Natural for Mainframes

システムコマンド

バージョン 4.2.5

October 2009

Natural

This document applies to Natural バージョン 4.2.5 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.

目次

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

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

コマンドは、コマンドプロンプト（===>）のコマンド行に入力できます。

また、一部のシステムコマンドは、PF キーまたはメインメニューからも使用できます。

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

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。
ABCDEF	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。
ABCDEF	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、下線部分がその用語の入力可能な省略形です。
abcdef	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。 例外：

要素	説明
	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

この章では、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）。
MAINMENU	Natural メインメニューモードのオン／オフの切り替え、またはユーザー定義メニューを作成するプログラムの呼び出しを行います。
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。
GLOBALS	Natural セッションパラメータの設定を変更します。
KEY	Natural セッションで使用するキーに、機能を割り当てます。
SYSCP	コードページ情報を提供します。また、Natural ソースオブジェクトのコードページの管理にも使用できます。
SYSPARM	Natural プロファイルパラメータの設定を変更します。
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。

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	エディタのワークエリアの内容を消去します。
DELETE	Natural システムファイルからプログラミングオブジェクトを削除します。 ソース形式、オブジェクト形式、またはその両方のプログラミングオブジェクトを削除できます。
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。
RENAME	Natural プログラミングオブジェクトに別の名前を指定します。
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。
STRUCT	ソースプログラムの構造的なインデントを実行します。これは、構造的な不整合の発見に役立ちます。

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
SYSDDM	Natural データ定義モジュール（DDM）を作成およびメンテナンスします。
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
BUS	SYSTP ユーティリティのサービスディレクトリメンテナンス機能を呼び出します。 BUS では、Natural バッファの使用状況に関する情報が提供されます。 **注意:** SYSBUS は BUS で置き換えられたため、使用できなくなりました。
DUMP	Naturalの異常終了（アベンド）を発生させたエラーを特定するための情報を、Software AG 技術サポート担当者向けに提供します。

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。
SYSADA (ADACALL)	ライブラリ SYSADA にある ADACALL ユーティリティを呼び出します。 ADACALL を使用すると、Adabas データベースに Adabas ダイレクトコール（ネイティブコマンド）を直接発行できます。
SYSBPM	SYSBPMユーティリティを呼び出します。このユーティリティは、バッファプールをモニタリングし、要求に合わせて調整するために使用します。
SYSEDT	Editorバッファプールサービス SYSEDT を呼び出します。このユーティリティは、バッファプールのランタイム情報の表示、そのパラメータの変更、および論理ワークファイルとリカバリファイルの削除に使用します。
SYSTP	SYSTP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すると、NaturalのTP モニタに固有のさまざまな特性をモニタリングおよび制御できます。
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。
TEST	オンラインテストおよびデバッグ用の Natural デバッガを呼び出します。 Natural デバッガを使用すると、アプリケーションのさまざまな面をテストして、プログラムの処理フローにあるエラーを特定することができます。
TEST DBLOG	データベースコールをロギングするための Natural DBLOG ユーティリティを呼び出します。

SQL データベースで使用するコマンド

コマンド	機能
LISTSQL	DB2 版 Natural および SQL/DS 版 Natural でのみ使用できます。 データベースアクセスに関係するプログラミングオブジェクト内にあるこれらのNaturalステートメント、およびステートメントが変換された SQL コマンドをリストします。
SQLERR	DB2 版 Natural および SQL/DS 版 Natural でのみ使用できます。 SQL エラーに関する情報を表示します。
SYSDB2	Natural for DB がインストールされている場合、このコマンドによって Natural Tools for DB2 が呼び出されます。 詳細については、『*データベース管理システムインターフェイス*』ドキュメントの「*Natural for DB*」で説明されている「*Natural Tools for DB2*」を参照してください。

Natural Optimizer Compiler で使用するコマンド

次のシステムコマンドは、Natural Optimizer Compiler がインストールされている場合にのみ使用できます。 詳細については、『*Natural Optimizer Compiler*』ドキュメントを参照してください。

コマンド	機能
NOCOPT	Natural の起動中に指定された Natural Optimizer Compiler オプションの現在の設定を表示し、これらの設定を変更できます。
NOCSHOW	PGEN オプションによって生成された出力のバッファ情報を提供します。 PGEN オプションを指定すると、Natural Optimizer Compiler は、生成されたコードおよび内部 Natural 構造を出力します。
NOCSTAT	Natural Optimizer Compiler での処理に適しているプログラムの統計データを提供します。

その他のコマンド

コマンド	機能
AIV	定義されているすべてのアプリケーション独立変数（AIV）を表示します。
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。
ROUTINES	現在のライブラリ内にあるどのオブジェクトがどの外部サブルーチンを使用しているかを表示します。
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。
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。

コマンド	機能
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 AIV

AIV

このコマンドは、現在アクティブになっている、アプリケーションに依存しない変数（AIV）をすべて表示するために使用します。

表示されるリストで、AIV をコマンド DI でマークすると、その AIV の内容を表示できます。

内容は、英数字形式または 16 進形式で表示できます。 英数字形式と 16 進形式を切り替えるには、PF10 キーと PF11 キーを使用します。

詳細については、次のセクションを参照してください。

- 『*ステートメント*』ドキュメントの DEFINE DATA ステートメント（「*アプリケーションに依存しない変数の定義*」）
- 『*プログラミングガイド*』の「*ユーザー定義変数*」

6 BUS

BUS

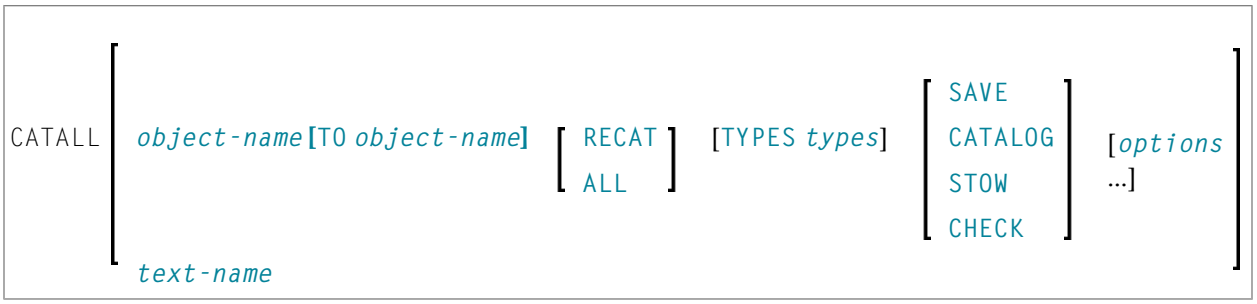
このコマンドは、SYSTPユーティリティのバッファ使用統計機能を呼び出すために使用します。現在のNaturalセッションに割り当てられているバッファ、そのサイズ、および実際に使用されているバッファスペースに関する情報が表示されます。

注意: BUS コマンドは、SYSBUS コマンド（現在は使用不可）と同じ機能を提供します。

詳細については、『*ユーティリティ*』ドキュメントの「*バッファ使用統計*」を参照してください。

アプリケーションプログラミングインターフェイス：USR1019N。『*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。

7 CATALL



このコマンドは、現在のライブラリに存在するソース形式、オブジェクト形式、またはその両方のすべてのオブジェクトを保存するために使用します。

追加オプションを付けずに CATALL コマンドを入力すると、［**Catalog Objects in Library**］画面が表示されます。この画面を使用して、次の機能を実行できます。 上記のコマンド構文を使用して、CATALL コマンドを直接発行することもできます。

CATALLU2 サブプログラムを使用することによって、［**Catalog Objects in Library**］画面の機能をデフォルトで選択することもできます。 また、CATALLU2 をバッチモードまたはコマンドモードで呼び出すことを可能にすることもできます。 このサブプログラムは、ライブラリ SYSTEM（FNAT）にソース形式で提供されています。 サブプログラムをアクティブにするには、ソースで記述されているとおりに変更してから、カタログして SYSLIB にコピーします。 このサブプログラムは、［**Catalog Objects in Library**］画面が出力される前に呼び出されます。

『*Natural 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。

カタログするオブジェクトの指定（［Catalog Objects from/to］フィールド）

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に対して CATALL を実行するには、［**from**］フィールドにアスタリスク（*）をオブジェクト名として指定します。

特定範囲のオブジェクト群に対して CATALL を実行するには、システムコマンド LIST の説明と同様に、［**from**］フィールドの名前にアスタリスク表記（*）やワイルドカード表記（?）を使用できます。

または、［**from**］フィールドと［**to**］フィールドに、アスタリスク表記やワイルドカード表記を使用せずに該当するオブジェクト名を入力することによって、特定範囲のオブジェクト群の始まりと終わりを指定することもできます。

これらのフィールドに設定を入力する代わりに、選択リストからオブジェクトを選択することもできます。

さらに、［**from**］フィールドを使用して、CATALL コマンドのリストを含めたテキストタイプのオブジェクトの名前を入力することもできます。 このテキストに含まれる CATALL コマンドが実

行されます。 このようなテキストは、手動で作成したり、選択リストの使用時に自動的に作成したりすることもできます。

［Recatalog Only Existing Modules］フィールドと［Catalog All Sources］フィールド

このオプションは、CATALOG 機能および STOW 機能にのみ適用されます。

- 上記2つのうち最初のフィールドをマークすると、現在のライブラリにすでにオブジェクトモジュールが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カタログされません。
- 2つ目のフィールドをマークすると、選択されているすべてのオブジェクトがカタログされます。

注意: このオプションは、コピーコードタイプおよびテキストタイプのオブジェクトには適用されません。

オブジェクトタイプの選択

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。したがって、すべてのオブジェクトタイプが X でマークされています。

特定の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 X を空白で上書きします。

機能の選択

選択したオブジェクトに適用する機能を、SAVE、CATALOG、STOW、または CHECK から 1 つ選択できます。 これらの機能は、同名のシステムコマンドに対応します。

注意: STOW を選択した場合でも、コピーコードタイプとテキストタイプのオブジェクトは保存されます。 CATALOG を選択した場合には、これらのオブジェクトは保存されません。

オプションの選択

CATALL 処理に関して、次のオプションを 1 つ以上選択できます。

Condition Code in Batch	CATALL をバッチモードで実行し、このオプションを文字でマークすると、CATALL の実行中に構文エラーが検出されるか、または指定したオブジェクト範囲にオブジェクトが見つからない場合に、コンディションコード 55 が返されます。このオプションは CATALOG と STOW のみに適用されます。
Renumber Source-Code Lines	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行わない場合は、このフィールドの X を空白で上書きします。
Keep Result List	CATALL では結果リストが生成されます。 結果リストを維持して後で使用するには、このフィールドを文字でマークします。 ライブラリ SYSEXT には、結果リストの出力に使用できるアプリケーションプログラミングインターフェイス USR1024N が含まれています。 結果リストは、別の CATALL コマンドを使用して再表示することもできます。結果リストにはパラメータも保存されているため、結果リストが生成された CATALL のパラメータは有効です。バッチ処理でライブラリに結果リストが含まれている場合、この結果リストは CATALL コマンドで自動的に表示されます。 この場合は、バッチジョブ CATALL によってメッセージが発行されますが、モジュールがカタログされないため、ジョブはコンディションコード 56 で終了します。 オンライン処理でライブラリに結果リストが含まれている場合は、以前の結果リストを表示するか、または新しい CATALL の実行を開始するかを尋ねられます。
Processing Information	オンライン処理中は、処理ステータス情報が CATALL によってスクロール表示されます。 バッチ処理中は、エラーの原因となったモジュールのみが CATALL によって出力されます。 これらの表示を抑制するには、このフィールドの X を空白で上書きします。
Error Report	処理の最後に、発生したエラーのリストが CATALL によって表示されます。 エラーリストの表示を抑制するには、このフィールドの X を空白で上書きします。
Extended Error Report	エラーレポートが、ディレクトリ情報、エラーの行、エラーメッセージを示した拡張形式で出力されます。 拡張エラーレポートを出力するには、このフィールドを X でマークします。
PF4 AddOp	PF4 キーを押すと、追加オプションを選択または入力できるウィンドウが表示されます。 Error Text Member：Natural テキストメンバの名前を入力します。 CATALL 実行のエラーリストが、このテキストメンバに書き込まれます。

選択リスト

特定のオブジェクトのみに CATALL を使用する場合は、該当するオブジェクトを選択リストから選択できます。

このためには、最初に［Select Function］および［Select Options］で希望する項目を選択してから、PF5 キーを押します。現在のライブラリに保存されているオブジェクトのリストが表示されます。

このリストは、システムコマンド LIST のリストに対応します。選択リストのスクロールや、リストの新規選択条件の指定も、LIST コマンドの場合と同じ方法で行います。

リストの［Cmd］列を文字でマークすることにより、希望するオブジェクトを選択します。現在の選択リストのすべてのオブジェクトを同時に選択するには、PF5 キーを押します。次に、リストをスクロールして他の選択条件を指定し、オブジェクトをさらに選択します。

処理対象にするオブジェクトをすべて選択した後、PF3 キーを押します。

ウィンドウが表示されます。ここでは、選択したオブジェクトの集合を他の CATALL 処理で再利用するために保存することができます。次のいずれかの手順に従います。

- ウィンドウで名前を入力すると、選択したオブジェクトの集合が CATALL コマンドの形式で、入力した名前のテキストタイプのオブジェクトに自動的に保存されます。このテキスト名は、［**Catalog Objects in Library**］画面の［**Catalog Objects from**］フィールドで今後使用できます。
- 保存しない場合は、ウィンドウに何も入力しないで Enter キーを押します。

選択したオブジェクトの CATALL による処理が開始されます。

ダイレクトコマンド構文

［**Catalog Objects in Library**］画面で実行可能な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。

object-name **TO** *object-name*	［**Catalog Objects in Library**］画面の［**Catalog Objects from**］フィールドと［**Catalog Objects to**］フィールドに対応します。「*カタログするオブジェクトの指定（［Catalog Objects from/to］フィールド）*」を参照してください。
RECAT/ALL	［**Catalog Objects in Library**］画面の［**Recatalog Only Existing Modules**］オプションまたは［**Catalog All Sources**］オプションに対応します。RECAT はデフォルトです。「*［Recatalog Only Existing Modules］フィールドと［Catalog All Sources］フィールド*」を参照してください。

TYPES *types*	［**Catalog Objects in Library**］画面でマークされた各オブジェクトタイプに対応します。「*オブジェクトタイプの選択*」を参照してください。 可能な *types*： 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 C コピーコード T テキスト 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 4 クラス * すべてのタイプ（デフォルト） *types* 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。
SAVE/CATALOG/STOW/CHECK	［**Catalog Objects in Library**］画面の同名のアクションに対応します。「*機能の選択*」を参照してください。 CATALOG はデフォルトです。

options	次のオプションは、［**Catalog Objects in Library**］画面の［**Select Options**］に対応します。「*オプションの選択*」を参照してください。可能な *options*： CC：コンディションコードを返します。 NOREN：ソースコード行の自動番号再設定を行いません。 KEEP：結果リストを維持します。 NOSCROLL：オンライン処理の場合は、処理ステータス情報のスクロール表示を抑制します。バッチ処理の場合は、エラーの原因となったモジュールのみの出力を抑制します。 NOREPORT：エラーレポートを抑制します。 FULL：エラーレポートを拡張します。 EL <*text-member*> [R] EL <*text-member*>：エラーリストを Natural テキストメンバに出力します。 R：すでにメンバが存在する場合は、<*text-member*> の後に R（置き換え）が指定されない限り、EL パラメータは無効になります。 **注意:** NOREPORT と NOSCROLL を*両方とも*指定すると、KEEP も自動的に適用されます。
text-name	［**Catalog Objects in Library**］画面の［**Catalog Objects from**］フィールドのテキスト名の指定に対応します。「*カタログするオブジェクトの指定（［Catalog Objects from/to］フィールド）*」を参照してください。

例：

手順 7.1. 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を格納するには

■ 次のコマンドを入力します。

```
CATALL * STOW KEEP CC NOREN
```

このコマンドには暗黙的な RECAT が含まれるため、次のコマンドと同じ効果があります。

```
CATALL * RECAT STOW KEEP CC NOREN
```

手順 7.2. すべてのオブジェクトを格納するには

■ 次のコマンドを入力します。

```
CATALL * ALL STOW KEEP CC NOREN
```

注意: 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。

8 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。

以下の項目も参照してください。

『*Natural システムアーキテクチャ*』ドキュメントの「*Natural コンパイラ*」
『*Natural の使用*』ドキュメントの「*オブジェクトの命名規則*」

重要: 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。

CATALOG	*object-name* 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現在のライブラリにカタログされます。 既存のオブジェクトコードは置き換えられます。
CATALOG *object-name*	新しいオブジェクトが作成されます。 *object-name* には、新しいオブジェクトがカタログされるとき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。
CATALOG *object-name* *library-id*	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 *library-id* 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。

注意: 有効ではないパラメータモジュールに 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。

9 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。

モード	動作
オンライン	構文エラーが検出されると、構文チェックは停止します。対応するエディタでは、エラーが含まれるソースコード行が E でマークされます。
バッチ	すべてのステートメントのチェックが完了するまで構文チェックは継続します。検出されたすべてのエラーは出力リストに記載されます。

注意: 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* システムアーキテクチャ』ドキュメントの「*Natural* コンパイラ」も参照してください。

10 CLEAR

CLEAR

このコマンドは、エディタのソースワークエリアをクリアするために使用します。 新しいプログラムを作成するときに、ソースワークエリアに別のオブジェクトがある場合に使用することができます。

11 CMS

このコマンドは VM/CMS 環境にのみ適用されます。このコマンドを使用すると、CMS コマンドを Natural 内部から実行できます。

```
CMS CMS-command
```

CMS-command の代わりに指定されたこのコマンドが CMS コマンドプロセッサに渡され、通常の CMS 対話式コマンドモードの場合とまったく同様の完全な CMS コマンド解決が実行されます。

CMS の詳細については、『*オペレーション*』ドキュメントの *VM/CMS 環境での Natural* に関するセクションを参照してください。

12 COMPOPT

```
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パラメータモジュールまたはプロファイルパラメータ CMPO 内のパラメータマクロ NTCMPO の対応するキーワードサブパラメータを使用して設定されます。 ライブラリを変更すると、COMPOPT の各オプションはそれぞれのデフォルト値にリセットされます。

構文説明

COMPOPT	オプションなしで COMPOPT システムコマンドを発行すると、［**Compilations Options**］画面が表示されます。 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。
COMPOPT *option=value*	個々のオプションのキーワードについては、次で説明します。 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 のいずれかまたはその両方で設定されたデフォルト設定に戻ります。

コンパイラキーワードパラメータの指定

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 *option=value*）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

- 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
- CHKRULE - マップ内の INCDIR ステートメントの検証

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。

KCHECK - キーワードチェック

ON	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。
OFF	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

ON

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、および配列の添字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。

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。

このチェックを行うには、次の条件が必須です。

- 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。
- 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。

これに該当しない場合、PCHECK=ON には何の効力もありません。

CATALL コマンドを PCHECK=ON で使用する場合の問題点

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。

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。

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。

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になります。 解決策： ■ コンパイラオプション PCHECK=OFF を設定します。 ■ 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。 ■ 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。 ■ 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。 ■ コンパイラオプション PCHECK=ON を設定します。 ■ 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。
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

DDM に定義されるデータベースフィールドは、次の 2 種類の名前で記述されます。

- ショートネーム。長さ 2 文字で、Natural によってデータベース（特に Adabas）との通信に使用されます。
- ロングネーム。長さ 3～32 文字（基盤となるアクセス対象データベースタイプが DB2/SQL の場合は 1～32 文字）で、Natural プログラミングコード内のフィールドへの参照に使用されます。

特別な状況下では、Naturalプログラム内のデータベースフィールドへの参照に、ロングネームの代わりにショートネームを使用できます。このことは、Natural Security なしでレポーティングモードで実行している場合、およびデータベースアクセスステートメントにビューへの参照ではなく DDM への参照が含まれている場合に当てはまります。

フィールド名がショートネーム参照とみなされるかどうかは、名前の長さによって判断されます。 フィールド識別子が 2 文字の場合は、ショートネーム参照とみなされます。それ以外の長さのフィールド名は、ロングネーム参照とみなされます。 データベースフィールドに関するこのような標準解釈ルールを追加で指定および制御するには、コンパイラオプション DBSHORT を ON または OFF に設定します。

ON	ショートネームは、データベースフィールドの参照に使用できます。 ただし、DBSHORT=ONが設定されている場合でも、次のような場合には、通常、データベースショートネームは*使用できません*。 ■ ビューを作成するとき、フィールドの定義に使用する場合 ■ ビューフィールドがプログラミングコード内で使用される場合 ■ 以前、変数を定義するために DEFINE DATA LOCAL ステートメントが使用された場合 ■ Natural Security で実行されている場合 これはデフォルト値です。
OFF	データベースフィールドの参照には、ロングネームのみを使用できます。 各データベースのフィールド識別子は、文字数に関係なく、ロングネームの参照であるとみなされます。 2 文字の名前が指定されており、この名前がロングネームではなくショートネームとしてのみ見つかる場合は、コンパイル時に構文エラー NAT0981 が発生します。 これにより、DDMに定義されている識別子長が2バイトのロングネームを使用できるようになります。 この DDM を使用してアクセスする、基盤となるデータベースが SQL（DB2）であり、かつ、名前の長さが 2 文字のテーブル列が存在する場合、このオプションを指定する必要があります。 ただし、Adabas など、他のすべてのデータベースタイプについては、名前の長さが2バイトのロングフィールドを定義しようすると、DDM の生成時に拒否されます。 また、ショートネーム参照が使用されていない場合（DBSHORT=OFF によって強制的に設定できます）、プログラムは Natural Security でコンパイルされたかどうかに無関係になります。

例：

DDM EMPLOYEES に以下のデータベースフィールド定義があるとします。

Short Name	Long Name
AA	PERSONNEL-ID

例 1：

```
OPTIONS DBSHORT=ON
READ EMPLOYEES
  DISPLAY AA      /* data base short name AA is allowed
END
```

例 2：

```
OPTIONS DBSHORT=OFF
READ EMPLOYEES
  DISPLAY AA      /* syntax error NAT0981, because DBSHORT=OFF
END
```

例 3：

```
OPTIONS DBSHORT=ON
DEFINE DATA LOCAL
1 V1 VIEW OF EMPLOYEES
  2  PERSONNEL-ID
END-DEFINE
READ V1 BY PERSONNEL-ID
  DISPLAY AA     /* syntax error NAT0981, because PERSONNEL-ID is defined in view;
                 /* (even if DBSHORT=ON)
END-READ
END
```

PSIGNF - パック十進数の正記号の内部表現

ON	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。
OFF	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。

TSENABL - 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。

TSENABL=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。

ON	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。
OFF	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に適用されます。 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。

ON	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。
VID	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。
OFF	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=abccddee
```

要素	説明
a	x'F9'
b	x'22' または x'21'（DB ステートメントに依存）
cc	物理データベース番号（2 バイト）
dd	物理ファイル番号（2 バイト）
ee	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
bbbbb	STOD 値の 2～6 バイト
c	物理ファイル番号 - 下位バイト

- GFID=*abbbbbb*：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

注意: 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。

ON	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。
OFF	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を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、およびシルコンフレクス（^）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をアポストロフィ（'）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に出力されます。

```
      #Vari
  --------------------

  ABC
```

LOWSRCE=ON の場合、上記の例は次のように出力されます。

```
      #VARI
  --------------------

  ABC
```

TQMARK - 引用符の変換

ON	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。
OFF	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、次のように出力されます。

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

TQMARK OFF の場合、次のように出力されます。

```
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／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。

ON	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。
OFF	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 とコードページのサポート*』ドキュメントの「*CPAGE コンパイラオプション*」も参照してください。

ON	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。
OFF	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。

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

CHKRULE - マップ内の INCDIR ステートメントの検証

CHKRULE オプションを使用すると、マップのカタログ処理中の検証を有効または無効にすることができます。

ON	INCDIR 検証が有効になります。 INCDIR 制御ステートメント内で参照されているファイル（DDM）またはフィールドが存在しない場合は、コンパイル時に構文エラー NAT0721 が発生します。 Natural マップを作成するとき、別の既存のプログラミングオブジェクト内にすでに定義されているフィールドを含めることができます。 このことは、変数を定義できるほとんどすべてのオブジェクト、および DDM で可能です。 含められたフィールドがデータベース変数である場合は、マップエディタの組み込み動作によって、（含められたフィールド以外に）追加の INCDIR ステートメントがマップステートメントの本文に自動的に追加されます。これにより、マップをコンパイル（STOW）するとき、Predict ルールのアップロードと組み込みがトリガされます。 この機能は、INCLUDE ステートメントが処理されるときの動作と類似しています。 ただし、ソース行は、コピーコードオブジェクトから取得する代わりに Predict から受け取ります。 ルールを検索する場合は、DDM 名（ファイル名とみなされる）とフィールド名が検索キーとなります。 いずれも INCDIR ステートメントに指定されます。 コンパイル時に要求される INCDIR ルールは存在する必要はないため、必ずしも Predict で見つかる必要はありません。 つまり、検索されたルールが見つからなくてもエラーではありません。 フィールドが DDM からマップに組み込まれると、対応する INCDIR ステートメントが作成され、現在の DDM とフィールド名が Predict の既存のルールを要求するための 「検索キー」 として含められます。 ただし、コピープロセスの後に DDM の名前が変更された場合は、有効でなくなった古い DDM 名が引き続き INCDIR ステートメントで使用されます。 この場合、ルールはロードされず、このことはプログラマに通知されません。 また、この状況は、DDM の名前を変更した場合以外にも発生します。 このことが最も一般的に発生するのは、何らかのミスによって不適切な FDIC ファイルが割り当てられた場合です。 この場合、DDM 名は有効ですが、現在の Predict システムファイルでその名前は見つかりません。 この結果、DDM がまったく存在しない場合と同様、Predict から追加されるはずの処理ルールが含まれなくなります。

OFF	INCDIR 検証は無効になります。 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

- 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。

ON	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。
OFF	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ィ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ィールドを表します。

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と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と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に処理されます。 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

⚠ **重要:** 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。

ON	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（下記参照）を出力します。
OFF	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	可能	不明	002

機能	バージョン 4.2	バージョン 4.1	理由コード

X-array の例：
 	
`DEFINE DATA LOCAL`
`1 #ARR (A10/1:*)`			

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：
 	
`*PARSE-COL`
`*PARSE-LEVEL`
`*PARSE-NAMESPACE-URI`
`*PARSE-ROW`
`*PARSE-TYPE`
`*CODEPAGE`
`*LOCALE`
`*TYPE`
`*CURRENT-UNIT`
`*UBOUND`
`*LBOUND`
	可能	不明	007

未使用	-	-	008
`INCLUDE` で提供されるソースパラメータの長さとタイプ	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

機能	バージョン 4.2	バージョン 4.1	理由コード

例：
 	
`INCLUDE COPY01 'WRITE *LINE'`
`'WRITE *PROGRAM'`			

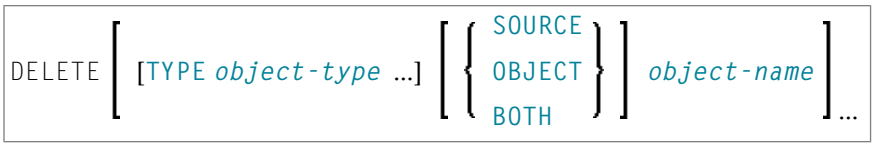
データビューでの Adabas LA フィールドにおける次のいずれかの定義 ■ 253 バイトを超えるサイズ ■ タイプ DYNAMIC	可能	不明	010

13 CPINFO

このコマンドは、システム変数 *LOCALE および *CODEPAGE の内容、ソースエリアの現在のコードページ、関連パラメータの現在の設定、NATICU バージョン、Unicode バージョンなど、関係のあるすべての Natural コードページ設定の表示、および NATCONFG モジュールに定義されているコードページの表示に使用します。 詳細な説明については、該当する Natural ヘルプテキストを参照してください。

詳細については、『*Unicode およびコードページのサポート*』ドキュメントを参照してください。

14 DELETE



このコマンドは、Natural システムファイルから Natural プログラミングオブジェクトを削除するために使用します。

注意: 現在エディタのワークエリアにあるソースは、DELETE コマンドの影響を受けません。

『*Natural*』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。

構文説明

object-name	*object-name* として、削除するオブジェクトの名前を指定します。 また、オブジェクトのソースコードおよびオブジェクトモジュールのいずれかあるいはその両方を削除するかどうかを次のように指定できます。 SOURCE：ソースコード OBJECT：オブジェクトモジュール BOTH：ソースコードとオブジェクトモジュールの両方 これがデフォルトです。 SOURCE/OBJECT/BOTH の指定は、以降のすべてのオブジェクト名、つまり SOURCE/OBJECT/BOTH が次に指定されるまで適用されます。 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、*object-name*にアスタリスク表記を使用します。
object-type	*object-name* のアスタリスク表記に加えて、特定タイプのオブジェクトのみを削除する場合は *object-type* も指定できます。 *object-type* に可能な設定は、システムコマンド EDIT に対する設定と同じです。 さらに、X 設定（= グローバル、ローカル、パラメータの各データエリア）と U 設定（= サブプログラム、サブルーチン、ヘルプルーチン）を指定できます。 **注意:** 個々のオブジェクトを完全な名前で指定する場合は、オブジェクトタイプを指定する必要はありません。

選択リスト

アスタリスク表記を使用すると、選択リストが表示されるので、削除するオブジェクトをマークします。 各オブジェクトについて、ソースコード、オブジェクトモジュール、またはその両方を削除するかどうかを、該当する文字（S、O、B）で指定できます。

DELETE コマンドのみを入力した場合でも、現在のライブラリに保存されているすべてのオブジェクトが記載された選択リストが表示されます。

誤削除に対する保護

誤って削除した場合の保護として、名前の入力によるオブジェクトの削除を確認するウィンドウが自動的に表示されます。

複数のオブジェクトを指定または選択している場合は、追加のウィンドウが表示され、各オブジェクトの削除を個別に確認するか、あるいは指定または選択したすべてのオブジェクトを確認せずに削除するかを指定できます。

例

次のコマンドでは、TOM、DICK、HARRY という名前の 3 つのプログラミングオブジェクトが削除されます。

```
DELETE TOM DICK HARRY
```

次のコマンドでは、プログラミングオブジェクト JOHN のソースとオブジェクトモジュール、プログラミングオブジェクト PAUL と GEORGE のソース、およびプログラミングオブジェクト RINGO のオブジェクトモジュールが削除されます。

```
DELETE JOHN SOURCE PAUL GEORGE OBJECT RINGO
```

次のコマンドでは、現在のライブラリに存在するすべてのプログラミングオブジェクトの選択リストが表示されます。

```
DELETE
```

次のコマンドでは、現在のライブラリに存在するすべてのマップのソースの選択リストが表示されます。

```
DELETE TYPE M SOURCE *
```

次のコマンドでは、現在のライブラリに存在し、ソースまたはオブジェクト形式で保存され、名前が D で始まるグローバル、ローカル、パラメータのすべてのデータエリアの選択リストが表示されます。

```
DELETE TYPE GLA D*
```

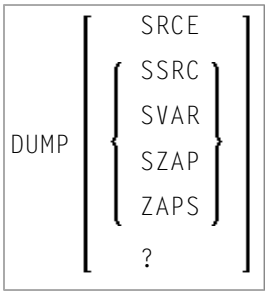
次のコマンドでは、現在のライブラリに存在し、オブジェクト形式で保存され、名前が "YYZ" で始まるすべてのプログラミングオブジェクトの選択リストが表示されます。

```
DELETE OBJECT YYZ*
```

次のコマンドでは、マップ TOM と DICK のソースとオブジェクトモジュール、マップ HARRY のソース、プログラム JOHN のソース、およびプログラム PAUL のオブジェクトモジュールが削除されます。

```
DELETE TYPE M TOM DICK SOURCE HARRY TYPE P JOHN OBJECT PAUL
```

15 DUMP



このコマンドは、Naturalシステムの異常終了（アベンド）の原因となったエラーを特定する目的で Software AG 技術サポートの担当者に情報を提供するために使用します。 この情報を Software AG 技術サポートに転送して、エラー診断と修正を行ってください。

DUMP	アベンド情報（コアコンテンツ）を表示します。
DUMP SRCE	製品ごとに適用されたソース変更の一覧を表示します。
DUMP SSRC	製品ごとに適用された特殊なソース変更の一覧を表示します。
DUMP SVAR	TP モニタとオペレーティングシステムに依存するシステム変数、および追加情報を表示します。
DUMP SZAP	適用されたすべての特殊 ZAP のリストを表示します。
DUMP ZAPS	適用されたすべての ZAP のリストを表示します。
DUMP ?	DUMP コマンドのその他のオプションが表示されます。DUMP メニューで疑問符（?）を入力したときに表示されるヘルプ画面の説明と同じです。 エラー診断での必要に応じて、Software AG 技術サポートの担当者が、このオプションを使用する場合と方法を指示します。

16 EDIT

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。 これについては、以降のドキュメントで説明されています。

関連コマンド：READ。

『*Natural*』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。

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
  GLOBAL
  HELPROUTINE
  LOCAL
{ MAP                }
  PARAMETER
  PROGRAM
( { SUBPROGRAM } )
( { N          } )
  SUBROUTINE
  TEXT
```

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。
- クラスはプログラムエディタで編集します。
- 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、、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。

注意: 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。

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。

注意: EDIT DESCRIPTION では、*object-name* は Predict プログラム定義の Natural メンバとして定義されている名前にする必要があります。

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。

注意: *library-id* の設定値を "SYS"（SYSTEM を除く）で始めることはできません。

Natural Security 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 *object-type* *	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。

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
```

17 EDT

EDT コマンドの代わりに EDIT コマンドを使用することをお勧めします。

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

このコマンドは、Naturalラインエディタを呼び出して編集モードに入ります。編集モードは、既存のNaturalオブジェクト（プログラム、コピーコード、サブルーチン、サブプログラム、ヘルプルーチン）の編集に使用できます。編集モードに入ると、次に示すサブコマンドやPFキーを使用して、任意の行に移動して変更することができます。

EDT モードを終了するには .E を使用します。

構文説明

object-name	*object-name* には、編集するオブジェクトの名前を最大 8 文字で入力します。 *object-name* を入力すると、このオブジェクトが Natural によってソース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。 オブジェクト名なしで EDT コマンドを入力し、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、行 0010 でオブジェクトを入力するように求められます。 オブジェクト名を指定せず、ソースプログラムワークエリアにオブジェクトが1つ存在する場合は、このオブジェクトの最初の数行が表示されます。
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以外のライブラリに存在する場合は、オブジェクトが存在するライブラリの ID を指定する必要があります。 *library-id* の設定値を "SYS"（SYSTEM を除く）で始めることはできません。 Natural Security がアクティブの場合は、ライブラリ ID の入力は許可されません。

EDT のサブコマンド

行の編集では、次のサブコマンドを使用できます。

コマンド	機能
.B	一番下に移動します。
.C*nnnn*(*m*)	*nnnn* で特定された行をコピーします。 *m* は、コピーされる行数を示します。
.C'*text*'(*m*)	*text* で始まる行をコピーします。 *m* は、コピーされる行数を示します。
.D	行を削除します。
.D(*n*)	*n* 行分を削除します。

コマンド	機能
.E	ラインエディタを終了します。
.I	行を挿入します。
.I(*program*)	*program* を挿入します。
.M*nnnn*	*n* で特定された行を移動します。
.M'*text*'(*m*)	*text* で始まる行を移動します。 *m* は、移動する行数を示します。
.R	番号を再設定します。
.S'*text*'	*text* をスキャンします。
.T	一番上に移動します。
.*nnnn*	*nnnn* 行目に移動します。
.+*n*	*n* 行分前に移動します。
.-*n*	*n* 行分後ろに移動します。

EDT のファンクションキー

行の編集では、次の PF キーを使用できます。

キー	コマンド	機能
PF1	.-18	18 行分後ろにスクロールします。
PF2	.T	一番上にスクロールします。
PF3	.B	一番下にスクロールします。
PF4	.+5	5 行分前に移動します。
PF5	.+10	10 行分前に移動します。
PF6	.+18	18 行分前に移動します。
PF7	.R	番号を再設定します。
PF8	.I	行を挿入します。
PF9	.E	ラインエディタを終了します。
PF10	.E,RUN	ラインエディタを終了してプログラムを実行します。
PF11	.E,SAVE,RUN	ラインエディタを終了し、保存してプログラムを実行します。
PF12	.E,CAT,SAVE,EX	ラインエディタを終了し、カタログおよび保存してプログラムを実行します。

18 EXECUTE

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

このコマンドは、プログラムタイプの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。

構文説明

EXECUTE	キーワード EXECUTE はオプションであり、*program-name*（実行するプログラムの名前）だけで指定できます。 **注意:** プログラムエディタのコマンド行に入力するときは、システムコマンド EXECUTE を EX に*省略しないでください*。プログラムエディタでは、プログラムエディタコマンドの EX として解釈されるためです。
REPEAT	実行されるプログラム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。
program-name	実行するプログラムの名前。 *library-id* を指定しない場合、Natural では、指定されたプログラムが現在のライブラリまたは現在の steplib 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 steplib は SYSTEM です。
library-id	プログラムが別のライブラリにある場合は、そのライブラリの *library-id* 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 *library-id* は指定できません。
parameter	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。 **注意:** パラメータ値は、端末コマンド %L またはプロファイルパラメータ LC=ON に関係なく、常に大文字に変換されます。

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
```

19 FIN

FIN

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。

20 GLOBALS

```
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。

構文説明

GLOBALS	GLOBALS コマンドをパラメータなしで入力すると、画面が表示され、ここでパラメータ設定を変更できます。
parameter	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。 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。 次に例を示します。 `GLOBALS DC=, ID=.`

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
CC	バッチモードでのエラー処理
CF	端末コマンドの文字
CPCVERR	コードページ変換エラー
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
DO	出力データの表示順序
DU	ダンプ生成
EJ	ページ換え
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
IA	INPUT 割り当て文字
ID	INPUT 区切り文字
IM	入力モード

パラメータ	機能
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
MT	最大 CPU 時間
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
PD	NATPAGE のページ制限
PM	出力モード
PS	Natural レポートのページサイズ
REINP	不正データに対する内部 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
SL	ソース行の長さ
SM	ストラクチャードモードでのプログラミング
THSEPCH	千桁単位セパレータ文字
TS	システムライブラリでプログラムからの出力を変換
WH	ホールド状態でのレコードの待機
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、変更に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。

21 HELP

このコマンドは、Natural のヘルプユーティリティを呼び出すために使用します。 Natural のステートメントやコマンド、エラーメッセージなどに関する情報を検索できます。

HELP	ヘルプメニューを表示します。
HELP *statement*	指定された *statement* のヘルプ情報を表示します。
HELP *command*	指定された *command* のヘルプ情報を表示します。
HELP [NAT]*nnnn*	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 NAT*nnnn* の長文テキストが表示されます。
HELP NAT	すべてのエラーメッセージに関する情報を表示します。
HELP USER*nnnn*	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。
HELP USER*nnnn* **[***library-name***]**	指定された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。
HELP USER	現在のライブラリのライブラリ固有のすべてのメッセージの選択リストを表示します。
HELP USER [*library-name***]**	指定されたライブラリのライブラリ固有のすべてのメッセージの選択リストを表示します。

HELP ERROR	最後に発生したエラーの長文テキストを表示します。

22 INPL

INPL [R]

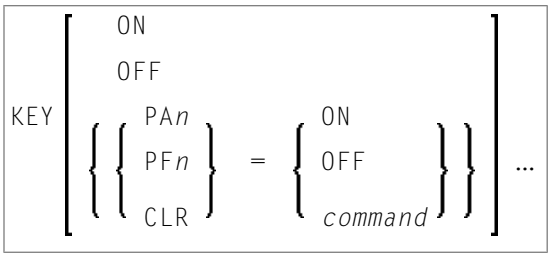
このコマンドは、Natural の INPL ユーティリティを呼び出すために使用します。 このユーティリティは、Software AGインストールデータセットをシステムファイルにロードするためにのみ使用します。詳細については、INPL オンラインヘルプおよびプラットフォーム固有のインストールドキュメントを参照してください。

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。

INPL	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。
INPL R	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、『*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。

23 KEY



このコマンドは、ビデオ端末のキーボードのキーに機能を割り当てるために使用します。 割り当てた機能を変更、アクティブ化、非アクティブ化することもできます。

対象となるキーは次のとおりです。

- PA1～PA3
- PF1～PF24
- CLEAR

上記の各キーに対し、次の機能のいずれかを割り当てることができます。

- Natural のシステムコマンド
- Natural の端末コマンド
- ユーザー定義コマンド

コマンドモード（NEXT プロンプト）で対応するキーを押すたびに、割り当てられたコマンドが実行されます。

注意:

1. システムコマンド KEY を使用した割り当ては、ほとんどの場合、プログラムの SET KEY ステートメントによる割り当てとは独立しています。
2. ファンクションキー割り当ては Natural 管理者がプロファイルパラメータ KEY を使用して行うこともできます。
3. このコマンドは、バッチモードでは実行できません。

コマンドの割り当て

パラメータなしで KEY コマンドのみを入力すると、［**Function-Key Assignments**］画面が表示されます。 この画面では、入力フィールドにコマンド名を入力することによって、個別キーにコマンドを割り当てることができます。

キーに別のコマンドを割り当てるには、入力フィールドの既存エントリを上書きします。

コマンドの割り当てを削除するには、入力フィールドのエントリを削除するか、または空白で上書きします。

KEY コマンドでキーを直接指定することによって、個別キーにコマンドを割り当てることもできます。 次に例を示します。

```
KEY PF1=CLEAR
```

割り当てるコマンドに空白が含まれる場合は、コマンドをアポストロフィで囲む必要があります。 次に例を示します。

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

すべてのキーのアクティブ化／非アクティブ化 - KEY ON/OFF

コマンド KEY OFF/ON を使用すると、ファンクションキーのすべての割り当てを非アクティブにしたり、再びアクティブにしたりすることができます。

KEY OFF	ファンクションキーを押すと、そのキーがアクティブではないことを示すメッセージが返されます。
KEY ON	KEY OFF によって以前に非アクティブ化されたすべてのファンクションキーの割り当てを再びアクティブにします。

キーのアクティブ化／非アクティブ化は、［**Function-Key Assignments**］画面の右上隅にある［**Activate Keys**］フィールドの ON/OFF のエントリを上書きすることによっても可能です。

注意: CLEAR キーはアクティブ化／非アクティブ化できません。 他の機能が割り当てられていない限り、このキーの機能は端末コマンド %% と同じです。 コマンド KEY ON/OFF および［**Activate Keys**］フィールドは、CLEAR キーに影響を与えません。

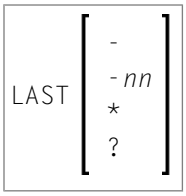
個別キーのアクティブ化／非アクティブ化 - KEY key=ON/OFF

コマンド KEY *key*=OFF/ON を使用すると、特定の *key* に割り当てられたコマンドを非アクティブにしたり、再びアクティブにしたりすることができます。

KEY *key*=**OFF**	特定の*key*に割り当てられたコマンドを非アクティブにします。 次に例を示します。 `KEY PF24=OFF`
KEY *key*=**ON**	以前に非アクティブ化されたコマンド割り当てを再びアクティブにします。 次に例を示します。 `KEY PF24=ON`

注意: コマンド `KEY CLR=ON/OFF` は使用できません。上記の注も参照してください。

24 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、コマンド行または［NEXT］行に配置され、実行可能になります。
LAST -	最後に発行されたコマンドが、コマンド行または［NEXT］行に配置され、実行可能になります。 再びLAST -を入力すると、最後の1つのコマンドがコマンド行または［NEXT］行に配置されます。 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。 **注意:** 手動で繰り返し入力する代わりに、システムコマンドKEYによってLAST -をPFキーに割り当てることもできます。
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、コマンド行または［NEXT］行に配置され、実行可能になります。
LAST *	ウィンドウが表示され、最後に発行された20個のコマンドが示されます。10個以上のコマンドが表示される場合には、PF8とPF7を使用して上下にスクロールします。 ■ *単一*のコマンドを再実行するには、カーソルでコマンドをマークしてF5キーを押すか、または任意の文字でコマンドをマークしてEnterキーを押します。

	■ *複数*のコマンドを再実行するには、実行する順に番号でコマンドをマークしてEnterキーを押します。番号の昇順でコマンドが実行されます。
LAST ?	LAST コマンドのヘルプ機能を呼び出します。

25 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。

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。

手順 25.1. 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 つをカーソルでマークし、Enter キーを押します。

  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

注意: 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。

26 LIST

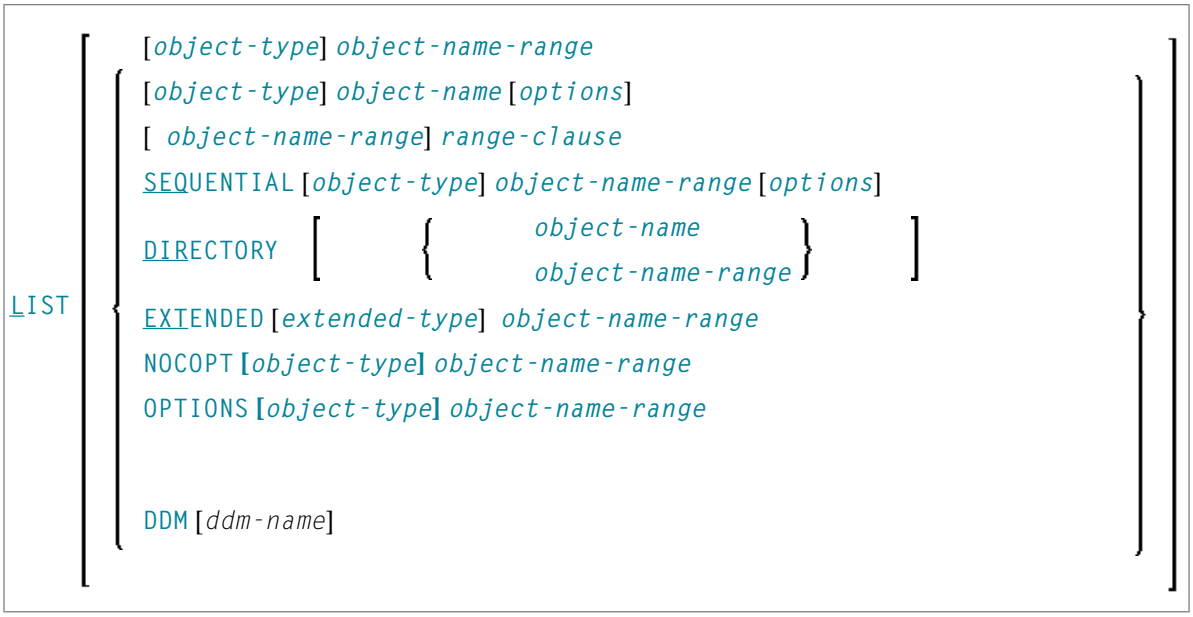
このシステムコマンドは、1つのオブジェクトのソースコードを表示したり、現在のライブラリに含まれている 1 つまたは複数のオブジェクトをリストしたりするために使用します。 LIST コマンドのオプションについては以下で説明します。

この章では、次のトピックについて説明します。

LIST XREF、LIST COUNT、および LISTSQL について説明している他のドキュメントも参照してください。

アプリケーションプログラムインターフェイス: USR1054N、USR1055N、USR1056N、USR2018N。『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。

構文の概要

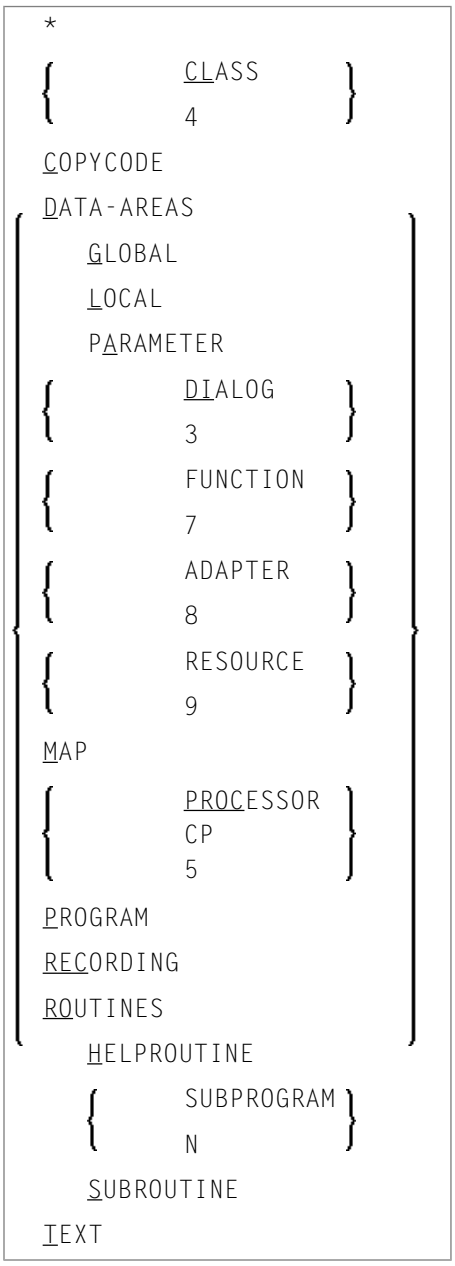


注意:

1. キーワード DDM の代わりに、キーワード VIEW（短縮形 V）を使用することもできます。
2. LIST では最大 244 文字を含む長い行を表示できるため、プロファイルパラメータ LS を使用して、できる限り大きな行サイズを設定する必要があります。可能な場合は、LS=250 に設定してください。

object-type

object-type には、下に示すいずれかのオブジェクトタイプまたはアスタリスク（*）を指定できます。



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（最大長は8文字、例外：LIST EXTENDEDでは 32 文字）。

object-name-range

*object-name-range*の代わりに、アスタリスク（*）とワイルドカード（?）を指定することもできます。

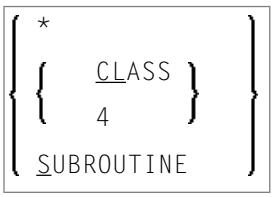
- 現在のライブラリにあるすべてのオブジェクトをリストするには、*object-type* ではなく、*object-name-range* にアスタリスク（*）を指定します。
- 特定のタイプのオブジェクトをすべてリストするには、*object-name-range*に特定の *object-type* とアスタリスク（*）を指定します。
- 特定の範囲にあるオブジェクトをリストするには、*object-name-range* にアスタリスク表記とワイルドカード表記を使用します。
  - アスタリスク表記は、*object-name-range*にアスタリスク（*）を指定するオプションです。アスタリスクは、任意の長さの文字列を表します。
  - ワイルドカード表記は、*object-name-range* に疑問符（?）を指定するオプションです。疑問符は、任意の 1 文字を表します。
- *object-name-range*では、1つまたは複数のアスタリスク表記とワイルドカード表記を組み合わせることができます。
- 特定の開始値から、または特定の終了値まで、すべてのオブジェクトをリストするには、それぞれ > 表記または < 表記を使用します。
- 表記<と>を互いに組み合わせること、またはアスタリスクやワイルドカード表記と組み合わせることはできません。これらの表記は、オブジェクトのリストを表示するためにのみ使用されます（下記の「*オブジェクトのリスト*」を参照）。

options

options の詳細については、「*オプション*　」を参照してください。

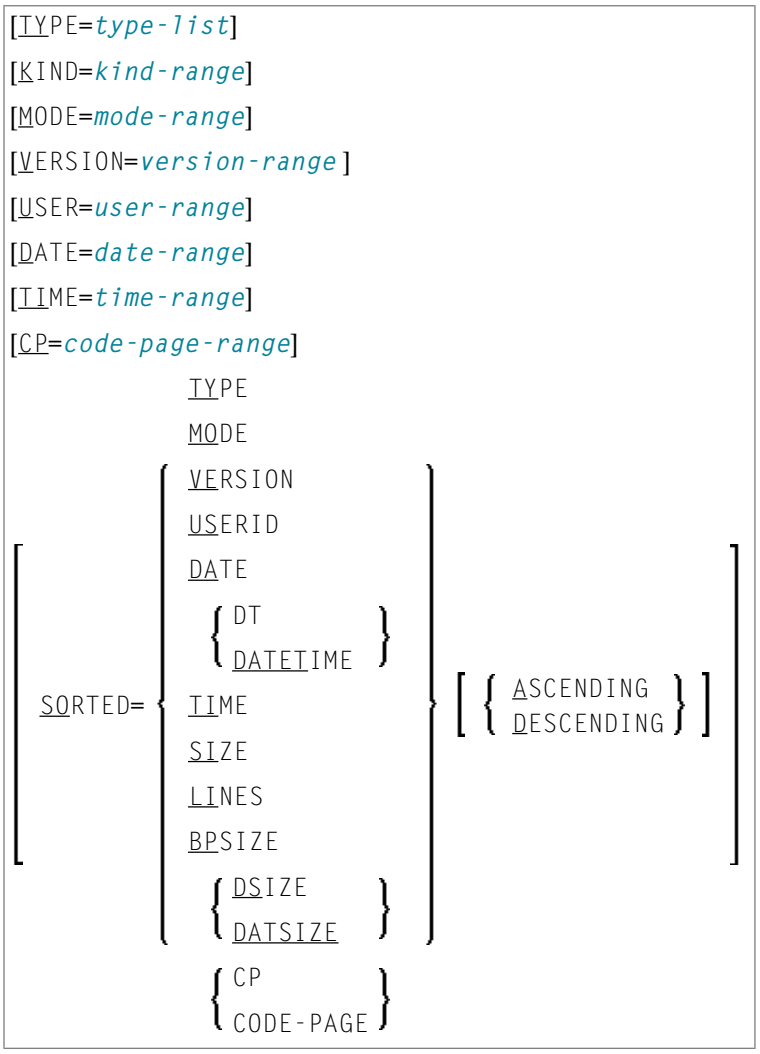
extended-type

extended-type には、下に示すいずれかのオブジェクトタイプまたはアスタリスク（*）を指定できます。



詳細については、下記の「LIST EXTENDED」を参照してください。

range-clause



type-list	*（すべてのタイプ）、または最大 11 個の有効な Natural オブジェクトタイプの 1 バイト文字（例：プログラムは P、マップは M）	
kind-range	*	すべてのオブジェクトをリストします。
	S	ソースオブジェクトのみをリストします。
	C	カタログ化オブジェクトのみをリストします。
	S/C	ソースおよびカタログとして存在しているオブジェクトのみをリストします。
	S/	ソースおよびカタログとして存在しているオブジェクトのみをリストします。
	/C	カタログのみとして存在しているオブジェクトのみをリストします。
	W	格納されたオブジェクトのみをリストします。
mode-range	*	すべてのオブジェクトをリストします。
	S	ストラクチャードモードのオブジェクトのみをリストします。
	R	レポートモードのオブジェクトのみをリストします。
version-range	Natural オブジェクトの Natural バージョン。 「用語集」でバージョンの定義も参照してください。 有効なバージョンフォーマット：*V.R.SM*（ここで、*V* は 1 桁のバージョン、*R* は 1 桁のリリース、*SM* は 2 桁のシステムメンテナンスレベルです） バージョンの範囲を指定することもできます（下記の「範囲表記」を参照）。	
user-range	Natural プログラミングオブジェクトを保存またはカタログしたユーザーの ID。 単一のユーザー ID またはユーザー ID の範囲を指定します（下記の「*range-notation*」を参照）。	
date-range	保存またはカタログ日付が指定された日付の範囲内にあるオブジェクトをすべて選択します。 単一の日付または日付の範囲を指定します。 有効な日付フォーマット:*YYYY-MM-DD* 有効な日付の範囲： ■ 先頭文字（例：2002*） ■ 開始値（例：2002-05>） ■ 終了値（例：2003-02<） 日付には、次の特殊な入力値を使用できます。	
	TODAY（+/-*nnnn*）	当日の日付が含まれるすべての項目。 この後には、+*nnnn* または –*nnnn* を付け加えることができます。*nnnn* は最大 4 桁の数値です。

		これを付け加えた場合、現在の日付に *nnnn* 日分を加算または減算した日付が対象として算出されます。 開始値オプション（>）または終了値オプション（<）と組み合わせることができます。例えば、T0-1> では、最近2日以内に保存またはカタログされたすべてのオブジェクトが選択されます。
	YESTERDAY	当日の1日前の日付が含まれるすべての項目。
	MONTH	当月内の日付の範囲が含まれるすべての項目。
	YEAR	当年内の日付の範囲が含まれるすべての項目。
time-range	保存またはカタログ日付が指定された時刻の範囲内にあるオブジェクトをすべて選択します。 単一の時刻または時刻の範囲を指定します。 有効な時刻フォーマット：*HH:II:SS*（*HH* = 時、*II* = 分、*SS* = 秒）。 有効な時刻の範囲： ■ 先頭文字（例：10:*） ■ 開始値（例：10:30>） ■ 終了値（例：11:20<）	
code-page-range	単一のコードページまたはコードページの範囲を指定します（下記の「*range-notation*」を参照）。	

range-notation

- 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、アスタリスク（*）を使用します。
- 特定の範囲にあるオブジェクトをリストするには、アスタリスク表記とワイルドカード表記を使用します。
  - アスタリスク表記は、アスタリスク（*）を指定するオプションです。アスタリスクは、任意の長さの文字列を表します。
  - ワイルドカード表記は、疑問符（?）を指定するオプションです。疑問符は、任意の1文字を表します。
- 1つまたは複数のアスタリスク表記とワイルドカード表記を組み合わせることができます。
- 特定の開始値から、または特定の終了値まで、すべてのオブジェクトをリストするには、それぞれ > 表記または < 表記を使用します。
- 表記<と>を互いに組み合わせること、またはアスタリスクやワイルドカード表記と組み合わせることはできません。

ワークエリアの内容のリスト

LIST	パラメータなしで LIST コマンドのみを入力すると、ワークエリアの内容がリストされます。

個々のソースコードの表示

LIST *object-name* [*options*]	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。 LIST コマンドで 1 つのオブジェクト名を入力する場合は、*object-type* を指定する必要はありません。 *object-type* を指定する場合は、*object-name* も指定する必要があります。
LIST *object-type object-name* [*options*]	

ソースの連続表示

LIST SEQUENTIAL *object-name-range* [*options*]	いずれの場合も、*object-name-range* にアスタリスク（*）とワイルドカード（?）表記のいずれかまたは両方を使用する必要があります。 選択条件に一致するすべてのオブジェクトのソースが、連続的に表示されます。
LIST SEQUENTIAL *object-type object-name-range* [*options*]	

オブジェクトのリストの表示

LIST *object-name-range*	どちらの場合も、*object-name-range* にアスタリスク（*）とワイルドカード（?）のいずれかまたは両方を使用する必要があります。 指定した選択条件に一致するすべてのオブジェクトのリストが表示されます。 リストから表示するオブジェクトを選択するには、ファンクションコード LI を使用します（「*オブジェクトへの機能の実行*」を参照）。
LIST *object-type object-name-range*	

事前選択したオブジェクトのソート済みリストの表示

LIST *object-name-range* **LIST** *object-name-range* *range-clause*	どちらの場合も、*object-name-range* にアスタリスク（*）とワイルドカード（?）のいずれかまたは両方を使用する必要があります。 指定した選択条件に一致するすべてのオブジェクトのリストが表示されます。 リストから表示するオブジェクトを選択するには、ファンクションコード LI を使用します（「*オブジェクトへの機能の実行*」を参照）。 *range-clause* では、追加の選択条件とソート条件を指定します。 下記の例も参照してください。

カタログされたサブルーチンとクラスのロングネームの表示

LIST EXTENDED *object-name-range* **LIST EXTENDED** *extended-type* *object-name-range*	カタログされたサブルーチンとクラスのロングネームのリストを表示します。 名前のオプションについては、上記の「*object-name-range*」を参照してください。

カタログ化オブジェクトの NOC オプションの表示

LIST NOCOPT [*object-type*] *object-name-range*	Natural Optimizer Compiler（NOC）を使用してコンパイルされたカタログ化オブジェクトと CATALOG の実行中に使用された初期 NOC オプションのリストを表示します。 名前のオプションについては、上記の「*object-name-range*」を参照してください。

カタログ化オブジェクトのコンパイラオプションの表示

LIST OPTIONS [*object-type*] *object-name-range*	カタログ化オブジェクトと CATALOG の実行中に使用されたコンパイラオプションのリストを表示します。 名前のオプションについては、上記の「*object-name-range*」を参照してください。 デフォルトでは、最終的なコンパイラオプション（CATALOG の実行の最後でアクティブだったオプション設定）が表示されます。 Natural バージョン 4.2.5 以降でカタログされたオブジェクトについては、初期コンパイラオプション（CATALOG の実行が開始された時点でアクティブだったオプション設定）または変更されたコンパイラオプション（ソースコード内で変更されたオプション設定）を表示で

	きます。 オンラインマップの範囲フィールドについては、対応するヘルプマップを参照してください。

ディレクトリ情報の表示

LIST DIRECTORY	現在ワークエリアにあるオブジェクトのディレクトリ情報を表示します。 ■ 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（レポーティングまたはストラクチャード）、TP システム、端末 ID、オペレーティングシステム、トランザクション、Natural バージョン、コードページ情報（使用可能な場合）、ソースサイズ ■ 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、TP モニタシステム、端末 I/O、トランザクション、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／バージョン、使用されている GDA、グローバルデータのサイズ、DATSIZE でのサイズ、バッファプールでのサイズ、OPT コードのサイズ（Natural Optimizer Compiler によって生成されたマシンコードのサイズ）、初期の OPT 文字列（STOW 時刻に有効な OPT プロファイルパラメータの値）、コンパイラオプション
LIST DIRECTORY *object-name*	指定されたオブジェクトのディレクトリ情報（上記の LIST DIRECTORY の説明を参照）を表示します。 *object-name-range* の代わりにアスタリスク（*）とワイルドカード（?）表記のいずれかまたは両方を使用すると、対応する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。
LIST **DIRECTORY** *object-name-range*	
LIST *object-name* **WITH DIRECTORY**	このコマンドは、最初に指定されたオブジェクトの情報（上記の LIST DIRECTORY の説明を参照）を表示し、その後、オブジェクトのソースコードをリストします。

DDM（ビュー）の表示

LIST DDM	すべての DDM のリストを表示します。
LIST DDM *ddm-name*	1 つの DDM 名を指定すると、指定した DDM が表示されます。 *ddm-name* には、単一の DDM 名（最大 32 文字まで）を指定するか、または特定の範囲にある DDM のリストを表示する場合は *object-name-range* のように範囲を指定します。

注意: キーワード DDM の代わりに、キーワード VIEW（短縮形 V）を使用することもできます。

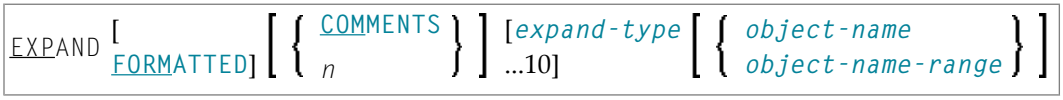
オプション

options の代わりに、次のオプションのいずれかを指定することもできます。

```
{ [[WITH] DIRECTORY] [NUMBERS OFF] [expand-option]
  formatted-option
  CONVERTED                                        }
```

DIRECTORY	このコマンドでは、最初に指定されたオブジェクトの情報（下記の「*ディレクトリ情報の表示*」を参照）を表示し、その後、オブジェクトのソースコードをリストします。
NUMBERS OFF	デフォルトでは、オブジェクトのソースコードが、ソースコードでの行番号を付けてリストされます。 ソースコードを行番号なしでリストするには、NUMBERS OFF オプションを指定します。 「*リストされたソースに使用するサブコマンド*」セクションでサブコマンド NUMBERS ON および NUMBERS OFF も参照してください。
CONVERTED	デフォルトでは、システムファイルとして保存されているコードページにあるソースがリストされます。 デフォルトコードページ（システム変数 *CODEPAGE を参照）にあるソースをリストするには、このオプションを指定します。

expand-option



EXPAND *object-name* **EXPAND** *object-name-range*	EXPANDオプションを使用すると、リストされたオブジェクトによって参照された他のオブジェクト（コピーコード、データエリア、マップ、ヘルプルーチン、外部サブルーチン、サブプログラム、FETCH ステートメントで呼び出されるプログラム、エラーメッセージ）のソースを、リストされたオブジェクトのソースの*中*に表示できます。 このオプションは、特にバッチモードで役立ちます。 例えば、リストされたソースプログラムに INCLUDE ステートメントが含まれている場合、インクルードされているコピーコードのソースコードを、INCLUDE ステートメントの直後にリストされるソースプログラムの中に表示できます。 これ以降の説明では、ソースの中にリストされるオブジェクトは 「拡張オブジェクト」 と記載されています。 拡張オブジェクト内のサブコマンド リストされている拡張オブジェクトの中では、次のサブコマンドのみが使用可能です。 PRINT + - - . 「*オブジェクトのリストの使用例*」を参照してください。
EXPAND FORMATTED	EXPAND FORMATTED オプションは、格納されたデータエリア（ソースオブジェクトとカタログ化オブジェクトのタイムスタンプが同じ）およびソース内にリストされるマップにのみ関係しています データエリアについては、次のことが適用されます。 ■ FORMATTED を指定しない場合、データエリアの表示は、データエリアエディタと類似した表示になります。 ■ FORMATTED を指定した場合、データエリアの表示は、DEFINE DATA ステートメントと類似した表示になります。これは、格納されたデータエリア（つまり、ソースオブジェクトとカタログ化オブジェクトのタイムスタンプが同じ）にのみ適用されます。「サブコマンド FORMAT」も参照してください。 マップについては、次のことが適用されます。 ■ FORMATTED を指定しない場合は、マップソースがリストされます。 ■ FORMATTED を指定した場合は、マップ*レイアウト*が表示されます（つまり、ランタイム時にユーザーに対して表示されるマップと同じ）。

<table>
<tr><td>EXPAND COMMENTS

EXPAND n</td><td colspan="3">オプション EXPAND COMMENTS を使用すると、拡張オブジェクトの最初のコメント行のみがリストされます。つまり、拡張オブジェクトは、コメント行ではない最初のソースコード行の直前までがリストされます。

オプション EXPAND n を使用すると、拡張オブジェクトの先頭の n 行のみがリストされます。

これらのオプションのいずれも使用しない場合は、拡張オブジェクトの全体がリストされます。</td></tr>
<tr><td>expand-type</td><td colspan="3">expand-typeには、拡張オブジェクトのオブジェクトタイプを指定します。次の expand-types を指定できます。

<table>
<tr><td>P</td><td>プログラム</td><td rowspan="11">複数の expand-typeを指定する必要がある場合は、これらを間に空白を入れずに任意の順序で指定できます。例えば、リストされたソースの中にマップ、コピーコード、およびサブルーチンを表示するには、expand-typeに MCS を指定します。</td></tr>
<tr><td>N</td><td>サブプログラム</td></tr>
<tr><td>S</td><td>外部サブルーチン</td></tr>
<tr><td>H</td><td>ヘルプルーチン</td></tr>
<tr><td>G</td><td>グローバルデータエリア</td></tr>
<tr><td>L</td><td>ローカルデータエリア</td></tr>
<tr><td>A</td><td>パラメータデータエリア</td></tr>
<tr><td>M</td><td>マップ</td></tr>
<tr><td>C</td><td>コピーコード</td></tr>
<tr><td>E</td><td>エラーメッセージ</td></tr>
<tr><td>4</td><td>クラス</td></tr>
<tr><td>*</td><td>すべてのオブジェクトタイプ</td></tr>
</table>
</td></tr>
<tr><td>object-name</td><td colspan="3">object-nameまたは object-name-rangeには、リストされたメインのソースの中で表示する拡張オブジェクトの名前を指定します。</td></tr>
<tr><td>object-name-range</td><td colspan="3">object-nameまたは object-name-rangeと同じ表記を使用できます（< と > を除く）。</td></tr>
</table>

formatted-option

FORMATTED オプション

FORMATTEDオプションは、格納（STOW）されたデータエリア（つまり、ソースオブジェクトとカタログ化オブジェクトのタイムスタンプが同じ）およびマップに適用されます。

FORMATTED	格納（**STOW**）されたデータエリア： データエリアに対して FORMATTED オプションを指定すると、データエリアはフォーマットされて表示されます。つまり、表示は、DEFINE DATA ステートメントと類似した表示になります。「サブコマンド FORMAT」も参照してください。 これは、格納されたデータエリア（つまり、ソースオブジェクトとカタログ化オブジェクトのタイムスタンプが同じ）にのみ適用されます。デフォルトでは、データエリアは未フォーマットの状態で表示されます。つまり、表示は、データエリアエディタと類似した表示になります。 デフォルト設定は、LIST プロファイルで変更できます。 下記の「*個別リストプロファイルの定義*」を参照してください。また、「サブコマンド FORMAT」も参照してください。 マップ： マップに対して FORMATTED オプションを指定すると、マップ*レイアウト*が表示されます。つまり、ランタイム時にユーザーに対して表示されるマップと同じになります。

マップをリストするための **FORMATTED** オプション

マップをリストしている場合、キーワード FORMATTED の他にオプションを指定できます。

['*c*'] ['*c*']	充填文字の使用 入力フィールド（AD=A と AD=M）および出力フィールド（AD=O）に対して充填文字 *c* を指定して、これらのフィールドを表示することもできます。 充填文字として任意の文字を指定できます。 次の例では、すべての入力フィールドがアンダースコア（_）付きで、およびすべての出力フィールドがハッシュ記号（#）付きで表示されます。 `LIST MAP map-name FORMATTED '_' '#'`

SETTINGS	マップ設定： 指定したマップのマップ設定を表示します。 `LIST MAP map-name FORMATTED SETTINGS`
FIELDS	フィールドサマリ： フィールドサマリ、つまり指定したマップ内にあるフィールドのリストを表示します。 `LIST MAP map-name FORMATTED FIELDS`
EXTFIELDS	拡張フィールド編集情報： すべてのマップフィールドの拡張フィールド編集情報が表示されます。 `LIST MAP map-name FORMATTED EXTFIELDS`

マップの処理ルールの表示

以下のオプションを指定すると、マップによって使用される処理ルールが表示されます。 処理ルールは、それをフィールドに割り当てた順序、およびフィールドごとのランクの順序で表示されます。

RULES	すべてのルール： `LIST MAP map-name FORMATTED RULES` 指定したマップのすべてのルールが表示されます。
INLINERULES	インラインルールのみ： `LIST MAP map-name FORMATTED INLINERULES` 指定したマップのインラインルールのみ表示されます。
FREERULES	フリールールのみ： `LIST MAP map-name FORMATTED FREERULES` 指定したマップのフリールールのみ表示されます。
AUTORULES	自動ルールのみ： `LIST MAP map-name FORMATTED AUTORULES` 指定したマップの自動ルールのみ表示されます。

「*ソースのリスト*」セクションでサブコマンド `LAYOUT` と `FORMAT` も参照してください。

オブジェクトのリスト

オブジェクト名でアスタリスクまたはワイルドカード表記を使用すると、指定した選択条件に一致するすべてのオブジェクトのリストが表示されます。 リストから表示や印刷などを行うオブジェクトを選択するには、ファンクションコードでそれらのオブジェクトをマークするか、Natural システムコマンドまたは LIST サブコマンドをコマンド行に入力します。

このセクションでは、表示されているオブジェクトのリスト（例えば、LIST * コマンドを発行した後に表示される）で使用可能な機能、サブコマンド、およびファンクションコードについて説明します。 次のトピックについて説明します。

- 列ヘッダーの説明
- オブジェクトの選択リストのスクロール
- 選択リストの条件変更
- 選択リストに表示される情報
- 選択リストでの項目の強調表示
- 選択リストのサブコマンド
- オブジェクトへの機能の実行
- オブジェクトのリストのソート
- オブジェクトのリストの使用例

列ヘッダーの説明

オブジェクトのリストには、次の列があります。

列	説明
Cmd	この列には、選択リスト内のオブジェクトに対して実行する機能のコードを入力できます。「*オブジェクトへの機能の実行*」を参照してください。
Name	オブジェクトの名前。
Type	オブジェクトのタイプ。
S/C	オブジェクトが、ソースとカタログのいずれかまたは両方として存在しているかどうかを示します。
SM	オブジェクトを作成するときに使用された Natural プログラミングモード。 S = ストラクチャードモード、R = レポーティングモード。
Version	オブジェクトを作成またはカタログするときに使用された Natural の製品バージョン。
User ID	オブジェクトを作成またはカタログしたユーザーのユーザー ID。
Date, Time	オブジェクトが作成またはカタログされた日付と時刻。

オブジェクトの選択リストのスクロール

表示されたオブジェクトのリストでは、次の方法でリストをスクロールできます。

- リストを 1 ページ分進む方向または戻る方向にスクロールするには、それぞれ PF8 キーまたは PF7 キーを押します。
- リストを先頭または最後までスクロールするには、それぞれ PF6 キーまたは PF9 キーを押します。

選択リストの条件変更

オブジェクトのリストを表示すると、列ヘッダーの直下にあるフィールドに、現在の選択リストの選択条件が示されます。 選択条件は、これらのフィールドの値を上書きすることによって変更できます。各フィールドで設定可能な値に関する情報は、そのフィールドに疑問符（?）を入力すると表示されます。

選択リストに表示される情報

オブジェクトに対してソースモジュールとオブジェクトモジュールの両方が存在している場合（［**S/C**］列で示されている場合）、ソースの情報は表示されますが、オブジェクトモジュールの情報は表示されません。

注意: ソート機能がアクティブになっていると、ソースとオブジェクトモジュールは別々に表示されます。例えば、リストがオブジェクトの日付でソートされ、さらにソースとオブジェクトモジュールの日付値が異なっている場合は、これに該当します。

手順 26.1. ソースオブジェクトとカタログ化オブジェクトの詳細を表示するには

- 右へシフトするには、PF11 キーを押します。

  または:

  左へシフトするには、PF10 キーを押します。

注意: デフォルトでは、パフォーマンス上の理由から、ソースオブジェクトのソース行の行数は計算されません。 ソース行の行数を表示する必要がある場合は、サブコマンド `COUNTSOURCE ON` を入力するか、または `LIST` プロファイル（下記の「*個別リストプロファイルの定義*」を参照）でパラメータ `COUNT-SOURCE-LINES` を `Y` に設定できます。

選択リストでの項目の強調表示

リストページの左端にある項目が強調表示されている場合、これはオブジェクトのソースとそのオブジェクトモジュールに間に不整合があることを示しています。 不整合の詳細については、オブジェクトをファンクションコード LD でマークして、オブジェクトのディレクトリ情報をリストしてください。 不整合を解消するには、通常、オブジェクトを再度格納します（ファンクションコード ST）。

選択リストのサブコマンド

オブジェクトのリストには、コマンド行に Natural システムコマンドまたは LIST サブコマンドを入力できます。 次のサブコマンドが有効です。

コード	機能	
CODE-PAGE または CP	ON	各オブジェクトのコードページ情報を表示します。これはデフォルト値です。
	OFF	コードページ情報を表示しません。
SC	スキャン対象値を含むオブジェクトのみをリストします（長いリストでのみ使用可能）。	
SC OFF	スキャンモードをオフに切り替えます。	
SHORT	オブジェクトの短いリストを表示します。つまり、オブジェクト名のみを表示します（スキャンモードがオフの場合にのみ使用可能）。	
LONG	すべてのフィールドが使用可能な長いリストに切り替えます。	
PRINT	オブジェクトのリストを印刷します。	
EXTENDED	サブルーチンまたはクラスのロングネームを表示します。LIST EXTENDED * と同じです。	
ALL *fx*	すべての表示されているオブジェクトに、ファンクションコード *fx* を入力します（*fx* は、リストされているオブジェクトに有効なファンクションコード）。	
SORT	ソートウィンドウを呼び出します（下記の「*オブジェクトのリストのソート*」を参照）。	
COUNTSOURCE	ON	ソースオブジェクトのソース行の行数を表示します。
	OFF	ソースオブジェクトのソース行の行数を表示しません。
MARK-LONG-LINES	ON	ソースオブジェクトのリストにある長い行の先頭の 2 つの位置に L を挿入して、その行をマークします。 LIST プロファイルでデフォルト値を指定できます。「*個別リストプロファイルの定義*」を参照してください。
	OFF	ソースオブジェクトのリストにある長い行をマークしません。

コード	機能	
DEFINE-DATA	ON	リストされているデータエリアソースを、デフォルトの DEFINE DATA フォーマットで表示します（LIST *dataarea* FORMATTED と同じ）。 LISTプロファイルでデフォルト値を指定できます。「*個別リストプロファイルの定義*」を参照してください。
	OFF	リストされているデータエリアソースを、未フォーマットの状態で表示します。
LISTPROFILE	LIST プロファイルのパラメータの現在値を表示します（下記の「*個別リストプロファイルの定義*」を参照）。	
NOCOPT	Natural Optimizer Compiler（NOC）を使用してコンパイルされたカタログ化オブジェクトと CATALOG の実行で使用された初期 NOC オプションのリストを表示します。LIST NOCOPT * と同じです。「*カタログ化オブジェクトの NOC オプションの表示*」を参照してください。	
OPTIONS	カタログ化オブジェクトとカタログ化に使用された初期コンパイルオプションのリストを表示します。LIST OPTIONS * と同じです。「*カタログ化オブジェクトのコンパイラオプションの表示*」を参照してください。	
REUSE	ON	再使用モードをオンに切り替えます。 ［**Cmd**］列に入力したコマンドを実行した後に、最後に表示したリストを再度使用します。ただし、次のコマンドは除きます。 E ED（編集） CA（カタログ） UC（UNCAT） S ST（格納） D DE（削除） RE（名前の変更）
	OFF	再使用モードをオフに切り替えます。 ［**Cmd**］列に入力したコマンドの実行後に、リストを再構築します。
REFRESH	現在表示されているリストを再構築します。 このサブコマンドは、特に再使用モードがオンになっている場合に使用できます。	
+	1 ページ分進む方向にスクロールします。	
-	1 ページ分戻る方向にスクロールします。	
++	オブジェクトリストの最後（ボトム）までスクロールします。	
--	オブジェクトリストの先頭（トップ）までスクロールします。	
?	コマンド行のヘルプ。	

オブジェクトへの機能の実行

選択リストのオブジェクトに対して機能を実行するには、左側の列（タイトルが［**Cmd**］）で適切なファンクションコードを指定してオブジェクトをマークします。

選択リストにある複数のオブジェクトを、異なるファンクションコードでマークすることもできます。この場合、機能は 1 つずつ順番に実行されます。

次のファンクションコードが使用可能です（可能な省略形には下線を表示）。

コード	機能
?	マークしたオブジェクトに使用可能なすべての機能を示すウィンドウが表示されます。このウィンドウには、選択されたオブジェクトに実際に使用可能な機能のみがリストされます。例えば、オブジェクトがサブルーチンの場合は機能を実行できませんが、オブジェクトがソース形式だけで使用可能な場合は機能を実行できます。 ウィンドウからは、マークしたオブジェクトに対して実行される機能を選択できます。
CA	オブジェクトをコンパイルして、オブジェクト形式で保存します（システムコマンド CATALOG と等価）。
DE	オブジェクトを削除します（システムコマンド DELETE と等価）。
DL	オブジェクトをメインフレームからパーソナルコンピュータにダウンロードします（Natural Connection がインストールされている場合にのみ使用可能）。
ED	オブジェクトのソースコードを編集します（システムコマンド EDIT と等価）。
EX	オブジェクトを実行します（システムコマンド EXECUTE と等価）。
LC	オブジェクトのソースコードをデフォルトのコードページ *CODEPAGE に変換してリストします（LIST *object-name* CONVERTED と等価）。
LD	オブジェクトのディレクトリ情報をリストします（LIST DIRECTORY *object-name* と等価）。
LE	オブジェクトのソースコードを展開形式でリストします（LIST *object-name* EXPAND * と等価）。
LF	フォーマットされたデータエリアまたはマップを表示します（LIST *object-name* FORMATTED と等価）。
LI	オブジェクトのソースコードをリストします。
LN	サブルーチンやクラス（カタログ化オブジェクトが存在する場合にのみ可能）、またはリソースのロングネームを表示します。
NO	CATALOG（カタログ化オブジェクトが存在する場合のみ可能）の実行中に使用された Natural Optimizer Compiler（NOC）オプションを表示します。
OP	CATALOG（カタログ化オブジェクトが存在する場合のみ可能）の実行中に使用された初期、最終、および変更された Natural コンパイラオプションを表示します。 初期および変更されたコンパイラオプションは、Natural バージョン 4.2.5 以降でカタログされたオブジェクトについてのみ表示されます。
PR	オブジェクトのソースコードを印刷します。
RE	オブジェクトの名前を変更します（システムコマンド RENAME と等価）。

コード	機能
RU	オブジェクトのソースコードを実行（つまり、コンパイルして実行）します（システムコマンド RUN と等価）。
ST	オブジェクトをソース形式およびオブジェクト形式で格納します（システムコマンド STOW と等価）。
UC	オブジェクトモジュールを削除します（システムコマンド UNCATALOG と等価）。
.	選択リストのウィンドウを終了します。

オブジェクトのリストのソート

LIST コマンドでは、複数のソート条件を指定して、表示されているオブジェクトのリストをソートすることが可能です。

注意: この機能を使用するには、Natural プロファイルパラメータ SORT の WRKSIZE（ソートプログラムによって使用されるワークバッファのサイズ）を適切な値に設定する必要があります。 ソートできるリストの最大サイズは、ワークバッファのサイズによって制限されます。

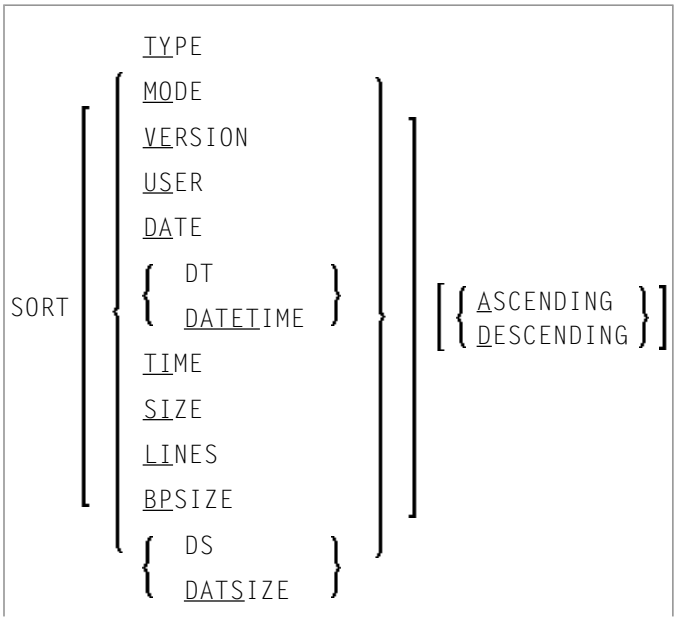
手順 26.2. ソート機能を起動するには

- PF4 キーを押します。

  または:

  オブジェクトのリストに SORT サブコマンドを入力します。

SORT サブコマンドの構文



```
        { CP        }
        { CODE-PAGE }
        MEMBER
        OFF
```

PF4キーを押すとウィンドウが表示され、リストまたはソートフィールドをソートするかどうか、およびソート順を指定できます。次のソートフィールドにより、リストを昇順または降順でソートできます。

ソートフィールド	ソート構文でのキーワード
Natural オブジェクトタイプ	TYPE
プログラミングモード（レポーティングまたはストラクチャードモード）	MODE
バージョン	VERSION
ユーザー ID	USER
日付	DATE
日付／時刻	DATETIME
時刻	TIME
ソースサイズ	SIZE
ソース行の番号	LINES
バッファプールサイズ	BPSIZE
DATSIZE（ローカルデータバッファのサイズ）	DS/DATSIZE
コードページ	CP/CODE-PAGE
サブルーチンまたはクラスのメンバ名（拡張選択リストたけで使用可能）	MEMBER

ソートを開始した後は、選択リストの条件を変更するたびに、ソート済みのリストが作成されます。

手順 26.3. ソートモードをオフに切り替えるには

- サブコマンド SORT OFF を入力します。

  または:

  PF4 キーを押して［**Sort Options**］ウィンドウを呼び出し、ソート機能を無効します。

ソート済みのリストは、ライブラリ WORKPLAN 内の Natural テキストオブジェクトで作成されます。 テキストオブジェクトの名前は、LIST コマンドによって生成されます。 LIST プロファイルが有効になっている場合（下記の「*個別リストプロファイルの定義*」を参照）、テキストオブジェクトとライブラリの名前は LIST プロファイルで指定できます。

オブジェクトのリストの使用例

`LIST *`	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。
`LIST S *`	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 SYS*`	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 M SYS*`	名前が SYS で始まるすべてのマップをリストします。
`LIST C *CODE`	名前が CODE で終わるすべてのコピーコードをリストします。
`LIST NAT*AL`	名前が NAT で始まり AL で終わるすべてのオブジェクトをリストします。NAT と AL の間にどの文字がいくつあるかは関係ありません。この名前には、NATURAL と NATIONAL だけでなく NATAL も含まれます。
`LIST DOO?`	4 文字の名前が DOO で始まるすべてのオブジェクトをリストします。この名前には、DOOR と DOOM は含まれますが、DOO または DOODLE は含まれません。
`LIST M NAT?AL`	名前が NAT で始まり AL で終わり、さらに NAT と AL の間に 1 文字だけある、すべてのマップをリストします。この名前には、NAT1AL と NAT2AL は含まれますが、NATAL または NATIONAL は含まれません。
`LIST M *1*`	名前の中に 1 があるすべてのマップをリストします。
`LIST M F>`	名前が F で始まる最初のマップ以降にある、すべてのマップをリストします。
`LIST M MA<`	先頭のマップから名前が MA のマップまで（存在する場合）、すべてのマップをリストします
`LIST N?T*AL`	NATAL、NATURAL、NAT*vr*AL 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 E* TYPE=PM KIND=S DATE=YEAR SORTED=DATE ASCENDING`	名前が E で始まり、さらに当年内に保存されたプログラムとマップの全ソースオブジェクトのリストを作成します。 リストは、オブジェクトの日付の昇順でソートされます。

ソースのリスト

以下では次のトピックについて説明します。

- リストされたソースに使用するサブコマンド
- サブコマンド FORMAT

- Cursor-Sensitive オブジェクトの選択

リストされたソースに使用するサブコマンド

オブジェクトのソースコードをリストしている場合は、コマンド行に次のサブコマンドのいずれかを入力できます。

サブコマンド	機能
+	1 ページ後にスクロールします。
-	1 ページ前にスクロールします。
++	ソースの終わり（ボトム）までスクロールします。
BOTTOM	
--	ソースの先頭（トップ）までスクロールします。
TOP	
+*n*	*n* 行前にスクロールします。
-*n*	*n* 行後ろにスクロールします。
nnnn	行番号 *nnnn* にスクロールします。
CONVERTED	「オプション」の CONVERTED を参照してください。
DBFNR ON	ソースコードのヘッダー行に、ソースライブラリのデータベース ID（DBID）とファイル番号（FNR）を表示します。
DBFNR OFF	ソースコードのヘッダー行に、ソースライブラリのデータベース ID（DBID）とファイル番号（FNR）を表示しません。 これはデフォルト値です。
EXPAND	「*expand-option*」を参照してください。
FIELDS	マップにのみ適用されます。フィールドサマリ、つまり、マップ内にあるフィールドのリストを表示します。
FIND	指定された *value* を含むソース行のみを表示します。 FIND コマンド自体のみを入力すると、ウィンドウが表示されます。このウィンドウでは、検索する *value* を入力すること、および絶対検索を行うかどうかを指定できます。 FIND コマンドの後に ABSOLUTE を指定しない場合は、*value* が単独の単語である場合にのみ検索されます。 これはデフォルトです。 コマンドの後に ABSOLUTE を指定すると、*value* は長い文字列に含まれている場合でも検索されます。
FIND *value*	
FIND ABSOLUTE *value*	
FORMAT	データエリアとマップにのみ適用されます。フォーマットされたデータエリアやマップ、およびマップに関連する他の項目を表示します。
LAYOUT	マップにのみ適用されます。マップレイアウトを表示します。つまり、ランタイム時にユーザーに対して表示されるものと同じ形式でマップが表示されます。
NUMBERS ON	ソースコードを行番号付きで表示します。 これはデフォルト値です。

サブコマンド	機能
NUMBERS OFF	ソースコードを行番号なしで表示します。
PRINT	リストされているソースを印刷します。
REF	テーブルに指定された *value* を含むソースコード行の行番号を表示します。 REF コマンド自体だけを入力すると、ウィンドウが表示されます。このウィンドウでは、検索する *value* を入力すること、および絶対検索を行うかどうかを指定できます。 REF コマンドの後に ABSOLUTE を指定しない場合は、*value* が単独の単語である場合にのみ検索されます。 これはデフォルトです。 コマンドの後に ABSOLUTE を指定すると、*value* は長い文字列に含まれている場合でも検索されます。
REF *value*	
REF ABSOLUTE *value*	
RULES	マップにのみ適用されます。マップによって使用される処理ルールを表示します。ルールは、それをフィールドに割り当てた順序、およびフィールドごとのランクの順序で表示されます。
SCAN	指定された *value* を含むすべての行を強調表示にします。 ソースは、*value* を含む最初の行にスクロールされます。 SCAN コマンドのみを入力すると、ウィンドウが表示されます。このウィンドウでは、検索する *value* を入力すること、および絶対検索を行うかどうかを指定できます。 SCAN コマンドの後に ABSOLUTE を指定しない場合は、*value* が単独の単語である場合にのみ検索されます。 これはデフォルトです。 コマンドの後に ABSOLUTE を指定すると、*value* は長い文字列に含まれている場合でも検索されます。
SCAN *value*	
SCAN ABSOLUTE *value*	
SCAN= or SC=	最後の SCAN *value*（または PF5 を押す）の次のオカレンスをスキャンします。
SETTINGS	マップにのみ適用されます。マップのマップ設定を表示します。
ZOOM [*expand-type*...10] *object-name*	ZOOM コマンドに単一の *object-name* を指定すると、リストされているソース内の名前をカーソルでマークするのと同じ効果があります（「*Cursor-Sensitive オブジェクトの選択*」を参照）。選択したオブジェクトは、ウィンドウに表示されます。 *object-name* または *object-name-range* にアスタリスク／ワイルドカード表記を使用している場合、選択したすべてのオブジェクトは、リストされたソース内で参照されている順序でウィンドウに表示されます。 *expand-type* の指定方法は、*expand-option* の場合と同じです。 ZOOM で呼び出されたウィンドウに表示されるオブジェクトには、通常のリストされたソースと同じサブコマンドが使用可能です（PRINT、EXPAND、および ZOOM を除く）。 また、アスタリスクやワイルドカード表記を使用して複数のオブジェクトを表示している場合は、コマンド NEXT および PREV（また
ZOOM [*expand-type*...10] *object-name-range*	

サブコマンド	機能
	は PF4 キーと PF5 キー）を使用して、ウィンドウ内の 1 つのオブジェクトから、それぞれ次のオブジェクトまたは前のオブジェクトに移動できます。
.	Exit.

注意: デフォルトでは、ソースコードのヘッダー行に、ソースライブラリのデータベース ID（DBID）とファイル番号（FNR）は表示されません。 ソースライブラリの DBID と FNR を表示する必要がある場合は、サブコマンド `DBFNR ON` を入力するか、または `LIST` プロファイルでパラメータ `SOURCE-LIST-WITH-DBID-FNR` を"Y"に設定します（下記の「*個別リストプロファイルの定義*」を参照）。

サブコマンド FORMAT

このサブコマンドは、格納されたデータエリア（つまり、ソースオブジェクトとカタログ化オブジェクトのタイムスタンプが同じ）およびマップにのみ適用されます。

データエリアの場合、このサブコマンドはオプション `FORMATTED` に相当します。

List プロファイルでは、データエリアのデフォルトの表示方法を指定できます。

- フォーマット済み（つまり、`DEFINE DATA` ステートメントと類似した表示）
- 未フォーマット（つまり、Natural データエリアエディタと類似した表示）

オブジェクトのリストでは、サブコマンド `DEFINE-DATA ON/OFF` を使用して、`LIST` コマンドを実行するデフォルトの時刻を設定できます。

データエリアがデフォルトのフォーマットでリストされているか、またはデータエリアのソースコードを `DEFINE DATA` フォーマットに変換できない場合は、対応するメッセージが表示され、データエリアは未フォーマットの状態でリストされます。

マップに対してサブコマンド `FORMAT` を入力するとウィンドウが表示され、表示されているマップに関連する 1 つまたは複数の追加項目を選択できます。

- マップ設定（サブコマンド `SETTINGS` に相当）。
- マップレイアウト（サブコマンド `LAYOUT` に相当）。 この項目を選択すると、充填文字を指定するオプションを使用して、入力フィールド（`AD=A` と `AD=M`）および出力フィールド（`AD=O`）を表示できます。 充填文字として任意の文字を指定できます。
- フィールドサマリ（サブコマンド `FIELDS` に相当）。
- 処理ルール（サブコマンド `RULES` に相当）。

選択した項目が、選択ウィンドウに表示されている順序で連続的に表示されます。

`FORMAT` モードでは、通常のリストされたソース（上記を参照）と同じ、スクロール用のサブコマンド（`B` を除く）およびサブコマンド `FIELDS`、`LAYOUT`、`PRINT`、`RULES`、`SETTINGS` が使用可能です。 以下で説明されているように、各項目には追加のサブコマンドが使用できます。

マップレイアウトの追加サブコマンド

S>*n*	マップレイアウトを *n* 列だけ右にシフトします。
S<*n*	マップレイアウトを *n* 列だけ左にシフトします。

フィールドサマリリストの追加サブコマンド

EXTEND	すべてのマップフィールドの拡張フィールド編集情報を表示します。 個別のフィールドの拡張フィールド編集情報を表示するには、フィールドサマリリストにあるフィールド名をカーソルでマークし、Enter キーを押します。
RULES *nn*	フィールド *nn* に割り当てられている処理ルールを表示します。*nn* は連続したフィールド番号です（フィールドサマリリストの先頭の列）。 また、フィールドの処理ルールを表示するには、コマンド行に R を入力し、続いてフィールドサマリリストにあるフィールド名をカーソルでマークして Enter キーを押します。
SCAN [ABSOLUTE] *value*	「*リストされたソースに使用するサブコマンド*」と同じです。
SCAN =	

処理ルールの追加サブコマンド

SCAN [ABSOLUTE] *value*	「*リストされたソースに使用するサブコマンド*」と同じです。
SCAN =	

Cursor-Sensitive オブジェクトの選択

リストされているソースの中では、そのソース内で参照されているオブジェクトの *name* をカーソルで選択できます。選択したオブジェクトのソースは、ウィンドウにリストされます。

ウィンドウ内に表示されているソースには、「通常」 のリストされたソースと同じサブコマンドが利用可能です（PRINT、EXPAND、および ZOOM を除く）。

個別リストプロファイルの定義

LIST コマンド用に個別のプロファイルを定義できます。 そのために、Natural ではテキストオブジェクト LISTPROF が SYSLIB で提供されています。

LISTPROF には、対応するデフォルト（COUNT-SOURCE-LINES など）を指定して、一般またはユーザー固有のプロファイルを入力できます。 これらのデフォルトは、LIST コマンドを開始するときに使用されます。

手順 26.4. LISTPROF に定義された値を有効にするには

1 テキストオブジェクト LISTPR-S を、SYSLIB から任意のライブラリにコピーします。

2 変更を加えます。

3 テキストオブジェクト LISTPR-S を、名前に LISTPROF を指定して保存します。

4 テキストオブジェクト LISTPROF をライブラリ SYSLIB にコピーします。

5 LIST コマンドを呼び出します。

詳細については、ライブラリ SYSLIB にあるテキストオブジェクト LISTPR-S を参照してください。

27 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。

LIST COUNT	オブジェクトの総数を表示します。
LIST COUNT *	オブジェクトタイプごとの数を表示します。
LIST COUNT *name*<**<**	名前が *name* 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。
LIST COUNT *name*>	名前が *name* 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。
LIST COUNT*name* *****	名前が *name* で始まるオブジェクトのみの数を表示します。

注意: `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。

28 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。

29 LISTSQL

```
LISTSQL [object-name]
```

LISTSQL コマンドは、Natural for DB2 および Natural for SQL/DS でのみ使用可能です。

コマンド `LISTSQL` を使用すると、データベースアクセスに関係するプログラミングオブジェクトのソースコード内にあるこれらの Natural ステートメント、およびステートメントが変換された対応する SQL コマンドをリストできます。

詳細については、次を参照してください。

- 『*データベース管理システムインターフェイス*』ドキュメントの「*Natural for DB2*」にある「*LISTSQL* コマンド」
- 『*データベース管理システムインターフェイス*』ドキュメントの「*Natural for SQL/DS*」にある「*LISTSQL* コマンド」

30 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。

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

手順 30.1. 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。

31 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。

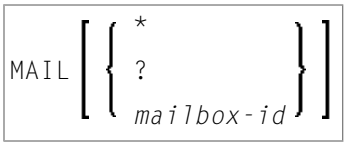
LOGON では、すべての Natural グローバルデータエリア、アプリケーション独立変数（AIV）、SET KEY ステートメントで行われたすべての割り当て、保持されている ISN リストが解放されます。 DDM バッファエリア内の DDM も解放されます。

『*Natural の使用*』ドキュメントの「*ライブラリの命名規則*」も参照してください。

LOGON *library-id*	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。	
LOGON *library-idpassword*	Adabas パスワードを指定します。『*Natural Security*』ドキュメントの「*Library Maintenance*」セクションで「*Session Parameters*」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*』ドキュメントの「*Logging On*」を参照してください。

32 MAIL



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。

MAIL	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。
MAIL *	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。
MAIL ?	
MAIL *mailbox-id*	*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「*Mailboxes*」を参照してください。

33 MAINMENU

MAINMENUコマンドを使用すると、Naturalメインメニューモードを有効または無効にすることができます。

このコマンドは、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。

MAINMENU	メインメニューモードをオンに切り替えます。 これがデフォルトです。
MAINMENU ON	
MAINMENU OFF	メインメニューモードをオフに切り替えます。
MAINMENU *user-program*	Naturalメインメニューの代わりにユーザー定義プログラムを起動することによって、ユーザー定義メニューを呼び出します。

Naturalプロファイルパラメータ MENU も参照してください。

34 NOCOPT

NOCOPT

NOCOPT コマンドを使用すると、Natural の起動中に指定された Natural Optimizer Compiler オプションの現在の設定を表示または変更できます。

NOCOPT の詳細については、『*Natural Optimizer Compiler*』ドキュメントの「*Activating the Optimizer Compiler*」を参照してください。

35 NOCSHOW

NOCSHOW

NOCSHOW コマンドを使用すると、Natural Optimizer Compiler の PGEN オプションによって生成される出力のバッファ情報を提供できます。

NOCSHOW の詳細については、『*Natural Optimizer Compiler*』ドキュメントの「*Optimizer Options*」セクションで「*Output of the PGEN Option*」を参照してください。

36 NOCSTAT

NOCSTAT

NOCSTAT コマンドを使用すると、Natural Optimizer Compiler での処理に適しているプログラムの統計データを提供できます。

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

37 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 Command*」を参照してください。

38 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*』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。

READ *object-name*	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 *object-name* 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。
READ *object-name* *library-id*	読み込むオブジェクトが存在するライブラリです。 *object-name* と *library-id* の両方を指定した場合、Natural 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。

39 RENAME

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す。

『*Natural*』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。

RENAME	パラメータを指定せずに RENAME コマンドを発行すると、［**Rename Object**］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。	
old-name	*old-name* 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。	
new-name	今後オブジェクトは *new-name* に指定した名前で保存されます。	
new-type	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 *new-type* 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。 *new-type* に指定可能な値は、次のとおりです。	
	4	クラス
	5	プロセッサ
	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
	O	マクロ
	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

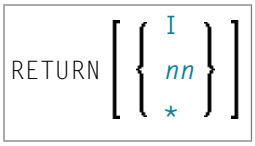
40 RENUMBER

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。

RENUMBER	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。
RENUMBER (*n*)	*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。

41 RETURN



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。

アプリケーションプログラミングインターフェイス：USR1026N 『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。

RETURN	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返されます（SETUP 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。 **Natural Security** 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。
RETURN I	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）。
RETURN *nn*	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。
RETURN *	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。

42 ROUTINES

ROUTINES

ROUTINES コマンドを使用すると、現在のライブラリ内にあるどのオブジェクトがどの外部サブルーチンおよびクラスを使用しているかを確認できます。

現在のライブラリにあるすべてのオブジェクトが、オブジェクトによって呼び出される外部サブルーチンやクラスの名前、外部サブルーチンやクラスが含まれているサブルーチンやクラスのオブジェクト名とともにリストされます。

オブジェクト自体がサブルーチンまたはクラスである場合は、それに含まれるサブルーチンまたはクラスの名前が表示されます。

43 RPCERR

RPCERR

RPCERR コマンドを使用すると、RPC に関連している場合には、最後の Natural エラー番号とメッセージを表示できます。また、最後のBroker理由コードと関連メッセージも表示できます。さらに、最後の Broker コールからのノードおよびサーバー名も取得可能です。

詳細については、『*Natural リモートプロシージャコール（RPC）*』ドキュメントの「*Natural RPC 環境の運用*」セクションで「*RPC セッションのステータスのモニタ*」を参照してください。

44 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を実行できます。

以下の項目も参照してください。

『*Natural システムアーキテクチャ*』ドキュメントの「*Natural コンパイラ*」
『*Natural の使用*』ドキュメントの「*オブジェクトの命名規則*」

RUN	*program name* 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がコンパイルおよび実行されます。
REPEAT	REPEAT を指定すると、プログラム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。 プログラムが終了すると、Naturalはコマンドモードに入ります。
program-name	実行するプログラムの名前です。 *program-name* 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。
library-id	実行するプログラムが存在するライブラリです。 *program-name* と *library-id* 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。 *library-id* の設定値を SYS（SYSTEM を除く）で始めることはできません。

45 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ソースオブジェクトとして Natural システムファイルに保存するために使用します。

以下の項目も参照してください。

『*Natural の使用*』ドキュメントの「*オブジェクトの命名規則*」
背景情報については、『*Natural システムアーキテクチャ*』ドキュメントの「*Natural コンパイラ*」

注意: 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。

SAVE	*object-name*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現在のライブラリに保存されます。 既存のソースコードは置換されます。
SAVE *object-name*	新しいソースオブジェクトが作成されます。 *object-name* として、保存するソースオブジェクトの名前を指定します。 新しいソースオブジェクトは現在のライブラリに保存されます。 ソースオブジェクトが存在する場合、コマンドは拒否されます。
SAVE *object-name* *library-id*	ソースオブジェクトを別の名前で保存するか、または新しく作成したオブジェクトを保存する場合、ソース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、*object-name* の後に必要な *library-id* を指定する必要があります。 新しいソースオブジェクトが作成されます。ソース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。

46 SCAN

SCAN

SCAN コマンドは、オブジェクト内の文字列を検索するために使用します。このとき、オプションとして文字列を別の文字列で置換することもできます。

対象になるのは、1つのオブジェクト、指定した設定で始まるすべてのオブジェクト、またはライブラリ内のすべてのオブジェクトです。 SCAN も、対象を特定のオブジェクトタイプに制限できます。

⚠ **重要:** ソースワークエリアは SCAN コマンドによって使用されます。 そのため、SCAN コマンドを使用する前に、SAVE または STOW コマンドを発行する必要があります。

この章では、次のトピックについて説明します。

メニューオプション

SCAN コマンドを入力すると、次の内容の［SCAN］メニューが表示されます。

フィールド	入力設定	
コード	T	統計 次の情報を返します。 ■ スキャンされたオブジェクト数 ■ スキャン対象値が見つかったオブジェクト数 ■ スキャン対象値が見つかったソースコード行の数
	L	スキャン対象値を含むオブジェクトのリスト スキャン設定が見つかったすべてのオブジェクトのリストを表示します。このリストから、さらに処理するために個々のオブジェクトを選択できます。 必要に応じて、結果画面の行を直接、または適切な SCAN サブコマンド（下記参照）を使用して、変更することができます。 スキャンされた行を任意のオブジェクト用に変更できます。ただし、マップおよびデータエリア、またはロックされたオブジェクト（「ソースオブジェクトのロック」を参照）を除きます。 オブジェクト全体を変更するには、E サブコマンドを入力して、対応するエディタを呼び出します。 以前に結果画面で何らかの変更を行っていた場合は、更新を確認するプロンプトが表示されます。

フィールド	入力設定	
		オブジェクトを編集し終わったら、オブジェクトを保存してエディタを終了します。 その後、スキャン処理を継続できます。
	S	スキャン対象値のあるオブジェクト行 スキャン対象値が見つかった各ソースコード行を次々に表示します。 必要に応じて、結果画面の行を直接、または適切な SCAN サブコマンド（下記参照）を使用して、変更することができます。 スキャンされた行を任意のオブジェクト用に変更できます。ただし、マップおよびデータエリア、またはロックされたオブジェクト（「*ソースオブジェクトのロック*」を参照）を除きます。 オブジェクト全体を変更するには、E サブコマンドを入力して、対応するエディタを呼び出します。 以前に結果画面で何らかの変更を行っていた場合は、更新を確認するプロンプトが表示されます。 オブジェクトを編集し終わったら、オブジェクトを保存してエディタを終了します。 その後、スキャン処理を継続できます。
スキャン対象値	スキャンする文字列。 **注意:** Natural によって小文字が大文字に変換されるのを防ぐには、端末コマンド %L を使用します。	
置換値	スキャン対象値を置き換える値。 ［**Replace value**］オプションは、マップ、データエリア、記録、ダイアログと関数、またはロックされたオブジェクト（「*ソースオブジェクトのロック*」）には影響しません。	
Library	スキャン対象のライブラリの ID。 デフォルトは現在のライブラリです。 指定されたライブラリが SYSTEM の場合は、FUSER ファイル内のライブラリがスキャンされます。 指定されたライブラリの名前が "SYS" から開始し、SYSTEM ではない場合は、FNAT ファイル内のライブラリがスキャンされます。	
オブジェクト名	スキャン対象のオブジェクト：	
	空白	すべてのオブジェクト
	*	
	object-name>	名前が *name* 以上であるすべてのオブジェクト
	object-name<	名前が *name* 以下であるすべてのオブジェクト
	一定範囲のオブジェクト群でスキャンする場合は、システムコマンド LIST の記述と同様にして、オブジェクト名にアスタリスク表記（*）またはワイルドカード表記（?）を使用できます。 『*Natural 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。	

フィールド	入力設定	
Object type(s)	検索は、特定のオブジェクトタイプに制限できます。使用できるタイプの選択リストを表示するには、このフィールドに疑問符（?）を入力します。 このフィールドを空白のままにするかアスタリスク（*）を入力した場合は、任意のタイプのオブジェクトがスキャンされます。	
完全スキャン	Y	スキャンが「完全」に行われます。つまり、スキャンされる値は、より長い文字列の一部であっても、任意のフォームで見つかります。
	N	デフォルトでは、完全スキャンは行われません。 **注意:** データエリアでは、このパラメータの値に関係なく、スキャンは常に完全スキャンです。
Selection list	Y	ライブラリ、名前、コードTまたはSのタイプ（上記を参照）で指定されたオブジェクトのリストを表示します。このリストから、さらに処理するために（任意の文字でマーキングして）個々のオブジェクトを選択できます。
	N	デフォルトでは、選択リストは表示されません。
Trace	Y	トレース機能を有効にします。
	N	デフォルトでは、トレース機能は無効になります。

SCAN のサブコマンド

次のサブコマンドを、スキャン操作によって生成された結果画面のコマンド行に入力できます。

コマンド	機能
空白	通常のスキャン処理を続けます。
Q	スキャン処理を終了します。
.	
E	フルスクリーンエディタを使用してオブジェクトを編集します。
EDT	エディタを使用してオブジェクトを編集します。
LIST	現在ソースワークエリアに表示されているオブジェクトをリストします。
LET	最後に Enter キーが押された後に行われたすべての行変更を無視します。
I	現在スキャン中のオブジェクトを無視し、いずれの変更も保存せず、次のオブジェクトに移ってスキャンを続行します。
.D	行を削除します。行の横に、削除されたことを示す "D" が表示されます。
.L	最後に Enter キーが押された後に行われたすべての変更を無視します。行コマンド .D で以前に削除された行も復元します。

編集ルール

- 結果画面のソースオブジェクトの行の長さは、72 文字に制限されます。 72 文字を超える行は "L" でマークされ、変更はできません。
- ［**Replace value**］オプションが使用されている場合、オブジェクトが結果画面で修正された場合、またはその両方の場合は、I、Q またはドット（.）が指定されていない限り、次のオブジェクトがスキャンされる前にオブジェクトが常に保存されます。
- PASSW、PASSWORD=、CIPHER= または CIPH= を含む行は無視されます。

SCAN キーワード

SCAN 機能は、次のキーワードを指定することにより、バッチモードまたはオンラインモードで呼び出すことができます。

キーワード	説明
FUNC	ファンクションコード
SVAL	スキャン対象値
LIB	Library
RVAL	置換値
OBJ	オブジェクト名
TYPE	オブジェクトタイプ
ABSOL	完全スキャン

注意: ダイレクトコマンドでは、意図しないスキャン／置換結果を回避するために、空白が埋め込まれた値を避ける必要があります。 オンラインモードでは、空白が埋め込まれた値のスキャンは、［**SCAN**］メニューでのみ使用できます。

キーワードを使用した **SCAN** コマンドの例：

```
SCAN FUNC=S,SVAL=value,LIB=SYSTEM,OBJ=PGM0*,TYPE=S
```

```
SCAN FUNC=S,SVAL=value,RVAL=value,OBJ=PGM1
```

バッチモードでの SCAN

SCAN コマンドは、呼び出しごとに 1 つの機能のみ処理して、指定された無効なデータの影響を最小化します。 上で説明したキーワードまたは位置パラメータを使用できます。

位置パラメータは次のように指定します。

```
SCAN func, scan-value, replace-value, library, object-name, object-type, absolute
```

位置パラメータに使用できる値は、「メニューオプション」で説明しています。

重要: 小文字または埋め込みの空白を含む値をスキャンするには、SCAN コマンドと同じバッチジョブの行に *scan-value* を指定せずに、別のデータ行で、オンラインマップに従ったデータを入力します。「メニューオプション」を参照してください。

埋め込みの空白があるスキャン／置換値の使用例：

```
SCAN S,MOVE LEFT,MOVE RIGHT,SYSTEM,PGMO*,N,*,N,N
```

Natural Security 環境での SCAN

Natural Security 環境で SCAN を使用できるようにするには、システムコマンド LIST、EDT、EDIT および READ を現在のライブラリのセキュリティプロファイルで許可する必要があります。［**Replace value**］オプションを使用する場合、またはソースを変更可能にする場合は、システムコマンド SAVE も許可する必要があります。

Natural Security 環境では、SCAN コマンドが一部のライブラリで使用できない場合があります。

ライブラリに対してストラクチャードモードのみ許可されている場合、レポーティングモードのオブジェクトはスキャンできますが、変更はできません。

47 SCRATCH

このコマンドは、互換性保持のためにのみサポートされています。代わりに DELETE コマンドを使用することを強くお勧めします。

48 SETUP

```
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。

この章では、次のトピックについて説明します。

アプリケーションプログラミングインターフェイス：USR1026N 『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。

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。

SETUP	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。
application-name	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。 *application-name* 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。 *application-name* 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。
command-name	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。 *command-name*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。 *command-name* 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。
I	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。 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。

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 I` 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。

49 SQLERR

SQLERR

このコマンドは、Natural for DB2 と Natural for SQL/DS でのみ使用できます。 SQL エラーに関する情報を取得するために使用できます。

詳細については、次を参照してください。

- 『*データベース管理システムインターフェイス*』ドキュメントの「*Natural for DB2*」部分にある「*SQLERR コマンド*」
- 『*データベース管理システムインターフェイス*』ドキュメントの「*Natural for SQL/DS*」部分にある「*SQLERR コマンド*」

50 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。

以下の項目も参照してください。

『*Natural システムアーキテクチャ*』ドキュメントの「*Natural コンパイラ*」
『*Natural の使用*』ドキュメントの「*オブジェクトの命名規則*」

STOW	*object-name* を指定しないでコマンドを使用する場合は、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。 既存のソースおよびオブジェクトコードは置換されます。
STOW *object-name*	このコマンドは、*object-name* 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。
STOW *object-name* *library-id*	*object-name* と *library-id* 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。

注意: 有効ではないパラメータモジュールに FDIC システムファイルが指定された場合は、STOW 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。

51 STRUCT

STRUCT

このコマンドには次の 2 つの目的があります。

- 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できます。
- さまざまな表示機能によって、プログラムの構造が明確になるため、構造の不整合を検出できます。

ただし、STRUCT は実際にカタログ化されているかどうかにかかわらず Natural ソースを処理するため、ソースの構文の正しさは解析されません。 ほとんどの場合、STRUCT は適切に構造化されたソース行を提供しますが、不明確で期待どおりに構造化されないソース行がある場合があります。

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

システムコマンド STRUCT を入力すると、［STRUCT］メニューが表示されます。 このメニューの機能は次のとおりです。

ワークエリアへの構造化されたソースの生成

この機能を使用して、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。

この機能は、エディタコマンド STRUCT の機能と同じです。

インデントでは、ソースコード行の長さが考慮されます。つまり、インデントされる行は、右のマージンを越えてシフトされません。「適切な」 インデントで右マージンを越える行のシフトが必要な場合は、可能な限り右に移動されるだけで、マージンを越えることはありません。

生成機能では、次のオプションを指定できます。

フィールド	説明	
ソース名	このフィールドには、インデントするソースの名前を入力します。 指定されたソースは、システムファイルからワークエリアに読み込まれてインデントされます。 ソース名を指定しない場合は、エディタのワークエリアに現在存在するオブジェクトがインデントされます。 ワークエリアが空の場合は、ソース名を指定する必要があります。	
Shift setting	このフィールドでは、ソースコード行をインデントする位置の数（1～9）を入力できます。 デフォルトでは、インデントは 2 つの位置ずつ行われます。	
Align Comments	Y	各コメント行は、上のステートメント行と同じだけインデントされます。ただし、行の先頭から始まるコメントはインデントされません。
	N	コメント行はインデントされません。
	L	コメント行は左寄せで表示されます。
Display Messages	Y	ストラクチャードモードプログラムを示すメッセージがワークエリアに生成され、「正しく」インデントできなかったソースコード行のリスト（上記を参照）が表示されます。
	N	このようなメッセージは表示されません。
Return to STRUCT	Y	生成機能が実行された後で、STRUCT メニューに戻ります。
	N	生成機能が実行された後で、STRUCT コマンドを発行した画面に戻ります。

注意: 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 Generate（生成）機能を実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
```

同じプログラムに Generate Structured Source 機能を適用すると、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
```

ソースの構造の表示

この機能では、オブジェクトの構造を明確にするいくつかの情報項目とともに、オブジェクトのソースコードを表示できます。

表示機能には、次のオプションがあります。

フィールド	説明
ソース名	このフィールドでは、表示するソースの名前を入力します。 指定されたソースがシステムファイルから読み込まれて表示されます。 ソース名を指定しない場合は、エディタのワークエリアに現在存在するオブジェクトが表示されます。 ワークエリアが空の場合は、ソース名を指定する必要があります。
Display Compressed	Y 同じ構造レベルにあるソースコード行は表示されません。画面の右側の構造テーブルが変更される原因となる行のみ表示されます。 一連の行番号のギャップから、表示されている特定の 2 行の間で表示されていない行数がわかります。 N すべてのソースコード行が表示されます。
Return to STRUCT	Y 表示機能が実行された後で、STRUCT メニューに戻ります。 N 表示機能が実行された後で、STRUCT コマンドを発行した画面に戻ります。

次の情報が表示されます。

Line Numbers	ステートメントブロックを閉じるすべてのステートメントについて、ステートメントブロックを開始する対応するステートメントのソースコード行番号が、ソースコードの左に表示されます。
Structure Table	ソースコードの右には、開いているステートメントブロックのインジケータを含むテーブルが表示されます。 開いている各ステートメントブロックにつき 1 文字が表示されます。 異なる文字は、異なるタイプのステートメントを表します（文字の説明を表示するには、PF1 キーを押します）。 ソースコード内の構造の不整合は、構造テーブルに表示されるメッセージで示されます。

構造情報の表示例：

```
14:17:47 - Structured Source ABC in Library XYZ -                  2003-02-04
  0010      DEFINE DATA LOCAL                                    *0
  0020      1 EMPL VIEW OF EMPLOYEES                             *0
  0030        2 PERSONNEL-ID                                     *0
  0040        2 FULL-NAME                                        *0
  0050          3 FIRST-NAME                                     *0
  0060          3 NAME                                           *0
  0070      1 VEHI VIEW OF VEHICLES                              *0
```

```
0080         2 PERSONNEL-ID                                          *O
0090         2 MAKE                                                  *O
0100 0010 END-DEFINE                                                 *O
0110      FIND EMPL WITH NAME = 'ADKINSON'                           *F
0120        IF NO RECORDS FOUND                                      *FJ
0130          WRITE 'NO RECORD FOUND'                                *FJ
0140 0120   END-NOREC                                                *FJ
0150        FIND (1) VEHI WITH PERSONNEL-ID = EMPL.PERSONNEL-I       *FF
0160          DISPLAY EMPL.PERSONNEL-ID FULL-NAME MAKE               *FF
0170 0150   END-FIND                                                 *FF
0180 0110 END-FIND                                                   *F
0190      END                                                        *
PF1=Help, PF2=Menu, PF3=Exit, PF6=Top, PF12=Cancel.
```

ワークエリアの現在の内容は、表示されるソースの影響を受けません。

ソースの構造の印刷

この機能では、オブジェクトのソースコードを構造情報とともに印刷できます。

印刷機能はソースの構造の表示機能に相当しますが、出力は画面に表示されずにプリンタに送信されます。

印刷機能には、表示機能と同じオプションがあります。

ワークエリアへのソース構造の書き込み

この機能では、システムファイルからソースを読み取り、構造情報に加えて、構造情報を説明するソースの先頭の複数行（行番号 0000）とともに、エディタのワークエリアに書き込むことができます。

書き込み機能には、ソースの構造の表示機能と同じオプションがありますが、ソース名を指定する*必要があります*。

ソースとその構造情報は、ワークエリアにテキストとして書き込まれ、システムコマンド `EDIT` で編集できます。

52 SYSADA

SYSADA

このコマンドは、ライブラリ SYSADA に含まれる ADACALL ユーティリティを呼び出します。

ADACALL ユーティリティによって、Adabas ダイレクトコール（ネイティブコマンド）をメインフレーム Natural から Adabas データベースに直接発行できます。

ADACALLユーティリティは、学習目的またはさまざまな問題やシナリオのテスト／分析に使用できます。

詳細については、『*ユーティリティ*』ドキュメントの「*ADACALL - Adabas ダイレクトコールの発行*」を参照してください。

53 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、『*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。

54 SYSBPM

SYSBPM

このコマンドは SYSBPM ユーティリティを呼び出します。

SYSBPM ユーティリティは、バッファプールキャッシュを含む Natural バッファプールの現在のステータス、およびバッファプールとバッファプールキャッシュに現在存在するオブジェクトに関する統計情報を提供します。

SYSBPM は、管理機能も提供します。

詳細については、『*ユーティリティ*』ドキュメントの「*SYSBPM ユーティリティ - バッファプールの管理*」を参照してください。

55 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、またソースのコードページ割り当てを確認または変更するために使用できます。

詳細については、『*ユーティリティ*』ドキュメントの「*SYSCP ユーティリティ - コードページの管理*」を参照してください。

56 SYSDB2

SYSDB2

このコマンドは、Natural for DB2 がインストールされている場合に、DB2 用の Natural ツールを呼び出します。

詳細については、『*データベース管理システムインターフェイス*』ドキュメントの「*Natural for DB2*」部分にある「*Natural Tools for DB2 の使用*」を参照してください。

57 SYSDDM

SYSDDM

このコマンドは、Natural データ定義モジュール（DDM）の作成およびメンテナンスに必要な機能を提供する SYSDDM ユーティリティを呼び出します。

詳細については、『エディタ』ドキュメントの「*SYSDDM ユーティリティ*」を参照してください。

Natural Single Point of Development の注意事項：

このコマンドは、リモート開発環境の Natural スタジオコマンド行経由では使用できません。DDM はノード DDM の下のツリービューにリストされ、SYSDDM ユーティリティの全機能はコンテキストメニューまたはメニューバー経由で有効になるためです。

58 SYSEDT

SYSEDT

このコマンドは、エディタバッファプールサービスの SYSEDT ユーティリティを呼び出します。 SYSEDT ユーティリティは、Natural 管理者のみを対象としています。 次の処理を行うために使用します。

- エディタバッファプールに関するパラメータとランタイム情報の表示
- パラメータの修正
- 論理作業ファイルおよびリカバリファイルの削除

詳細については、『*ユーティリティ*』ドキュメントの「*SYSEDT ユーティリティ - エディタバッファプールサービス*」を参照してください。

59 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、『*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。

60 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、『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。

61 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。

62 SYSFILE

```
SYSFILE [ { WORKFILE } ]
        [ { PRINTER  } ]
```

このコマンドは SYSTP ユーティリティの SYSFILE 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。

アプリケーションプログラミングインターフェイス：USR1007N 『*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。

SYSFILE	コマンド SYSFILE のみ入力した場合は、ワークファイルと出力ファイルの割り当てが順次に表示されます。
SYSFILE WORKFILE	ワークファイル割り当てが個別に表示されます。
SYSFILE PRINTER	出力ファイル割り当てが個別に表示されます。

詳細については、『*ユーティリティ*』ドキュメントの「*一般的な SYSTP 機能*」セクションの「*Natural 出力／ワークファイル - SYSFILE*」、および『*オペレーション*』ドキュメントの出力ファイルとワークファイルのサポートに関するプラットフォーム固有情報を参照してください。

63 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、『ユーティリティ』ドキュメントの「*SYSMAIN ユーティリティ - オブジェクトメンテナンス*」を参照してください。

64 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、『*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。

65 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、『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。

66 SYSPARM

SYSPARM

このコマンドは SYSPARM 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、Natural プロファイルに保存されている Natural プロファイルパラメータの文字列を作成およびメンテナンスします。

これらの Natural プロファイルは、Natural セッション開始のためにプロファイルパラメータ PROFILE で呼び出すこともできます。

SYSPARM コマンドには、主にバッチモードで使用されるさまざまなパラメータがあります（SYSPARM ユーティリティの説明の「*ダイレクトコマンドとバッチ処理*」を参照）。

詳細については、『*ユーティリティ*』ドキュメントの「*SYSPARM ユーティリティ*」を参照してください。

67 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。

アプリケーションプログラミングインターフェイス：USR0050N、USR2031N 『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。

コマンドを入力すると、ウィンドウが表示され、インストールされている製品ごとに次の情報が表示されます。

- 製品名
- 製品バージョンとリリース
- システムメンテナンス（SM）レベル
- インストール日と時刻

リストにある製品の一部については、ダイアログの［**Cmd**］列に行コマンドでマークすることにより、追加情報を取得できます。

行コマンド	説明
EX	詳細な製品情報を表示します。
HI	製品情報の履歴を表示します。
SC	製品のサブコンポーネントを表示します。

注意: 一部の製品については、行コマンドを使用できません。

SYSPROD コマンドインターフェイス（バッチ）

バッチ処理について、または書式設定されていないオンライン出力については、追加のコマンド行パラメータを指定して SYSPROD を呼び出すことができます。

```
SYSPROD { ALL                } [EX] [SC]
        { <product-code>     }
```

68 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

アプリケーションプログラミングインターフェイス：USR0010N、USR2013N、USR3013N 『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。

69 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、『*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。

70 SYSTP

SYSTP

このコマンドは、Natural のさまざまな TP モニタ固有特性をモニタリングおよび制御できる SYSTP ユーティリティを呼び出します。

詳細については、『*ユーティリティ*』ドキュメントの「*SYSTP ユーティリティ*」を参照してください。

71 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）

注意:

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。
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。

アプリケーションプログラミングインターフェイス：`USR2026N` 『*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。

72 TEST

このコマンドはデバッガを呼び出します。

TEST	パラメータを指定せずにシステムコマンド TEST を入力した場合は、デバッガのメインメニューが表示されます。
TEST ON	デバッガのテストモードを有効にします。
TEST OFF	デバッガのテストモードを無効にします。
debugger-commands	デバッグ機能を実行するダイレクトコマンドについては、『デバッガ』ドキュメントの*コマンドの概要とコマンド構文*に関するセクションで説明されています。

アプリケーションの任意のポイントからデバッガを呼び出すために、端末コマンド %<TEST も使用できます。

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。

『ユーティリティ』ドキュメントには、オンラインのテストおよびモニタに使用される他の Natural ユーティリティに関する情報も記載されています。

Natural Single Point of Development の注意事項：

このコマンドはメインフレーム専用です。 ユーザーが TEST というプログラムを作成した場合は、Windows（あるいは、UNIX または OpenVMS）ローカル環境でこのコマンドが発行されたときに、Natural によってそのプログラムが実行されます。 メインフレーム上の開発サーバーへのアクティブな接続がある場合は、Natural for Windows のもとでこのコマンドが発行されたときに、メインフレームユーティリティ TEST が呼び出されます。

73 TEST DBLOG

```
TEST DBLOG [parameters]
```

このコマンドは、データベースコールのログに使用される DBLOG ユーティリティを呼び出します。

TEST DBLOG	DBLOG ユーティリティを有効または無効にします。
parameters	TEST DBLOG に適用されるパラメータについては、『ユーティリティ』ドキュメントの「*TEST DBLOG コマンド*」セクションで説明されています。

詳細については、『ユーティリティ』ドキュメントの「*DBLOG ユーティリティ - データベースコールのロギング*」を参照してください。

『ユーティリティ』ドキュメントには、オンラインのテストおよびモニタに使用される他の Natural ユーティリティに関する情報も記載されています。

Natural Single Point of Development の注意事項：

メインフレーム上の開発サーバーへのアクティブな接続がある場合は、Natural for Windows のもとでこのコマンドが発行されたときに、Natural メインフレームユーティリティ DBLOG が呼び出されます。

74 UNCATALOG

このコマンドは、互換性保持のためにのみサポートされています。代わりに DELETE コマンドを使用することを強くお勧めします。

75 UNLOCK

このコマンドは、Natural メインフレーム環境の Natural ソースオブジェクトのローカルロック解除に使用します。

ロックされているソース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しできます。 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。

注意:

1. UNLOCK システムコマンドを使用する前提条件として、プロファイルパラメータ SLOCK を PRE に設定する必要があります。
2. 見つかったロックレコード数が多い場合は、表示されるリストがソートされない場合があります。 対処措置：ソートプログラムで使用されるワークバッファのサイズを増やします。プロファイルパラメータ SORT のキーワードサブパラメータ WRKSIZE を参照してください。

この章では、次のトピックについて説明します。

詳細については、『エディタ』ドキュメントの「ソースオブジェクトのロック」および『パラメータリファレンス』のプロファイルパラメータ「SLOCK」を参照してください。

『*Natural*』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マップが表示されます。

```
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]
       [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
object-name	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。
object-type	A1	*	**Natural** オブジェクトタイプ： *object-type* 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	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
M	マップ
L	ローカルデータエリア
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）

注意: 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リストされます。ここで、行コマンド U を使用して（下記を参照）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。

ロック解除リスト

ファンクションキー

ロック解除リストには、次のファンクションキーがあります。

PF1	ヘルプ	ヘルプを表示します。
PF3	終了	ロック解除リストに戻ります。
PF6	--	リストの先頭に移動します。
PF7	-	前のページに移動します。
PF8	+	次のページに移動します。
PF9	++	リストの終わり。
PF10	<	情報の最初の部分（タイプ、ライブラリ、データベース ID、ファイル番号）。
PF11	>	情報の 2 番目の部分（ロックしたユーザー、ロックした日付）。
PF12	Cancel	UNLOCK コマンドを取り消します。

行コマンド

U	ロック解除リストの［**Cmd**］列には、1 行または複数行にコマンド U を入力して、対応するオブジェクトをロック解除できます。 ロック解除の成功は、［**Message**］列の「"unlocked"」メッセージで示されます。

バッチ処理

エラーが発生しなかった場合は、見つかったすべてのロック済みオブジェクトがロック解除され、対応するメッセージが表示されます。

76 UPDATE

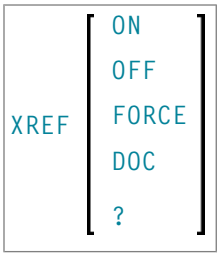
```
UPDATE { ON  }
       { OFF }
```

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。

UPDATE ON	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。
UPDATE OFF	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。

77 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。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。

XREF	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。
XREF ON	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。
XREF OFF	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。
XREF FORCE	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とき、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。
XREF DOC	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とき、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。
XREF ?	XREF ?を使用すると、 XREF コマンドのヘルプ機能を呼び出すことができます。

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

し