name
```

EDIT FUNCTION コマンドは、最大 32 文字のサブルーチン名（オブジェクト名ではない）を使用してサブルーチンを編集するために使用できます。



**Note:** この構文で使用されるキーワード FUNCTION が、上記のリストにある Natural オブジェクトタイプ7（機能用）と同じではないことに注意してください。通常、機能とは別の Natural オブジェクトによって呼び出される Natural オブジェクトです。このような機能では、呼び出し側の Natural オブジェクトからパラメータ値を受け取り、そのオブジェクトにデータを返すことができます。

## 例

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

上記のサブルーチンがオブジェクト名 CHCKSUB で保存されている場合は、サブルーチン CHECK-PARAMETERS を次のコマンドで編集できます。

```
EDIT S CHKSUB
```

または、次のコマンドも使用できます。

```
EDIT F CHECK-PARAMETERS
```



# 12 EXECUTE

---

▪ 構文説明 .....	54
▪ EXECUTE コマンドの例 .....	55

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

このコマンドは、プログラムタイプまたはダイアログタイプのNaturalオブジェクトモジュールを実行するために使用します。オブジェクトモジュールは、あらかじめNaturalシステムファイルにカタログしておく（つまり、オブジェクト形式で保存しておく）か、またはNaturalニュークリアスにリンクしておく必要があります。オブジェクトモジュールを実行しても、現在エディタワークエリアにあるソースには影響しません。

このchapterでは、次のトピックについて説明します。

『Natural スタジオの使用』ドキュメントの「オブジェクトの実行」も参照してください。

## 構文説明

<b>EXECUTE</b>	キーワード EXECUTE はオプションであり、 <i>program-name</i> （実行するプログラムまたはダイアログの名前）だけで指定できます。
<b>REPEAT</b>	実行されるプログラムまたはダイアログで複数の画面が出力される場合に、これらの画面をプロンプトの介入なしに順番に出力するには、キーワード REPEAT をキーワード EXECUTE とともに指定します。
<i>program-name</i>	実行するプログラムまたはダイアログの名前。 <i>library-id</i> を指定しない場合、Naturalでは、指定されたプログラムまたはダイアログが現在のライブラリまたは現在の <i>steplib</i> ライブラリのいずれかにある場合でしか実行できません。デフォルトの <i>steplib</i> は SYSTEM です。
<i>library-id</i>	プログラムまたはダイアログが別のライブラリにある場合は、そのライブラリの <i>library-id</i> を指定します。この場合は、指定したライブラリにプログラムまたはダイアログが実際に保存されているときに限り実行できます。  SYSTEM を除き、"SYS" で始まる <i>library-id</i> は指定できません。
<i>parameter</i>	キーワード EXECUTE なしでプログラム名を指定してプログラムを実行するときは、プログラムにパラメータを渡すことができます。これらのパラメータは、実行されるプログラムの最初の INPUT ステートメントによって読み込まれます。  各パラメータは、位置指定パラメータまたはキーワードパラメータとして、セッションパラメータ ID で指定する場合と同様に、個別指定を空白または INPUT 区切り文字でそれぞれ区切って指定できます。  <b>注意:</b> 渡されるパラメータの1つに空白が含まれているか、またはパラメータが空白を含む文字列である場合は、プログラム名の直後に入力デリミタが設定されているときのみ転送が実行されます。

---

## EXECUTE コマンドの例

---

```
EXECUTE PROG1
```

```
EXECUTE PROG1 ULIB1
```

```
PROG1
```

```
PROG1 VALUE1 VALUE2 VALUE3
```

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

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

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

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



# 13 FIN

---

EIN

このコマンドは、Naturalセッションを終了するために使用します。バッチモードだけでなく、オンラインセッションにも適用されます。

バッチモードセッションは、コマンド入力データセットでエンドオブファイル状態が検出された場合も終了します。



# 14 GLOBALS

---

▪ 構文説明 .....	60
▪ パラメータのリスト .....	60
▪ SET GLOBALS とその他のステートメントとの相互作用 .....	62

`GLOBALS [parameter=value ...]`

このコマンドは、Natural セッションパラメータを設定するために使用します。

 **Note:** このコマンドは、追加パラメータが指定されている場合に、バッチモードでのみ実行できます。例えば、`GLOBALS SM=ON` はバッチモードで実行できます。

このchapterでは、次のトピックについて説明します。

## 構文説明

<b>GLOBALS</b>	GLOBALS コマンドをパラメータなしで入力すると、ウィンドウが表示され、ここでパラメータ設定を変更できます。このウィンドウの詳細については、『Natural スタジオの使用』ドキュメントの「セッションパラメータの使用」を参照してください。
<i>parameter</i>	<p>パラメータは任意の順序で指定できますが、空白で区切る必要があります。</p> <p>指定する複数のパラメータが1つのコマンド行に入りきらない場合は、GLOBALS コマンドを複数に分けて発行する必要があります。</p> <p>次に例を示します。</p> <pre>GLOBALS DC=, ID=.</pre> <p>一部のセッションパラメータは、上述のウィンドウを使用する場合にのみ変更できますが、コマンド行では変更できません。</p>

## パラメータのリスト

次の表に、システムコマンド GLOBALS で指定できるセッションパラメータのリストを示します。

パラメータ	機能
CF	端末コマンドの文字
COMPR	RPC バッファ圧縮の設定
CPCVERR	コードページ変換エラー
DBSHORT	データベースショートネームの解釈
DC	小数点表記の文字
DFOUT	出力の日付フォーマット
DFSTACK	スタックの日付フォーマット
DFTITLE	標準レポートタイトルの日付出力フォーマット
DU	ダンプ生成

パラメータ	機能
EJ	ページ換え
ENDIAN	コンパイルしたオブジェクトのエンディアンモード
FCDP	ダイナミックに保護された入力フィールドの充填文字
FS	ユーザー定義変数のデフォルトのフォーマット/長さ設定
GFID	グローバルフォーマット ID
IA	INPUT 割り当て文字
ID	INPUT 区切り文字
IM	入力モード
LE	処理ループの制限を超過したときの処理
LS	行サイズ
LT	処理ループの制限
ML	メッセージ行の位置
NC	Natural システムコマンドの使用
OPF	ヘルプルーチンによる保護されたフィールドの上書き
PM	出力モード
PS	Natural レポートのページサイズ
REINP	無効データに対する内部的な REINPUT ステートメントの発行
SA	サウンド端末アラーム
SF	フィールド間の空白
SM	ストラクチャードモードでのプログラミング
SYMGEN	シンボルテーブルの生成
THSEPCH	千桁単位セパレータ文字
TIMEOUT	RPC サーバレスポンスに対する待ち時間
TRYALT	代替サーバアドレスの試行
WH	ホールド状態でのレコードの待機
XREF	Natural の XRef データの作成
ZD	Zero-Division チェック
ZP	ゼロ出力

## SET GLOBALS とその他のステートメントとの相互作用

---

### SET GLOBALS ステートメント

システムコマンド GLOBALS と SET GLOBALS ステートメントでは、変更と同じパラメータが提供されます。両方とも同じ Natural セッション内で使用できます。GLOBALS コマンドで指定したパラメータ値は、新しい GLOBALS コマンドまたは SET GLOBALS ステートメントで上書きされるか、セッションが終了するか、またはユーザーが別のライブラリにログオンするまで有効です。

### セッションパラメータ設定に影響を与えるその他のステートメント

一部のパラメータ値は、プログラムの実行中に LIMIT、EJECT、および FORMAT の各ステートメントを使用するか、または INPUT、DISPLAY、PRINT、および WRITE の各ステートメントに指定したフォーマットエントリで上書きできます。

# 15 HELP

---

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

このコマンドは、エラーメッセージのオンラインヘルプを呼び出すために使用します。



**Note:** このコマンドは、バッチモードでは実行できません。

<b>HELP</b>	ヘルプメニューを表示します。
<b>HELP [NAT]nnnn</b>	HELP と番号（最大 4 桁、接頭辞 "NAT" は任意）を入力すると、この番号に関する Natural のエラー条件の詳細メッセージテキスト、つまり、Natural システムのエラーメッセージ NATnnnn の長文テキストが表示されます。
<b>HELP USERnnnn</b>	現在のライブラリのライブラリ固有のエラーメッセージ番号 nnnn の長文テキストを表示します。

『Natural スタジオの使用』ドキュメントの「ヘルプの使用」も参照してください。



# 16 INPL

## INPL [R]

このコマンドは、Natural の INPL ユーティリティを呼び出すために使用します。このユーティリティは、Software AG インストールデータセットをシステムファイルにロードするためにのみ使用します。詳細については、オンラインヘルプおよびプラットフォーム固有のインストールドキュメントを参照してください。

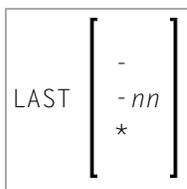
上記以外の場合は、オブジェクトハンドラを使用してオブジェクトをシステムファイルにロードします。

<b>INPL</b>	パラメータなしで INPL コマンドを入力すると、INPL ユーティリティが呼び出されます。
<b>INPL R</b>	INPL ユーティリティ機能の Natural Security リカバーが呼び出されます。これは、Natural Security がインストールされている場合にのみ使用できる機能です。  Natural Security ライブラリ SYSSEC へのアクセスを元の状態にリセットするために使用できます。これにより、ユーザー DBA、ライブラリ SYSSEC、および両者間のリンクが初期インストール後の状態に再定義される一方、その他すべての SYSSEC へのリンクはキャンセルされます。 『Natural Security』ドキュメントの「Countersignatures」セクションの「Inaccessible Security Profiles」も参照してください。

詳細については、『ツールおよびユーティリティ』ドキュメントの「INPL ユーティリティ」を参照してください。



# 17 LAST



LAST コマンドは、最後に実行されたコマンドを表示するために使用します。さらに、表示されたコマンドを再実行できます。また、これらのコマンドが実行される前に、コマンドを上書きすることもできます。

LAST コマンドでは、実際に入力したシステムコマンドのみを表示できます。入力したコマンドの結果として Natural が内部的に発行したコマンドは LAST コマンドでは処理できません。

LAST	最後に発行されたコマンドが、ダイアログボックスに配置され、実行可能になります。
LAST -	最後に発行されたコマンドが、ダイアログボックスに配置され、実行可能になります。 再び LAST - を入力すると、最後の1つのコマンドがダイアログボックスに配置されます。 繰り返し LAST - を入力すると、コマンドごとに「ページ」を戻すことができます。
LAST -nn	Natural では、発行されたコマンドの最後の20個が「記憶」されています。したがって、nn には20以内を指定する必要があります。 最後から nn 個のコマンドが、ダイアログボックスに配置され、実行可能になります。
LAST *	ダイアログボックスが表示され、最後に発行された20個のシステムコマンドが示されます。 ■ コマンドを再実行するには、必要なコマンドを選択し、[Copy] ボタンを使用してコマンドを [Selected Commands] リストボックスにコピーします。 ■ リストボックス内にある選択されたコマンドは、実行する前に修正できます。



# 18 LASTMSG

---

## LASTMSG

LASTMSG コマンドを使用すると、最後に発生したエラー状況に関する付加的な情報を表示できます。



**Note:** このコマンドは、リモートセッションでも使用できます。バッチモードですべての情報を読み取ることができます。

Natural でエラーメッセージが表示された場合、このエラーは、実際のエラーではなく別のエラーによって生成されたエラーである可能性があります（連鎖的に別のエラーを引き起こすことがあります）。このようなケースでは、LASTMSG コマンドを使用すると、発行されたエラーを、最初にエラー状況を引き起こしたエラーまでさかのぼってトレースすることができます。

LASTMSG コマンドを入力すると、最後に発生したエラー状況に対して表示されたエラーメッセージ、およびこのエラーを引き起こしたすべての先行する（表示されなかった）エラーメッセージを取得します。

『*Natural* スタジオの使用』ドキュメントの「ラストメッセージ」も参照してください。

### ▶手順 18.1. 対応するエラーに関する情報を表示するには

- これらのメッセージの1つを選択して [Details] ボタンをクリックするか、またはメッセージをダブルクリックします。

次の情報が表示されます。

- エラー番号
- エラーが発生した行番号
- エラーが発生したオブジェクトの名前、タイプ、およびレベル
- オブジェクトを含むライブラリの名前、データベース ID、およびファイル番号

- エラークラス (system = Natural が発行したエラー、user = ユーザーアプリケーションが発行したエラー)
- エラータイプ (runtime (ランタイム)、syntax (構文)、command execution (コマンド実行)、session termination (セッション終了)、program termination (プログラム終了)、remote procedure call (リモートプロシージャコール))
- エラーが発生した日付および時刻



**Note:** ライブラリ SYSEXT には、ユーザーアプリケーションプログラミングインターフェイス USR2006 があります。このユーザー API を使用すると、LASTMSG によって提供されたエラー情報を Natural アプリケーションに表示できます。

#### **Natural** リモートプロシージャコール (RPC) :

サーバーでのエラーの場合、データベース ID、ファイル番号、日付、および時刻は表示されません。

# 19 LIST

---

▪ 構文の概要 .....	72
▪ 個々のソースの表示 .....	74
▪ オブジェクトのリストの表示 .....	74
▪ ディレクトリ情報の表示 .....	75
▪ ビューの表示 .....	75
▪ リソースオブジェクトのファイル情報の表示 .....	76
▪ エラーメッセージコンテナのファイル情報の表示 .....	76

このシステムコマンドは、1つのオブジェクトのソースコードを表示したり、現在のライブラリに含まれている1つまたは複数のオブジェクトをリストしたりするために使用します。

 **Note:** このコマンドは、バッチモードでは実行できません。

このchapterでは、次のトピックについて説明します。

『*Natural* スタジオの使用』ドキュメントの「オブジェクトのリスト」、および [LIST XREF](#) と [LIST COUNT](#) の説明も参照してください。

## 構文の概要

---



*object-type*

```

*
{
  CLASS
  4
}
COPYCODE
DATA-AREAS
  GLOBAL
  LOCAL
  PARAMETER
  {
    DIALOG
    3
  }
  7 (機能用)
  8 (アダプタ用)
MAP
  {
    PROCESSOR
    CP
    5
  }
PROGRAM
ROUTINES
  HELPROUTINE
  {
    SUBPROGRAM
    N
  }
  SUBROUTINE
TEXT

```

*object-name*

*object-name*の代わりに、オブジェクトの名前を指定することもできます（長さは最大8文字）。また、アスタリスク（\*）表記を指定することもできます。下記の「例」を参照してください。

## 個々のソースの表示

LIST	パラメータなしで LIST コマンドのみを入力すると、ソースワークエリアの内容がリストされます。
LIST <i>object-name</i>	LIST コマンドで1つのオブジェクト名を入力する場合は、 <i>object-type</i> を指定する必要はありません。
LIST <i>object-type</i> <i>object-name</i>	<i>object-type</i> を指定する場合は、 <i>object-name</i> も指定する必要があります。  どちらの場合も、オブジェクトのソースコードがリストされます。

## オブジェクトのリストの表示

LIST <i>object-name</i>	現在のライブラリにあるすべてのオブジェクトをリストするには (DDMを除く)、 <i>object-type</i> ではなく、 <i>object-name</i> にアスタリスク (*)を指定します。
LIST <i>object-type</i> <i>object-name</i>	特定のタイプのオブジェクトをすべてリストするには、 <i>object-name</i> に特定の <i>object-type</i> とアスタリスク (*) を指定します。  特定の範囲にあるオブジェクトをリストするには、 <i>object-name</i> にアスタリスク表記 (*) とワイルドカード表記 (?) のいずれかまたは両方を使用します。

### 例

- 現在のライブラリにあるすべてのオブジェクトをリストします (DDM を除く)。

```
LIST *
```

- 現在のライブラリにあるすべてのサブルーチンをリストします。

```
LIST S *
```

- 名前が "SYS" で始まるすべてのオブジェクト (任意のタイプ) をリストします。

```
LIST SYS*
```

- 名前が "SYS" で始まるすべてのマップをリストします。

```
LIST M SYS*
```

- 現在のライブラリにあるオブジェクト PRG01 のディレクトリ情報をリストします。

LIST DIR PRG01

- NATAL、NATURAL、NATvrALなどのオブジェクトをすべてリストします。vrは、関連するバージョンおよびリリース番号を表しています。

LIST N?T\*AL

## ディレクトリ情報の表示

LIST DIRECTORY	<p>現在ソースワークエリアで最後にアクティブなオブジェクトのディレクトリ情報を表示します。</p> <ul style="list-style-type: none"> <li>■ ソースコード：            「保存された」日付と時刻、ライブラリ名、ユーザー ID、プログラミングモード（レポートまたはストラクチャード）、Natural バージョン、コードページ情報（使用可能な場合）、オペレーティングシステム、サイズ、エンコード</li> <li>■ オブジェクトコード：            「カタログされた」日付と時刻、ライブラリ名、ユーザー ID、プログラミングモード、Natural バージョン、コードページ情報（使用可能な場合）、オペレーティングシステム、サイズ、エンディアンモード</li> </ul> <p>Natural 以外（Natural の制御下でない）のエディタでソースコードを編集できるため、保存されたソースコードのディレクトリ情報は常に正確であるとは限りません。</p>
LIST DIRECTORY <i>object-name</i>	<p>指定されたオブジェクトに関するディレクトリ情報を表示します。 <i>object-name</i> にアスタリスク表記 (*) を使用した場合は、既存のオブジェクトのディレクトリ情報が連続して表示されます。</p>



**Note:** 表示されるコードページ情報には、コードページの先頭の 32 文字のみが示されません。

## ビューの表示

LIST VIEW	すべてのビューのリストを表示します（DDM）。
LIST VIEW <i>view-name</i>	<p>1つのビュー名を指定すると、指定したビューが表示されます。</p> <p><i>view-name</i> でアスタリスク表記を使用すると、すべてのビュー (*) または特定の範囲にあるビュー（例：A*）のリストを表示できます。</p>

## リソースオブジェクトのファイル情報の表示

---

<b>LIST RESOURCE</b> <i>name</i>	指定されたリソースオブジェクトに関するファイル情報を表示します。 <i>name</i> には、アスタリスク (*) 表記のみを使用できます。
----------------------------------	-------------------------------------------------------------------------

例 - 名前が W で始まるすべてのリソースオブジェクトのファイル情報を表示します。

```
LIST RESOURCE W*
```

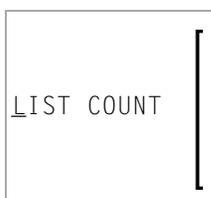
## エラーメッセージコンテナのファイル情報の表示

---

<b>LIST ERROR</b> <i>name</i>	指定されたエラーメッセージコンテナ <i>Nnn</i> APMSL.MSGに関するファイル情報を表示します。 <i>nn</i> には言語コードが入ります。 <i>name</i> には、アスタリスク (*) 表記のみを使用できます。
-------------------------------	------------------------------------------------------------------------------------------------------------------------

# 20 LIST COUNT

---



LIST COUNT コマンドを使用すると、現在ログオンしているライブラリに存在する Natural オブジェクトの数を取得できます。

<b>LIST COUNT</b>	オブジェクトの総数を表示します。
<b>LIST COUNT *</b>	オブジェクトタイプごとの数を表示します。
<b>LIST COUNT name&lt;</b>	名前が <i>name</i> よりも短いかまたは等しいオブジェクトの数を表示します。
<b>LIST COUNT name&gt;</b>	名前が <i>name</i> よりも長いかまたは等しいオブジェクトの数を表示します。
<b>LIST COUNT name*</b>	名前が <i>name</i> で始まるオブジェクトのみの数を表示します。

 **Note:** "undefined" (未定義) のオブジェクトタイプにリストされたオブジェクトがある場合、これは互換性のないバージョンのオブジェクトがライブラリ内に存在することを示しています。



# 21 LIST XREF

---

## LIST XREF

このコマンドは、Predict をインストールしている場合にのみ使用できます。

現在ログオンしているライブラリのアクティブクロスリファレンスをすべて表示します。

詳細については、『Predict』ドキュメントの「*Natural* の XREF」を参照してください。



# 22 LOGOFF

---

LOGOFF

関連コマンド：LOGON

LOGOFF コマンドを使用すると、ライブラリ ID を SYSTEM に設定し、Adabas パスワードを空白に設定できます。ソースプログラムワークエリアの内容は、このコマンドの影響を受けません。

LOGOFF によって、Natural グローバルパラメータの設定が影響を受けることはありません。

Natural Security 環境での LOGOFF の処理については、『Natural Security』ドキュメントの「Logging On」セクションの「How to End a Natural Session」を参照してください。



**Note:** LOGOFF では、Natural セッションが終了することはありません。

## ▶手順 22.1. セッションを終了するには

- システムコマンド **FIN** を使用するか、または **TERMINATE** ステートメントを含むプログラムを実行します。



# 23 LOGON

`LOGON library-id [password]`

関連コマンド：LOGOFF

LOGON コマンドは、ライブラリへのログオン、または新規ライブラリの作成に使用します。ライブラリ ID を指定すると、セッション中に保存するすべてのソースプログラムやオブジェクトプログラムは、指定ライブラリに保存されます（ただし、SAVE、CATALOG、または STOW コマンドで別のライブラリ ID を指定した場合を除く）。

LOGON コマンドによって、現在アクティブなウィンドウ内にあるソースプログラムが直接影響を受けることはありません。

LOGON では、すべての Natural グローバルデータエリア、アプリケーション独立変数（AIV）、SET KEY ステートメントで行われたすべての割り当て、保持されている ISN リストが解放されます。DDM バッファエリア内の DDM も解放されます。

<b>LOGON</b> <i>library-id</i>	ライブラリ ID は 1~8 文字で指定し、間に空白を入れないようにします。ライブラリ ID には、次の文字を使用できます。	
	A~Z	英大文字
	0-9	数字
	-	ハイフン
	_	下線
	ライブラリ ID の先頭文字は、英大文字にする必要があります。	
<b>LOGON</b> <i>library-id</i> <i>password</i>	Adabas パスワードを指定します。『Natural Security』ドキュメントの「ライブラリ管理」セクションで「セッションパラメータ」を参照してください。	

Natural Security 環境下での LOGON の処理については、『Natural Security』ドキュメンテーションの「ログオン」セクションを参照してください。



# 24 MAIL

```
MAIL [ { *  
        ?  
        mailbox-id } ]
```

MAIL コマンドを使用すると、Natural Security 環境下でメッセージをブロードキャストするための一種の「掲示板」として使用されるメールボックスを起動できます。メールボックスの内容と有効期限のいずれかまたは両方を変更できます。

<b>MAIL</b>	MAIL コマンドをパラメータなしで入力すると、メールボックス ID の入力を要求するウィンドウが表示されます。
<b>MAIL *</b>	使用可能なすべてのメールボックスを含むリストが表示されるため、メールボックスをこのリストから選択します。
<b>MAIL ?</b>	
<b>MAIL mailbox-id</b>	mailbox-id (最大 8 文字) を指定すると、対応するメールボックスが直接起動します。mailbox-id は、Natural Security で事前に定義しておく必要があります。

詳細については、『Natural Security』ドキュメントの「メールボックス」を参照してください。



# 25

## MAP

---

- Natural 開発サーバー環境への接続の確立 ..... 88
- Natural アプリケーションへの接続の確立 ..... 89

MAP コマンドでは、Natural コマンド行を使用して次の機能を実行できます。

このchapterでは、次のトピックについて説明します。

関連コマンド：[UNMAP](#)

## Natural 開発サーバー環境への接続の確立

コマンド行を使用してNatural 開発サーバーへの接続を確立する場合は、次のMAP コマンド構文が適用されます。

```
MAP ENVIRONMENT=environment-name server-name port-name[userid[password['parm=value;...']]]
```

この方法は、『SPoD を使用したリモート開発』ドキュメントの「Remote Development Using SPoD」セクションに記述されているダイアログと同じ効果があります。

<i>environment-name</i>	接続に使用されるエイリアス名。  <i>environment-name</i> を指定しない場合は、" <i>server(port)</i> " 形式のエイリアス名が生成されます。  環境名に空白が含まれる場合は、空白を一重引用符で囲む必要があります ('')。
<i>server-name</i>	メインフレーム、UNIX サーバー、または OpenVMS の Natural 開発サーバーの名前。
<i>port-name</i>	開発サーバーの TCP/IP ポート番号。
<i>userid</i>	開発サーバーにアクセスするためのユーザー ID。  <i>userid</i> としてアスタリスク (*) を入力すると、クライアントセッションのユーザー ID が使用されます。
<i>password</i>	開発サーバーに Natural Security をインストールしている場合は、必要なパスワードを指定します。  <i>password</i> としてアスタリスク (*) を入力すると、空のパスワード文字列が開発サーバーに送信されます。
<i>parm</i>	開発サーバーに対して動的パラメータが必要な場合、一重引用符 ('...') を使用してセッションパラメータを指定してください。

Natural 開発サーバーのセッションをアンマップするには、[UNMAP](#) コマンドを使用します。

## Natural アプリケーションへの接続の確立

コマンド行を使用して Natural アプリケーションへの接続を確立する場合は、次の MAP コマンド構文が適用されます。

```
MAP APPLICATION=application-name [userid [password]
```

この方法は、『SPoDを使用したリモート開発』ドキュメントの「アプリケーションのマップ/アンマップ」セクションに記述されているダイアログと同じ効果があります。「Natural アプリケーション」の詳細については、『Natural Single Point of Development』ドキュメントを参照してください。

<i>application-name</i>	アプリケーションの名前です。
<i>userid</i>	アプリケーションにアクセスするためのユーザー ID。  <i>userid</i> としてアスタリスク (*) を入力すると、クライアントセッションのユーザー ID が使用されます。
<i>password</i>	開発サーバーに Natural Security をインストールしている場合は、必要なパスワードを指定します。  <i>password</i> としてアスタリスク (*) を入力すると、空のパスワード文字列が開発サーバーに送信されます。

Natural 開発サーバーのアプリケーションをアンマップするには、UNMAP コマンドを使用するか、「アプリケーションのマップ/アンマップ」セクションに記述されているダイアログを使用できます。



# 26 PROFILE

---

PROFILE コマンドは、Natural Security がインストールされている場合にのみ使用可能です。

## PROFILE

PROFILE コマンドを使用すると、現在有効になっているセキュリティプロファイルを表示できます。このプロファイルは、現在の Natural 環境の実際の使用条件を知らせます。

詳細については、『*Natural Security*』ドキュメントの「PROFILE コマンド」を参照してください。



# 27 PURGE

---

PURGE [*object-name* ...]

PURGE コマンドを使用すると、1つまたは複数のソースオブジェクトを削除できます。



**Note:** プロファイルパラメータ RECAT が ON に設定されている場合、対応するカタログ化オブジェクトが存在すると、PURGE コマンドではそのソースが拒否されます。

<b>PURGE</b>	<i>object-name</i> を指定せずに PURGE コマンドを入力すると、現在のライブラリにあるすべてのオブジェクトのリストが表示されます。このリストでは、削除するオブジェクトをマークできます。
<b>PURGE</b> <i>object-name</i>	<i>object-name</i> として、削除するオブジェクトの名前を指定します。現在ログオンしているライブラリに保存されているオブジェクトだけを削除できます。  特定の文字列で始まるすべてのオブジェクトを削除する場合は、 <i>object-name</i> にアスタリスク (*) を使用します。



# 28 READ

---

`READ object-name [library-id]`

関連コマンド：[EDIT](#)

READ コマンドを使用すると、ソース形式で保存されているオブジェクトをソースワークエリアに転送できます。現在のソースワークエリア内にあるオブジェクトは、新たに読み込まれたオブジェクトによって上書きされます。

『*Natural* スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

<b>READ <i>object-name</i></b>	読み込むオブジェクトの名前です。  ライブラリ ID を指定せずに <i>object-name</i> のみを指定すると、そのオブジェクトが現在ユーザーがログオンしているライブラリに保存されている場合にのみ読み込まれます。
<b>READ <i>object-name</i></b>  <i>library-id</i>	読み込むオブジェクトが存在するライブラリです。  <i>object-name</i> と <i>library-id</i> の両方を指定した場合、 <i>Natural</i> では、指定されたライブラリ ID にそのオブジェクトが保存されているときにのみ読み込みが行われます。



# 29 REGISTER

---

```
REGISTER { class-module-name } [ [ { library-name } [ [ { ES } ] ] ] ]  
* * * [ { IM } ] ] ]  
* * * [ { EM } ] ] ]
```

関連コマンド：[UNREGISTER](#)

REGISTER コマンドを使用すると、Natural クラスを登録できます。それらは、Natural を起動したサーバー ID に登録されます。

詳細については、『オペレーション』ドキュメントの「*NaturalX* アプリケーションの管理」で「REGISTER コマンド」を参照してください。



# 30 RENAME

RENAME コマンドは、リモート開発環境のコマンド行経由で使用することはできません。

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

RENAME コマンドを使用すると、Natural プログラミングオブジェクトに別の名前を指定できます。また、オブジェクトタイプを変更することもできます。

名前を変更できるオブジェクトは、一度に1つのみです。名前を変更するオブジェクトは、現在ログオンしているライブラリに保存されている必要があります。一貫性を確保するために、Natural では、ソースコードとオブジェクトコードのいずれかまたは両方の名前が変更されません。

『Natural スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

<b>RENAME</b>	パラメータを指定せずに RENAME コマンドを発行すると、 <b>[Rename Object]</b> ウィンドウが表示され、コマンド行と同じパラメータを指定できます。	
<i>old-name</i>	<i>old-name</i> には、名前を変更するオブジェクトの現在の名前を指定します。	
<i>new-name</i>	今後オブジェクトは <i>new-name</i> に指定した名前で保存されます。	
<i>new-type</i>	ソース形式のオブジェクトの名前を変更する場合は、対応する文字を <i>new-type</i> に指定することによって、そのオブジェクトのオブジェクトタイプも変更できます。	
	<i>new-type</i> に指定可能な値は、次のとおりです。	
	3	ダイアログ
	4	クラス
	5	プロセッサ
	7	ファンクション
8	アダプタ	

	9	リソース
	A	パラメータデータエリア
	C	コピーコード
	G	グローバルデータエリア
	H	ヘルプルーチン
	L	ローカルデータエリア
	M	マップ
	N	サブプログラム
	P	プログラム
	S	サブルーチン
	T	テキスト
	Y	ルール
	Z	記録

『Natural スタジオの使用』ドキュメントの「オブジェクトの名前の変更」も参照してください。

# 31 RENUMBER

---

RENUMBER [(n)]

RENUMBER コマンドを使用すると、ソースワークエリア内にあるソースプログラムの番号を再設定できます。

<b>RENUMBER</b>	パラメータを指定せずに RENUMBER コマンドを入力した場合、番号再設定で使用される増分は 10 になります。
<b>RENUMBER (n)</b>	n には、番号再設定の増分として 1~10 の値を指定できます。



# 32 RETURN

RETURN  $\left[ \left\{ \begin{array}{l} I \\ nn \\ * \end{array} \right\} \right]$

RETURN コマンドを使用すると、前の（または最初の）Natural アプリケーションに戻ることができます。

<b>RETURN</b>	パラメータを指定せずに RETURN を発行した場合は、制御は前のアプリケーションに戻されます（ <b>SETUP</b> システムコマンドの定義）。前のアプリケーションに関する情報はすべて削除されます。前のアプリケーションがない場合は、制御は初期のアプリケーションに戻されます。  RETURN を発行しても戻り位置が設定されていない場合、RETURN コマンドは無視されます。  <b>Natural Security</b> の場合：  RETURN を発行して戻り位置が設定されていなかった場合には、LOGOFF コマンドが実行されます。
<b>RETURN I</b>	制御を初期のアプリケーションに直接戻します。このオプションを使用すると、以前のアプリケーションに関するすべての定義が削除されます（初期のアプリケーションを除く）。
<b>RETURN nn</b>	制御を nn 番目のアプリケーションに戻します。このオプションを使用すると、nn 番目以降のアプリケーションの情報はすべて削除されます。
<b>RETURN *</b>	現在設定されているすべての戻り位置のリストを表示します。このリストでは、戻り位置を選択できます。

詳細な説明および例については、**SETUP** コマンドを参照してください。



# 33 RUN

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

RUN コマンドを使用すると、ソースプログラムまたはダイアログをコンパイルして実行できます。ソースワークエリア内または Natural システムファイル内にあるプログラムやダイアログを実行できます。

『Natural スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

<b>RUN</b>	<i>program name</i> を指定しない場合、Natural では、現在ワークエリア内にあるプログラムまたはダイアログがコンパイルおよび実行されます。
<b>REPEAT</b>	REPEAT を指定すると、プログラムまたはダイアログが複数の出力画面を生成する場合に、間にプロンプトメッセージを出さずに連続で画面を出力します。プログラムまたはダイアログが終了すると、Natural はコマンドモードに入ります。
<i>program-name</i>	実行するプログラムまたはダイアログの名前です。  <i>program-name</i> を指定してライブラリ ID を指定しない場合は、指定のプログラムまたはダイアログが現在ログオンしているライブラリ ID のもとに保存されているときにのみ、そのソースプログラムまたはダイアログがソースワークエリアに読み込まれ、コンパイルして実行されます。現在のライブラリ ID で保存されていない場合は、エラーメッセージが表示されます。
<i>library-id</i>	実行するプログラムまたはダイアログが存在するライブラリです。  <i>program-name</i> と <i>library-id</i> の両方を指定する場合は、指定のライブラリ ID のもとにこのプログラムまたはダイアログが保存されているときにのみ、検索が行われ、コンパイルして実行されます。現在のライブラリ ID で保存されていない場合は、エラーメッセージが表示されます。  <i>library-id</i> の設定値を "SYS" (SYSTEM を除く) で始めることはできません。

『Natural スタジオの使用』ドキュメントの「オブジェクトの実行」も参照してください。



# 34 SAVE

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

関連コマンド： [STOW](#) | [CATALOG](#)

このコマンドは、現在エディタのワークエリアにあるプログラミングオブジェクトのソースコードを Natural システムファイルに保存するために使用します。

『Natural スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

 **Caution:** SAVE コマンドは、プロファイルパラメータ `RECAT` が ON に設定されている場合には使用できません。この場合は、[STOW](#) コマンドを使用してオブジェクトをコンパイルおよび保存します。

<b>SAVE</b>	<i>object-name</i> を指定しないでコマンドを使用する場合は、ソースワークエリア内の現在のソースオブジェクトがライブラリに保存されます。このライブラリから、（例えば <code>EDIT</code> または <code>READ</code> を使用して）オブジェクトがソースワークエリアに読み込まれます。既存のソースコードは置換されます。
<b>SAVE</b> <i>object-name</i>	新しいオブジェクトが作成されます。 <i>object-name</i> として、保存するオブジェクトの名前を指定します。新しいオブジェクトは現在のライブラリに保存されます。オブジェクトが存在する場合、コマンドは拒否されます。
<b>SAVE</b> <i>object-name</i> <i>library-id</i>	オブジェクトを別の名前で作成するか、または新しく作成したオブジェクトを保存するとき、オブジェクトはデフォルトで現在のライブラリに保存されます。別のライブラリに保存する場合は、 <i>object-name</i> の後に必要な <i>library-id</i> を指定する必要があります。新しいオブジェクトが作成され、オブジェクトが存在する場合はコマンドが拒否されます。

『Natural スタジオの使用』の「オブジェクトの保存」も参照してください。



# 35 SCAN

---

## SCAN

このコマンドは、Naturalオブジェクトとこれらのオブジェクト内の文字列を検索するために使用するダイアログボックスを表示します。このダイアログボックスの詳細については、『*Naturalスタジオの使用*』の「ライブラリ内のオブジェクトの検索」を参照してください。



**Note:** このコマンドは、バッチモードでは実行できません。

---

# 36 SCRATCH

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

このコマンドは、1つ以上のオブジェクトを（ソースとオブジェクトの両方の形式で）削除するときに使用します。ソースワークエリアの内容には影響しません。

<b>SCRATCH</b>	<i>object-name</i> または <i>object-name</i> を指定せずに、アスタリスク (*) を指定して
<b>SCRATCH *</b>	SCRATCH コマンドを入力すると、現在のライブラリのすべてのオブジェクトまたは選択したすべてのオブジェクトのリストが表示されます。そのリストで、削除 (SCRATCH) する各オブジェクトをマークできます。
<b>SCRATCH</b> <i>object-name</i>	<i>object-name</i> として、削除するオブジェクトの名前を指定します。現在ログオンしているライブラリ内に存在しているオブジェクトのみを削除できます。  特定の文字列で始まるすべてのオブジェクトを削除する場合は、 <i>object-name</i> にアスタリスク (*) を使用します。



**Note:** パラメータファイルに有効ではない FDIC ファイルが指定されると、SCRATCH コマンド発行時に Natural は該当するエラーメッセージを表示します。



# 37 SETUP

---

- 構文説明 ..... 114
- 例：SETUP/RETURN ..... 115

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

このコマンドは、RETURN コマンドで制御を返す先のアプリケーションを定義するために使用します。これにより、Natural セッション中にアプリケーション間を容易に移動できます。

このchapterでは、次のトピックについて説明します。

## 構文説明

SETUP システムコマンドで発行できるコマンド構文とパラメータについて説明します。パラメータを省略する場合は、INPUT 区切り文字を使用して、次のパラメータの先頭をマークできます。

<b>SETUP</b>	パラメータを何も指定せずに SETUP を発行すると、コマンド情報の入力のためのメニューが表示されます。
<i>application-name</i>	<p>制御が返されるアプリケーションの名前です。最大 8 文字 (A8) の英数字を使用できます。</p> <p><i>application-name</i> が空白の場合、LOGON コマンドは発行されません。これにより、同じアプリケーション内で複数の戻り位置が持てるようになります。</p> <p><i>application-name</i> が "*" の場合は、システム変数 *LIBRARY - ID の現在の設定 (SETUP が発行された時点の設定) が、RETURN 発行時に LOGON コマンドを作成するために使用されます。</p>
<i>command-name</i>	<p>アプリケーションに制御が返されたときに実行するコマンドの名前です。最大 60 文字 (A60) の英数字を使用できます。</p> <p><i>command-name</i> が空白の場合、LOGON の後にコマンドは発行されません。これは、すでにスタートアッププログラムが定義されているような Natural Security のアプリケーションに便利です。</p> <p><i>command-name</i> が "*" の場合は、システム変数 *STARTUP の現在の設定 (SETUP が発行された時点の設定) が、RETURN 発行時のスタートアップコマンドとして使用されます。</p>
<b>I</b>	<p>I オプションを指定すると、以前の SETUP コマンドで定義されていた戻り位置はすべて削除され、SETUP I で指定されたアプリケーションが新しい開始アプリケーションとして定義されます。</p> <p>非セキュリティ環境では、ライブラリ SYSTEM から別のライブラリにログオンし、戻り位置が設定されていないと、ログオンしたライブラリが開始戻り位置となります。</p>

## 例：SETUP/RETURN

1. Natural セッションを開始します（デフォルトのアプリケーション=APPL1）。

戻り位置 "APPL1" がレベル 1 に定義されます。

2. コマンド LOGON APPL2 を発行します。
3. 2つのコマンドをスタックするプログラムを実行します（戻り位置を設定し、別のアプリケーションに移動します）。

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

リターンポイント "APPL2, STARTUP MENU" がレベル 2 で定義されます。

4. LOGON APPL4 コマンドを発行します（他のアプリケーションを選択します）。
5. RETURN という設定を持つ PF キーを押します。Natural がユーザーに対して次のコマンドを発行します。

```
LOGON APPL2  
MENU
```

APPL2 に戻り、レベル 2 を削除します。

6. 次のコマンドが含まれたプログラムを実行します。

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

リターンポイント "APPL2, STARTUP MENU" がレベル 2 で定義されます。

7. 次のコマンドが含まれたプログラムを実行します。

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

リターンポイント "APPL5, STARTUP MENU" がレベル 3 で定義されます。

8. 次のコマンドが含まれたプログラムを実行します。

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

リターンポイント "APPL6, STARTUP MENU" がレベル 4 で定義されます。

9. 次のコマンドが含まれたプログラムを実行します。

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

リターンポイント "APPL7, STARTUP MENU" がレベル 5 で定義されます。

10. 次のコマンドが含まれたプログラムを実行します。

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

リターンポイント "APPL8, STARTUP MENU" がレベル 6 で定義されます。

11. RETURN 2 コマンドを発行します (2 レベル戻ります)。

Natural は 2 つ前のセッションである APPL7 にユーザーを戻します (ここで APPL8 の情報はすべて消去されます)。レベル 6 (APPL8) は削除され、レベル 5 (APPL7) は制御を受けてレベルが削除されます。

12. コマンド RETURN を発行します。

レベル 4 (APPL6) が制御を受け、レベルが削除されます。Natural は APPL7 の 1 つ前セッションである APPL6 にユーザーを戻します。

13. コマンド RETURN を発行します。

レベル 3 (APPL5) が制御を受け、レベルが削除されます。Natural は APPL6 の 1 つ前セッションである APPL5 にユーザーを戻します。

14. RETURN 1 コマンドを発行します。

レベル 2 (APPL2) が削除され、レベル 1 (APPL1) が制御を受けます。

# 38 STOW

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

関連コマンド：SAVE | CATALOG

このコマンドは、Natural システムファイルに Natural プログラミングオブジェクトを（ソース形式とオブジェクト形式で）保存するために使用します。このコマンドは、CATALOG とそれに続く SAVE とみなすことができます。

『Natural スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

<b>STOW</b>	<i>object-name</i> を指定しないでコマンドを使用する場合は、ソースワークエリア内の現在のソースオブジェクトと生成されたコードが、ソースワークエリアに（例えば、EDIT または READ を使用して）最後に読み込まれたオブジェクトの名前でライブラリに保存されます。ソースエリアに保持されるソースコードと生成されたコードが、同じ名前で現在のライブラリに保存されます。
<b>STOW</b> <i>object-name</i>	このコマンドは、 <i>object-name</i> という名前の新しいオブジェクト（ソースと生成されたコード）を現在のライブラリに保存するために使用します。オブジェクトがソースまたはカタログのいずれかの形式で存在する場合、コマンドは拒否されます。
<b>STOW</b> <i>object-name</i> <i>library-id</i>	<i>object-name</i> と <i>library-id</i> の両方を指定すると、新しいオブジェクトが作成され、指定のライブラリ ID にその名前で保存されます。オブジェクトがソースまたはカタログのいずれかの形式で存在する場合、コマンドは拒否されます。

 **Note:** 有効ではないパラメータファイルに FDIC システムファイルが指定されていると、STOW コマンドが発行されるときに Natural は該当するエラーメッセージを表示します。

『Natural スタジオの使用』の「オブジェクトの Stow」も参照してください。



# 39      STRUCT

---

- ソースコード行のインデント ..... 120

STRUCT [(n)]

このコマンドを使用して、エディタのワークエリアに現在あるプログラミングオブジェクトのソースコードのインデントを実行します。

<b>STRUCT</b>	デフォルトでは（つまり、"(n)" が指定されていない場合）、インデントは2単位の位置ずつ行われます。
<b>STRUCT (n)</b>	パラメータ "(n)" を提供して、インデントに使用するスペースの数を指定できます。 使用できる値は1~9です。例： STRUCT (5)

次のステートメントタイプが STRUCT コマンドの影響を受けます。

- 処理ループ（READ、FIND、FOR など）
- 条件ステートメントブロック（AT BREAK、IF、DECIDE FOR など）
- DO/DOEND ステートメントブロック
- DEFINE DATA ブロック
- インラインサブルーチン

このchapterでは、次のトピックについて説明します。

## ソースコード行のインデント

ソースコードの行に、プログラムの階層構造を反映したインデントを行うことができます。

 **Note:** レポーティングモードのプログラムに行われるインデントはストラクチャードモードのプログラムとは異なります。

### 部分的なインデント

特殊ステートメント `/*STRUCT OFF` および `/*STRUCT ON` を使用して構造的なインデントからプログラムソースを部分的に除外できます。このステートメントはソースコード行の先頭から入力する必要があります。この2つのステートメント間のソースコード行は STRUCT コマンドを発行しても元のままです。

例：

インデントする前のプログラムは次のとおりです。

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

同じプログラムをインデントすると、次のようになります。

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



# 40 SYSAPI

---

## SYSAPI

このコマンドは SYSAPI ユーティリティを呼び出します。

このユーティリティを使用して、Entire Output Management (NOM) などの Natural アドオン製品で提供されているアプリケーションプログラミングインターフェイス (API) を探すことができます。

各 API について、ユーティリティ SYSAPI は、API の機能説明を含み、API の効果のテストに使用できる 1 つ以上のプログラム例を提供します。

詳細については、『ツールおよびユーティリティ』ドキュメントの「SYSAPI-Natural アドオン製品の API」を参照してください。



# 41 SYSCP

---

SYSCP

このコマンドは SYSCP ユーティリティを呼び出します。

SYSCP ユーティリティは、コードページ情報を取得するために使用できます。

詳細については、『ツールおよびユーティリティ』ドキュメントの「SYSCP ユーティリティ - コードページの情報」を参照してください。



# 42      SYSERR

---

## SYSERR

このコマンドは SYSERR ユーティリティを呼び出します。

SYSERR ユーティリティを使用して、独自のアプリケーション固有メッセージを作成できます。

- SYSERR ユーティリティを使用してエラーや情報メッセージを Natural コードから分離し、個別に管理できます。
- 各種メッセージに対するメッセージの統一およびメッセージの定義と同様に、メッセージを別の言語に翻訳したり、メッセージに拡張メッセージを連結したりすることができます。
- SYSERR ユーティリティを使用して、既存の Natural システムメッセージのテキストを修正することもできますが、新規 Natural リリースではこの修正は失われるので推奨できません。

詳細については、『ツールおよびユーティリティ』ドキュメントの「SYSERR ユーティリティ」を参照してください。



# 43 SYSEXT

---

## SYSEXT

このコマンドは SYSEXT ユーティリティを呼び出します。

このユーティリティは、ライブラリ SYSEXT に含まれるさまざまな Natural アプリケーションプログラミングインターフェイスを表示するために使用します。

詳細については、『ツールおよびユーティリティ』ドキュメントの「*SYSEXT - Natural* アプリケーションプログラミングインターフェイス」を参照してください。



# 44 SYSEXV

---

## SYSEXV

このコマンドは、現在の Natural バージョンの新機能の例を備えた SYSEXV アプリケーションを呼び出します。

あるいは、コマンド LOGON SYSEXV と VERSION を使用して SYSEXV アプリケーションを開始できます。



# 45    SYSFILE

---

- リモートメインフレーム環境での SYSFILE ..... 134

**SYSFILE**

このコマンドは、ワークファイルと出力ファイルの情報を表示します。次の情報を取得できません。

- レポート
- 論理デバイス
- 物理デバイスの定義
- プリンタプロファイルの定義
- ワークファイルの定義

『*Natural* スタジオの使用』の「ワークファイルおよび出力ファイル」も参照してください。

ワークファイルと出力ファイルの詳細については、次を参照してください。

- 『*コンフィグレーションユーティリティ*』ドキュメントの「プリンタプロファイル」
- 『*コンフィグレーションユーティリティ*』ドキュメントの「デバイス/レポート割り当て」
- 『*オペレーション*』ドキュメントの「ワークファイル」

## リモートメインフレーム環境での SYSFILE

---

次のワークファイルおよび出力ファイル情報を取得できます。

- 割り当てのタイプ
- レコードフォーマット
- 論理レコード長
- ブロックサイズ
- ステータス
- ダイナミックパラメータ指定

# 46

## SYSINST

---

### SYSINST

このコマンドは、Natural インストーラを呼び出します。このユーティリティは、Natural Security などの Natural アドオン製品のインストールを支援します。

詳細については、『ツールおよびユーティリティ』ドキュメントの「インストーラ」を参照してください。



# 47

## SYSMAIN

---

### SYSMAIN

このコマンドは SYSMAIN ユーティリティを呼び出します。このユーティリティは、コピー、移動、削除など、Natural オブジェクトに対する操作を実行するために使用します。SYSMAIN ユーティリティは、ある環境の Natural システムにあるオブジェクトを、インポート機能を使用して別の環境に転送するために使用することもできます。

詳細については、『ツールおよびユーティリティ』ドキュメントの「SYSMAIN ユーティリティ」を参照してください。



**Note:** このコマンドは、バッチモードでは実行できません。



# 48

## SYSMN

---

SYSMN

このコマンドはメインフレームナビゲーションを呼び出します。

詳細については、『メインフレームナビゲーション』ドキュメントを参照してください。

---

# 49

## SYSNCP

---

### SYSNCP

このコマンドは SYSNCP ユーティリティを呼び出します。

詳細については、『ツールおよびユーティリティ』ドキュメントの「SYSNCP ユーティリティ」を参照してください。



# 50

## SYSOBJH

---

SYSOBJH

このコマンドはオブジェクトハンドラを呼び出します。オブジェクトハンドラを使用して、Natural 環境での配布のために Natural オブジェクトと Natural 以外のオブジェクトを処理します。

詳細については、『ツールおよびユーティリティ』ドキュメントの「オブジェクトハンドラ」を参照してください。



# 51 SYSPROD

---

## SYSPROD

このコマンドで、Naturalサイトにインストールされている製品を確認できます。現在のNaturalバージョン、Natural 選択ユニット、および Natural 環境で実行している製品に関する情報を取得します。

コマンドを入力すると、ダイアログが表示され、インストールされている製品ごとに次のような情報が表示されます。

- 製品名
- 製品バージョン
- パッチレベル
- インストール日と時刻
- 製品識別コード (ID)

『Natural スタジオの使用』の「製品情報」も参照してください。



# 52 SYSPROF

---

## SYSPROF

このコマンドは、Natural システムファイルの現在の定義を表示します。

各システムファイルについて、次の情報が **[System Files]** ページに表示されます。

- ファイル名
- データベース ID
- ファイル番号
- データベースタイプ

また、データベース ID とファイル番号の定義済み組み合わせごとに、次の情報を表示できます。

- ファイルシステム内のパス（ **[Files in File System]** ページ上）
- 割り当てられている論理ファイル番号（ **[All Files]** ページ上）

『Natural スタジオの使用』の「システムファイル」も参照してください。



# 53

## SYSRPC

---

### SYSRPC

このコマンドは SYSRPC ユーティリティを呼び出します。

SYSRPC ユーティリティは、リモートプロシージャコールをメンテナンスする機能を提供します。

詳細については、『ツールおよびユーティリティ』ドキュメントの「SYSRPC ユーティリティ」を参照してください。

SYSRPC ユーティリティ機能を適用して、サーバーシステムとクライアントシステム間の通信のフレームワークを確立する方法については、『*Natural* リモートプロシージャコール (RPC)』ドキュメントを参照してください。

---

# 54

## SYSWIZDB

---

### SYSWIZDB

このコマンドは、Naturalスタジオ内の開発ツールウィザードであるデータブラウザを呼び出します。これにより、ファイル構造を表示、出力、または保存することができます。

詳細については、『ツールおよびユーティリティ』ドキュメントの「データブラウザ」を参照してください。



# 55 SYSWIZDW

---

## SYSWIZDW

このコマンドは、特定の用途のためのダイアログを作成するツールであるダイアログウィザードを呼び出します。定義済みダイアログボックスに、目的に適合する複数のレイアウトを設定できます。

詳細については、『エディタ』ドキュメントの「ダイアログエディタ」セクションの「ダイアログウィザード」を参照してください。



# 56 TECH

---

## TECH

このコマンドは、Naturalセッションに関する次の技術情報およびその他の情報を表示します。

- ユーザー ID
- ライブラリ ID
- Natural バージョン、リリースおよび SM レベル
- スタートアップ (開始) トランザクション
- Natural Security インジケータ
- オペレーティングシステム名およびバージョン
- マシンクラス
- ハードウェア
- TP モニタ (メインフレームおよびリモート構成の Windows (\*TPSYS) のみ)
- デバイスタイプ
- 端末 ID (メインフレームおよびリモート構成の Windows のみ)
- コードページ
- ロケール
- 最後に発行されたコマンド
- 最後に発生したエラーに関する情報
- 現在アクティブなすべての STEPLIB の名前、データベース ID およびファイル番号
- 現在アクティブなプログラミングオブジェクトおよび上位レベルの全オブジェクトの名前、タイプ、レベル、および下位レベルのプログラミングオブジェクトを呼び出すステートメントの行番号 (メインフレーム、UNIX および OpenVMS のみ)

『Natural スタジオの使用』の「技術情報」も参照してください。

**Notes:**

1. 文字アプリケーションのみ：アプリケーションの任意のポイントからこの情報を表示するには、端末コマンド %<TECH を使用できます。さらに、現在アクティブなプログラミングオブジェクトおよび上位レベルのすべてのオブジェクトの名前、タイプ、レベル情報もまだ有効です。
2. このコマンドは、リモートセッションでも使用できます。バッチモードですべての情報を読み取ることができます。

# 57 UNCATALOG

UNCATALOG [*object-name* ...]

このコマンドは、1つ以上のオブジェクトモジュールを削除します。

矛盾を防ぐために、メニューコマンド [削除] を使用し、オブジェクトのソースコードとオブジェクトモジュールの両方を削除することをお勧めします。『Natural スタジオの使用』の「オブジェクトの削除」を参照してください。

現在ログオンしているライブラリに保存されているオブジェクトだけを削除できます。UNCATALOG コマンドはソースワークエリアの内容には影響しません。

<b>UNCAT</b>	UNCATALOG コマンドを入力するときに <i>object-name</i> を指定しないか、またはアスタリスク (*) を1つ指定すると、現在のライブラリにカタログされているすべてのオブジェクトのリストが表示されます。リスト上で、削除するオブジェクトをマークできます。
<b>UNCAT *</b>	
<b>UNCAT <i>object name</i></b>	<i>object-name</i> として、削除するオブジェクトの名前を指定します。 複数のオブジェクトを削除する場合、 <i>object-names</i> は1つ以上の空白（または現在定義されているデリミタ文字）で区切る必要があります。 特定の文字列で始まるすべてのオブジェクトを削除する場合は、 <i>object-name</i> にアスタリスク (*) を使用します。選択されたすべてのオブジェクトを含むリストが表示されます。そのリストで、削除する各オブジェクトをマークできます。



**Note:** パラメータファイルに有効ではない FDIC システムファイルが指定されると、UNCATALOG コマンド発行時に Natural は該当するエラーメッセージを表示します。

---

# 58 UNLOCK

---

▪ Natural オブジェクトのロック解除 .....	160
▪ ドキュメンテーションオブジェクトのロック解除 .....	161
▪ パラメータの説明 .....	161
▪ パラメータ処理と見つかったオブジェクトの表示 .....	163

このコマンドは、次をロック解除します。

- リモート開発環境の Natural ソースオブジェクト
- Predict バージョン 4.4 以上が Windows にインストールされている場合に、ローカル開発環境のドキュメンテーションオブジェクト
- オブジェクト記述プラグインがインストールされている場合に（『オブジェクト記述』ドキュメントを参照）、リモート開発環境のドキュメンテーションオブジェクト

ロックされているソースオブジェクトまたはドキュメンテーションオブジェクトを表示したり、必要に応じてそれらをロック解除したりすることができます。このコマンドは Natural 管理者だけが使用することを推奨します。ただし、管理者は Natural Security の各ユーザープロファイルでこのコマンドの使用を有効にすることができます。

このchapterでは、次のトピックについて説明します。

詳細については、『SPoDを使用したリモート開発』ドキュメントの「*Unlocking Objects Manually*」を参照してください。

『Natural スタジオの使用』ドキュメントの「オブジェクトの命名規則」も参照してください。

## Natural オブジェクトのロック解除

パラメータを指定せずにシステムコマンド UNLOCK が使用された場合、パラメータを入力できるダイアログが表示されます。

```
UNLOCK
```

次に、Natural オブジェクトをロック解除するためのダイレクトコマンド構文を示します。

```
UNLOCK [NATURAL] [OBJECT] object-name
      [TYPE object-type]
      [LIBRARY library-name]
      [DBID dbid] [FNR fnr]
      [PASSWORD password] [CIPHER cipher]
      [APPLICATION application-name]
      [USER locked-by]
      [DATE locked-on [locked-on2]]
```

## ドキュメンテーションオブジェクトのロック解除

次に、ドキュメンテーションオブジェクトをロック解除するためのダイレクトコマンド構文を示します。

```
UNLOCK DOCUMENT [OBJECT] object-name
      [TYPE object-type]
      [USER locked-by]
      [DATE locked-on [locked-on2]]
```

## パラメータの説明

各ケースでオブジェクト名を定義する必要があります。その他のパラメータのいずれかが指定されていない場合は、対応するデフォルト値が使用されます。

パラメータ	フォーマット /長さ	デフォルト値	説明																								
<i>object-name</i>	A33	*	ロック解除するオブジェクトの名前。アスタリスク (*) 表記または ">" を使用できます。																								
<i>object-type</i>	A1	*	<p><b>Natural</b> オブジェクトタイプ：</p> <p><i>object-type</i> には、下に示すいずれかのオブジェクトタイプコードまたはアスタリスク (*) を指定できます。</p> <table border="1"> <tbody> <tr> <td>P</td> <td>プログラム</td> </tr> <tr> <td>4</td> <td>クラス</td> </tr> <tr> <td>N</td> <td>サブプログラム</td> </tr> <tr> <td>S</td> <td>サブルーチン</td> </tr> <tr> <td>7</td> <td>ファンクション</td> </tr> <tr> <td>8</td> <td>アダプタ</td> </tr> <tr> <td>C</td> <td>コピーコード</td> </tr> <tr> <td>H</td> <td>ヘルプルーチン</td> </tr> <tr> <td>T</td> <td>テキスト</td> </tr> <tr> <td>3</td> <td>ダイアログ</td> </tr> <tr> <td>M</td> <td>マップ</td> </tr> <tr> <td>L</td> <td>ローカルデータエリア</td> </tr> </tbody> </table>	P	プログラム	4	クラス	N	サブプログラム	S	サブルーチン	7	ファンクション	8	アダプタ	C	コピーコード	H	ヘルプルーチン	T	テキスト	3	ダイアログ	M	マップ	L	ローカルデータエリア
P	プログラム																										
4	クラス																										
N	サブプログラム																										
S	サブルーチン																										
7	ファンクション																										
8	アダプタ																										
C	コピーコード																										
H	ヘルプルーチン																										
T	テキスト																										
3	ダイアログ																										
M	マップ																										
L	ローカルデータエリア																										

パラメータ	フォーマット /長さ	デフォルト値	説明								
			<table border="1"> <tr> <td>G</td> <td>グローバルデータエリア</td> </tr> <tr> <td>A</td> <td>パラメータデータエリア</td> </tr> <tr> <td>V</td> <td>DDM (ビュー)</td> </tr> <tr> <td>X</td> <td>アプリケーション</td> </tr> </table>	G	グローバルデータエリア	A	パラメータデータエリア	V	DDM (ビュー)	X	アプリケーション
G	グローバルデータエリア										
A	パラメータデータエリア										
V	DDM (ビュー)										
X	アプリケーション										
	A2	*	<p>ドキュメンテーションオブジェクトのタイプ：</p> <p>ドキュメンテーションオブジェクトタイプに関するユーザー定義の短い説明またはアスタリスク (*)。</p>								
<i>library-name</i>	A8	*	ロックされたオブジェクトが置かれているライブラリの名前。アスタリスク (*) 表記を使用できます。								
<i>dbid</i>	A5	現在のデータベース ID	定義済みライブラリのデータベース ID アスタリスク (*) またはフォーマット N5 で指定します。								
<i>fnr</i>	A5	現在のファイル番号	定義済みライブラリのファイル番号アスタリスク (*) またはフォーマット N5 で指定します。								
<i>password</i>	A8	空白	<p>使用されている場合は、指定されたシステムファイルのパスワード (<i>dbid</i> および <i>fnr</i>)。</p> <p>現在の FNAT または FUSER の <i>dbid</i> および <i>fnr</i> を使用する場合は、指定する必要はありません。</p> <p>このパラメータは、メインフレームリモート開発環境で、プロファイルパラメータ SLOCK=PRE がメインフレーム環境に設定されている場合にのみ使用できます。</p>								
<i>cipher</i>	A8	空白	<p>使用されている場合は、指定されたシステムファイルのサイファキー (<i>dbid</i> および <i>fnr</i>)。</p> <p>現在の FNAT または FUSER の <i>dbid</i> および <i>fnr</i> を使用する場合は、指定する必要はありません。</p> <p>このパラメータは、メインフレームリモート開発環境で、プロファイルパラメータ SLOCK=PRE がメインフレーム環境に設定されている場合にのみ使用できます。</p>								
<i>application-name</i>	A32	空白	<p>使用されている場合は、ロックされたオブジェクトが属するアプリケーションの名前。</p> <p>空白を指定した場合、アプリケーションにリンクされているかどうかにかかわらず、ロックされているすべてのオブジェクトが結果ウィンドウに表示されます。このウィンドウで、ロックを手動で解除できます。</p>								

パラメータ	フォーマット /長さ	デフォルト値	説明
<i>locked-by</i>	A8	現在のユーザー ID  または、バッチモードとセキュリティフラグ "F" でのアスタリスク (*)	オブジェクトがロックされる原因となったユーザーのID。アスタリスク (*) 表記を使用できます。  Natural Security が使用される場合は、Natural Security ユーザープロファイルでセキュリティロック解除フラグが "F" (強制ロック解除) に設定されている場合にのみ変更できます。
<i>locked-on</i>	A10	空白	異なる日付フォーマットに対して2つの日付パラメータを使用できます。  2005-09-28 (DTFORMプロファイルパラメータに従った日付フォーマット) 2005-09-28 11:27:20 Today Today + nnnn Today - nnnn Yesterday
<i>locked-on2</i>	A8		



**Note:** ロックは、Natural for Mainframes Version 4.2 以上に基づくメインフレームサーバー上でローカルに有効にすることもできます。この場合は、次の制限が適用されます。  
*application-name* を選択基準として使用することはできません。*dbid* と *fnr* については、アスタリスク表記 (\*) が使用されている場合に、現在の FNAT および FUSER システムファイルが検索されます。

## パラメータ処理と見つかったオブジェクトの表示

指定されたパラメータが有効で、完全なオブジェクト名が指定されている場合、また、対応するオブジェクトが見つかり、現在のユーザーによってロックされていた場合は、このオブジェクトが即時にロック解除され、対応するメッセージが表示されます。このことは、オブジェクト名がアスタリスク表記 (\*) を使用せずに直接指定され、現在のユーザーが自分でロックしたレコードをロック解除しようとした場合に適用されます。

指定されたパラメータのいずれかが無効な場合、またはオブジェクトが見つからない場合は、エラーメッセージボックスのあるロック解除ダイアログが表示されます。

次のケースでは、見つかったロック済みのオブジェクトが結果ウィンドウにリストされます。ここで、手動でロック解除できます。

- アスタリスク表記 (\*) または ">" (該当する場合) を使用した場合。
- 特定のオブジェクト名を指定しなかった場合。

■ アプリケーション名を指定しなかった場合。

ドキュメンテーションオブジェクトのオブジェクトタイプが一意でない場合は、オブジェクトタイプの横の非表示の例で内部オブジェクトタイプを調べてください。

結果ウィンドウの詳細については、『SPoDを使用したリモート開発』ドキュメントの「*Unlocking Objects Manually*」を参照してください。

# 59 UNMAP

---

■ 現在アクティブな環境／アプリケーションのアンマップ .....	166
■ Natural 開発サーバー環境のアンマップ .....	166
■ Natural アプリケーションのアンマップ .....	166

UNMAP コマンドにより、Natural コマンド行を使用して次の機能を実行できます。

Natural 開発サーバーまたは Natural アプリケーションをマップするには、システムコマンド `MAP` を使用するか、『*SPoD*を使用したリモート開発』ドキュメントの「アプリケーションのマップ／アンマップ」で説明されているダイアログを使用します。

## 現在アクティブな環境／アプリケーションのアンマップ

---

次のコマンド構文は、現在アクティブな Natural 開発サーバー環境または Natural アプリケーションをアンマップする場合に適用されます。

```
UNMAP
```

## Natural 開発サーバー環境のアンマップ

---

次のコマンド構文は、Natural 開発サーバー環境をアンマップする場合に適用されます。

```
UNMAP ENVIRONMENT=environment-name
```

*environment-name* は、接続のエイリアス名です。環境名に空白が含まれる場合は、空白を一重引用符で囲む必要があります（'）。

## Natural アプリケーションのアンマップ

---

次のコマンド構文は、Natural アプリケーションをアンマップする場合に適用されます。

```
UNMAP APPLICATION=application-name
```

*application-name* は、アンマップするアプリケーションの名前です。

# 60 UNREGISTER

---

```
UNREGISTER { class-module-name } [ [ { library-name } [ server-id ] ] ]  
*
```

関連コマンド：[REGISTER](#)

Natural クラスの登録を解除します。

詳細については、『オペレーション』ドキュメントの「*NaturalX* アプリケーションの管理」部分にある「UNREGISTER コマンド」を参照してください。



**Note:** Natural Security 環境では、このコマンドは1つのライブラリにのみ呼び出すことができます。これは、ライブラリ名を省略するか、特定のライブラリを使用する必要があることを意味します。アスタリスク (\*) は使用できません。



# 61 UPDATE

---

UPDATE	{ ON OFF }
--------	---------------

このコマンドは、プログラムによって実行されるデータベースの更新を不可能（または可能）にするために使用します。

<b>UPDATE ON</b>	更新を可能にします。このコマンドは、Natural のインストール時に Natural 管理者が更新を不可能にした場合は無効です。
<b>UPDATE OFF</b>	UPDATE、STORE または DELETE の各ステートメントで通常実行される更新を不可能にします。これらのステートメントを持つプログラムは正常に実行されますが、データベースの更新は行われません。更新処理が検出されると、データベースを更新せずにメッセージが表示されます。

システムコマンド **CHECK** が UPDATE OFF とともに使用されると、エラーメッセージが表示されます。UPDATE コマンドは、他の Natural システムコマンドには影響しません。



## 62 XREF

---

XREF	ON
	OFF
	FORCE
	DOC

このコマンドは、Predictをインストールしている場合にのみ使用できます。Predictの"アクティブクロスリファレンス"機能の使用を制御します。

アクティブクロスリファレンス機能は、あるプログラム／データエリアを参照するオブジェクトに関して、データディクショナリ内に、自動的にドキュメントを作成します。これらのオブジェクトとして、プログラム、サブプログラム、サブルーチン、ヘルプルーチン、マップ、データエリア、データベースビュー、データベースフィールド、ユーザー定義変数、処理ルール、エラー番号、ワークファイル、プリンタ、クラスおよび保持 (RETAIN) ISN 集合があります。

アクティブクロスリファレンスは、プログラム／データエリアがカタログされたときに生成されます。

クロスリファレンスデータは、システムコマンド `LIST` の XREF オプションを使用して参照できます。

アクティブクロスリファレンスの詳細については、Predictのドキュメントを参照してください。

次のコマンドオプションを使用できます。

<b>XREF</b>	パラメータなしでXREFコマンドを入力すると、メニュー／ダイアログが表示されてオプションを指定できます。
<b>XREF ON</b>	アクティブクロスリファレンス機能を有効にします。Naturalプログラム／データエリアをカタログするたびに、クロスリファレンスデータが適切なPredictエントリに保存されます。
<b>XREF OFF</b>	アクティブクロスリファレンス機能を無効にします。クロスリファレンスデータは保存されません。カタログされるオブジェクトの既存のクロスリファレンスデータは削除されます。
<b>XREF FORCE</b>	オブジェクトをカタログ化できるのは、オブジェクトにPredictエントリが存在する場合のみです。オブジェクトがカタログされる時、そのクロスリファレンスデータはPredictに保存されます。Predictエントリが存在しない場合は、オブジェクトをカタログ化することはできません。
<b>XREF DOC</b>	オブジェクトをカタログ化できるのは、オブジェクトにPredictエントリが存在する場合のみです。ただし、オブジェクトがカタログされる時、そのクロスリファレンスデータはPredictに保存されず、オブジェクトの既存のクロスリファレンスデータは削除されます。Predictエントリが存在しない場合は、オブジェクトをカタログ化することはできません。

### Natural Security の考慮事項

Natural Security がインストールされている場合、XREF 設定はライブラリセキュリティプロファイルでライブラリごとに設定されます。セキュリティプロファイルによっては、XREF コマンドのいくつかのオプションを利用できないことがあります。

# 索引

---

## C

CATALL  
システムコマンド, 17  
CATALOG  
システムコマンド, 21  
CHECK  
システムコマンド, 23  
CLEAR  
システムコマンド, 25  
COMPOPT  
システムコマンド, 27

## D

DEBUG  
システムコマンド, 45

## E

EDIT  
システムコマンド, 47  
EXECUTE  
システムコマンド, 53

## F

FIN  
システムコマンド, 57

## G

GLOBALS  
システムコマンド, 59

## H

HELP  
システムコマンド, 63

## I

INPL  
システムコマンド, 65

## L

LAST

システムコマンド, 67  
LASTMSG  
システムコマンド, 69  
LIST  
システムコマンド, 71  
LIST COUNT  
システムコマンド, 77  
LIST XREF  
システムコマンド, 79  
LOGOFF  
システムコマンド, 81  
LOGON  
システムコマンド, 83

## M

MAIL  
システムコマンド, 85  
MAP  
システムコマンド, 87

## P

PROFILE  
システムコマンド, 91  
PURGE  
システムコマンド, 93

## R

READ  
システムコマンド, 95  
REGISTER  
システムコマンド, 97  
RENAME  
システムコマンド, 99  
RENUMBER  
システムコマンド, 101  
RETURN  
システムコマンド, 103  
RUN  
システムコマンド, 105

## S

SAVE  
システムコマンド, 107  
SCAN  
システムコマンド, 109

SCRATCH  
システムコマンド, 111

SETUP  
システムコマンド, 113

STOW  
システムコマンド, 117

STRUCT  
システムコマンド, 119

SYSAPI  
システムコマンド, 123

SYSCP  
システムコマンド, 125

SYSERR  
システムコマンド, 127

SYSEXT  
システムコマンド, 129

SYSEXV  
システムコマンド, 131

SYSFILE  
システムコマンド, 133

SYSINST  
システムコマンド, 135

SYSMAIN  
システムコマンド, 137

SYSMN  
システムコマンド, 139

SYSNCP  
システムコマンド, 141

SYSOBJH  
システムコマンド, 143

SYSPROD  
システムコマンド, 145

SYSPROF  
システムコマンド, 147

SYSRPC  
システムコマンド, 149

SYSWIZDB  
システムコマンド, 151

SYSWIZDW  
システムコマンド, 153

## T

TECH  
システムコマンド, 155

## U

UNCATALOG  
システムコマンド, 157

UNLOCK  
システムコマンド, 159

UNMAP  
システムコマンド, 165

UNREGISTER  
システムコマンド, 167

UPDATE  
システムコマンド, 169

## X

XREF  
システムコマンド, 171

## し

システムコマンド, 1