Natural for UNIX

オペレーション

バージョン 8.4.1

2017 年 10 月

ADABAS & NATURAL

このマニュアルは Natural バージョン 8.4.1 およびそれ以降のすべてのリリースに適用されます。

このマニュアルに記載される仕様は変更される可能性があります。変更は以降のリリースノートまたは新しいマニュアルに記述されます。

ドキュメント IDは: NATUX-NNATOPERATIONS-841-20200614JA

目次

前書き

このドキュメントには、UNIX 環境で Natural を操作するための情報が記載されています。次の項目で構成されています。

プロファイルパラメータの使用方法	パラメータ階層に関する情報。プロファイルパラメータへの値のスタティックな割り当て方法、ダイナミックな割り当て方法、およびランタイムでの割り当て方法。
システムファイル	システムファイルと Natural オブジェクトのファイルシステムへの格納方法。システムファイル FNAT、FUSER および FDDM に関する情報。
ワークファイル	ワークファイルの定義方法。さまざまなワークファイルフォーマットに関する情報。
Natural バッファプール	Natural でのバッファプールの使用方法および起動方法。
バッファプールモニタ（NATBPMON）の使用	NATBPMON ユーティリティの呼び出し方法。このユーティリティで使用可能なコマンドに関する情報。
バッチモードでの **Natural**	バッチモードでの Natural の実行方法。必要な入力チャネルおよび出力チャネルに関する情報。バッチモードシミュレーションの使用方法。
NATCONV.INI を使用した異なる文字セットのサポート	ファイル *NATCONV.INI* 内のさまざまな文字セットの定義方法。
Natural の出口コード	スタートアップエラーを含む Natural の出口コードに関する情報。
Entire System Server インターフェイスの設定	Entire System Server 製品の Entire System Server インターフェイスをアクティブ化する方法。
SQL データベースアクセスの調整	SQL ドライバのステートメントテーブルの処理の構成方法。
ソートキー計算のユーザー出口 - NATUSK*nn*	他の言語の文字を正しいアルファベット順にソートする方法。
異常終了（アベンド）の処理	シグナルハンドラに関する情報。

数多くの管理機能の実行に使用できる Natural ユーティリティについては、個別に説明しています。詳細については、『*ツールおよびユーティリティ*』ドキュメントを参照してください。

セキュリティについても個別に説明しています。詳細については、『*Natural Security*』ドキュメントを参照してください。

1

表記規則

規則	説明
太字	画面上の要素を表します。
モノスペースフォント	*folder.subfolder:service* という規則を使用して webMethods Integration Server 上のサービスの保存場所を表します。
大文字	キーボードのキーを表します。同時に押す必要があるキーは、プラス記号 (+) で結んで表記されます。
斜体	独自の状況または環境に固有の値を指定する必要がある変数を表します。本文で最初に出現する新しい用語を表します。
モノスペースフォント	入力する必要があるテキストまたはシステムから表示されるメッセージを表します。Program code.
{ }	選択肢のセットを表します。ここから1つ選択する必要があります。中カッコの内側にある情報のみを入力します。{ } 記号は入力しません。
|	構文行で相互排他的な2つの選択肢を区切ります。いずれかの選択肢を入力します。| 記号は入力しません。
[ ]	1 つ以上のオプションを表します。大カッコの内側にある情報のみを入力します。[ ] 記号は入力しません。
...	同じ種類の情報を複数回入力できることを示します。情報だけを入力してください。実際のコードに繰り返し記号 (...) を入力しないでください。

オンライン情報

Software AG マニュアルの **Web** サイト

マニュアルは、Software AG マニュアルの Web サイト (**http://documentation.softwareag.com**) で入手できます。このサイトでは Empower クレデンシャルが必要です。Empower クレデンシャルがない場合は、TECHcommunity Web サイトを使用する必要があります。

Software AG Empower 製品のサポート **Web** サイト

もしまだEmpowerのアカウントをお持ちでないのなら、こちらへempower@softwareag.com電子メールにて あなたのお名前、会社名、会社の電子メールアドレスをお書きの上、アカウントを請求してください。

いったんアカウントをお持ちになれば、Empower **https://empower.softwareag.com/** のeServiceセクションにてサポートインシデントをオンラインで開くことができます。

製品情報は、Software AG Empower 製品のサポート Web サイト (**https://empower.softwareag.com**) で入手できます。

機能および拡張機能に関するリクエストの送信、製品の可用性に関する情報の取得、製品のダウンロードを実行するには、Products に移動します。

修正に関する情報を取得し、早期警告、技術論文、Knowledge Base の記事を読むには、**Knowledge Center** に移動します。

もしご質問があれは、こちらの**https://empower.softwareag.com/public_directory.asp** グローバルサポート連絡一覧の、あなたの国の電話番号を選んで、わたくし共へご連絡ください。

Software AG TECHcommunity

マニュアルおよびその他の技術情報は、Software AG TECHcommunity Web サイト (**http://techcommunity.softwareag.com**) で入手できます。以下の操作を実行できます。

- TECHcommunity クレデンシャルを持っている場合は、製品マニュアルにアクセスできます。TECHcommunity クレデンシャルがない場合は、登録し、関心事の領域として [マニュアル] を指定する必要があります。
- 記事、コードサンプル、デモ、チュートリアルにアクセスする。
- Software AGの専門家によって承認されたオンライン掲示板フォーラムを使用して、質問したり、ベストプラクティスを話し合ったり、他の顧客が Software AG のテクノロジをどのように使用しているかを学んだりすることが可能です。
- オープンスタンダードや Web テクノロジを取り扱う外部 Web サイトにリンクできます。

データ保護

Software AG製品は、EU一般データ保護規則(GDPR)を尊重した個人データの処理機能を提供します。 該当する場合、適切な手順がそれぞれの管理ドキュメントに記載されています。

2 プロファイルパラメータの使用方法

Natural プロファイルパラメータは、操作環境の外観とレスポンスに影響します。

パラメータの詳細については、『パラメータリファレンス』ドキュメントを参照してください。

パラメータ階層

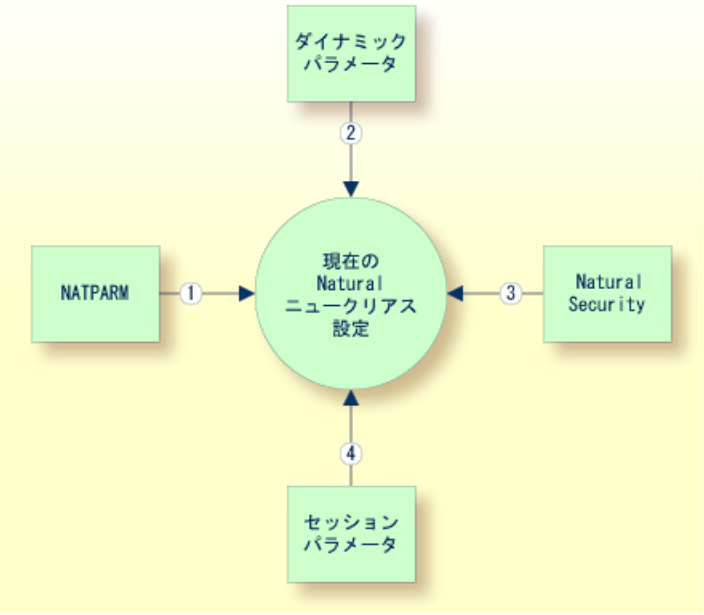
Naturalパラメータの値は異なるソースから取得されます。パラメータの優先度は以下のとおりです。

1. スタティックな割り当て
   最も優先度が低い。スタティックな割り当ては、Naturalパラメータファイル NATPARM.で指定されたパラメータによって行われます。
2. ダイナミックな割り当て
   ダイナミックな割り当ては、Naturalの起動時に個々のパラメータファイルおよび／または代替パラメータファイルを指定します。
3. ランタイム時の割り当て
   最も優先度が高い。ランタイム時の割り当ては、セッションパラメータを指定することによって、セッション中に行われます。

割り当てのタイプに関する詳細について、このセッションの以降を参照してください。

注意: Natural Security がアクティブな場合は、特定のパラメータの使用が制限されることがあります。

次の図は、パラメータ階層を示しています。



パラメータ値のスタティックな割り当て

デフォルトでは、ユーザーの Natural 環境の特質を決めるのに、パラメータファイル NATPARM 内のパラメータ指定が使用されます。初期状態のこのファイルには、Software AG によって提供されるデフォルト値が含まれます。コンフィグレーションユーティリティを使用して変更できます。

ヒント: デフォルトのパラメータファイル NATPARM は変更しないことをお勧めします。デフォルト値以外のパラメータ値で Natural を使用する場合は、独自のパラメータファイルを作成します（次のセクションも参照）。

パラメータ値のダイナミックな割り当て

ダイナミックパラメータを使用すると、Natural を開始するときに独自の環境を設定できます。セッションが開始されると、オペレーティングシステムはダイナミックパラメータの値を Natural に渡します。

現在の Natural セッションに有効なダイナミックパラメータ。これらは、デフォルトのパラメータファイル NATPARM で指定されたスタティックな割り当てを上書きします。

コンフィグレーションユーティリティを使用して、独自のパラメータファイルを作成することもできます。独自のパラメータファイルのいずれかを使用するには、Natural を開始するときにその名前を指定する必要があります。

手順 2.1. ダイナミックパラメータ値を使用して Natural を開始するには

- Natural の開始に使用されるコマンドにダイナミックパラメータとその値を追加します。

  例：プロファイルパラメータ PARM は、代替パラメータファイル MYPARM を使用して Natural を呼び出すために使用されます。プロファイルパラメータ SM および DTFORM の値は、MYPARM で定義された値の代わりに使用されます。

```
natural PARM=MYPARM SM=ON DTFORM=I
```

特殊文字

カッコやアスタリスクなどの特殊文字は、オペレーティングシステムによって解釈されます。したがって、これらの特殊文字を使用するパラメータは二重引用符で囲む必要があります。例：

```
natural "FNAT=(99,30) FUSER=(99,32)"
```

このルールの例外として、パラメータ FNAT、FDIC、FSEC、FDDM および FUSER は、引用符の使用を避けるためにカッコなしで指定することもできます。例：

```
natural FNAT=99,30 FUSER=99,32
```

指定した左カッコのそれぞれに対して、対応する右カッコも指定する必要があります。エスケープシーケンスは、ダイナミックパラメータではサポートされていません。

パラメータ値のランタイム時の割り当て

ランタイム時の割り当ては、セッションパラメータを設定することにより、セッション中に行われます。このセッションパラメータの値は、スタティックおよびダイナミックな割り当てを上書きします。

セッションパラメータは、システムコマンド GLOBALS で設定します。例：

```
GLOBALS SA=ON,IM=D
```

セッションパラメータは、プログラムの SET GLOBALS ステートメントで指定することもできます。例：

```
SET GLOBALS SA=ON IM=D
```

注意: セッションパラメータは、セッションレベルで設定できるほか（上記を参照）、プログラム、ステートメント、またはフィールドレベルで設定することもできます。詳細については、『パラメータリファレンス』の「セッションパラメータについて」を参照してください。

3 システムファイル

Natural for UNIX では、オペレーティングシステム機能からアクセスできるファイルにオブジェクトを保存します。オブジェクトが Adabas システムファイルに保存されている Natural for Mainframes とは異なり、Natural for UNIX では、ディスク上の特定のディレクトリにオブジェクトを保存します。そのため、Adabas などのデータベースは、Natural for UNIX でNatural を実行する必要はありません。

システムファイル構造

デフォルトでは、Natural ライブラリは、特定の Natural バージョンの Natural ルートディレクトリの下にサブディレクトリとして作成されます。サブディレクトリの名前はライブラリと同じです。

Natural オブジェクトは、サブディレクトリにファイルとして保存されます。Natural オブジェクトのファイル名には、次の形式があります。

```
file-name.NKT
```

file-name	これはオブジェクトの名前です。『*Natural の使用*』の「*オブジェクトの命名規則*」も参照してください。
N	拡張子の最初の文字は常に "N" です。これは 「Natural」 を表します。
K	拡張子の 2 番目の文字は、次のいずれかです。 S: ソースファイルの場合 G: 生成プログラムの場合 R: リソースの場合
T	拡張子の 3 番目の文字は、オブジェクトのタイプを表します。有効な値の詳細については、次のリストを参照してください。

例えば、ソースプログラム TESTPROG はファイル *TESTPROG.NSP* として保存され、マップ TESTMAP に対して生成されるコードはファイル *TESTMAP.NGM* として保存されます。

注意: ファイル名はオブジェクト名とは異なる場合があります。現在のオブジェクト名と対応する内部オブジェクト名の両方がファイル ***FILEDIR.SAG*** に記述されます。

可能な拡張子では、次のオブジェクトタイプに対してそれぞれの文字および数字が使用されます。

文字または数字	オブジェクトタイプ
A	パラメータデータエリア（PDA）
C	コピーコード
D	DDM
G	グローバルデータエリア（GDA）
H	Helproutine
L	ローカルデータエリア（LDA）
M	マップ
N	Subprogram
P	プログラム
S	サブルーチン
T	テキスト
4	クラス
5	コマンドプロセッサ
7	関数
8	アダプタ

アクセス権

デフォルトでは、Natural によって割り当てられたオブジェクトはアクセス権 "rw-rw-rw-".を取得します。これにより、オーナー（他のユーザー）のグループに属していないユーザーが、Natural プログラムを再カタログできるようにします。これが望ましくない場合は、UNIX ユーティリティ `umask` を実行し、適切なマスクを設定する必要があります。

Naturalアプリケーションを開発するユーザーは、アプリケーションに属するすべてのオブジェクトに対し、読み取りおよび書き込みアクセス権を保持する必要があります。通常の実稼働環境では、書き込みアクセス権がメンテナンスチームに制限される場合があります。アクセス権が低すぎるためにオブジェクトにアクセスできない場合、Naturalはオブジェクトが見つからなかったときと同様に動作します。

システムファイル FNAT および FUSER

Natural システムファイル `FNAT`（システムプログラム用）と `FUSER`（ユーザー作成プログラム用）は、異なるサブディレクトリにあります。

`FNAT` は以下のディレクトリ構造を想定しています。

```
FNAT
 └LIBDIR.SAG
 └SYSTEM
      └FILEDIR.SAG
      └SRC
      └GP
      └ERR
      └RES
 └SYS*
      └FILEDIR.SAG
      └SRC
      └GP
      └ERR
      └RES
```

ファイル *LIBDIR.SAG* は、FNAT でのみ利用可能です。Natural を使用してインストールされたすべての Software AG 製品に関する情報が含まれています。この情報は、システムコマンド SYSPROD を使用して表示できます。

FUSER は以下のディレクトリ構造を想定しています。

```
FUSER
 └SYSTEM
      └FILEDIR.SAG
      └SRC
      └GP
      └ERR
      └RES
 └user-library1
      └FILEDIR.SAG
      └SRC
      └GP
      └ERR
      └RES
```

ユーザーライブラリの名前を "SYS" という文字で始めることはできません。

ディレクトリ構造は、Natural のインストール中に生成されます。システムおよびユーザーライブラリを表すディレクトリには、次のものが含まれています。

- ***FILEDIR.SAG***
  このファイルには、Natural によって使用される内部ライブラリ情報が含まれています。詳細については、次の「*ファイル FILEDIR.SAG*」を参照してください。

- ***SRC***
  このサブディレクトリには、ライブラリに保存されているNaturalソースオブジェクトが含まれています。
- ***GP***
  このサブディレクトリには、ライブラリに保存されている生成されたNaturalプログラムが含まれます。
- ***ERR***
  このサブディレクトリには、ライブラリに保存されているエラーメッセージが含まれています。
- ***RES***
  このサブディレクトリには、ライブラリに保存されているプライベートリソースおよび共有リソースが含まれています。

DDMはローカルライブラリに保存できます。プログラムでDDMが使用されている場合、Naturalは最初に現在のライブラリを検索し、次に steplib を検索してから、ライブラリ SYSTEM を検索します。DDMが見つからない場合、プログラムはコンパイルされず、エラーメッセージが表示されます。ただし、**FDDM** モードが有効になっている場合は、Natural はシステムファイル FDDM.でのみ DDM を検索します。

システムファイル FNAT、FUSER および FDDM へのパスは、コンフィグレーションユーティリティで定義します。システムファイルはバージョンに依存します。したがって、Natural は現行のNaturalバージョンのシステムファイルにのみアクセスできます。FNAT システムファイルは1つのみ持つことをお勧めします。ただし、複数の FUSER システムファイルを定義することは可能です（例えば、目的ごとに異なる開発領域がある場合）。

システムファイル FDDM

システムファイル FDDM は、すべての DDM を保存できるコンテナです。

FDDM は以下のディレクトリ構造を想定しています。

```
FDDM
└SYSTEM
      └FILEDIR.SAG
      └SRC
      └GP
```

デフォルトでは、システムファイル FDDM はアクティブではありません。このモードを使用するには、次の説明に従って FDDM モードをアクティブにする必要があります。

- FDDM モードをアクティブ化します。
- DDM のシステムファイル FDDM への移行

- システムファイル FDDM が使用されているかどうかをチェックする

FDDM モードをアクティブ化します。

FDDM モードがアクティブの場合（グローバルコンフィグレーションファイルでデータベース ID とファイル番号の両方が 0 に一致しない場合）、すべての DDM が保存され、システムファイル FDDM に読み込まれます。ライブラリに格納された DDM は、Natural からアクセスできません。これは、すべての DDM がシステムファイル FDIC に保存されるメインフレームに類似しています。

グローバルコンフィグレーションファイルで FDDM システムファイルが未定義の場合、DDM は Natural ライブラリ FUSER および FNAT に保存されます。

手順 3.1. FDDM モードをアクティブ化するには

1 FDDM モードで DDM を保存する空のディレクトリを作成します。ディレクトリには、Natural の命名規則に対応する任意の名前を付けることができます。

2 コンフィグレーションユーティリティを呼び出します。

3 グローバルコンフィグレーションファイル（カテゴリ **System Files**）で、システムファイル FDDM のデータベース ID とファイル番号を割り当て、最初のステップで作成したディレクトリへのパスを定義します。

4 必要なパラメータファイルを選択します。

5 パラメータ FDDM を見つけます。

> **ヒント:** "FDDM" を検索して、このパラメータを見つけます。詳細については、『*コンフィグレーションユーティリティ*』ドキュメントの「*パラメータの検索*」を参照してください。

6 パラメータ FDDM には、グローバルコンフィグレーションファイルで定義したものと同じデータベース ID とファイル番号を指定します。

7 変更を保存します。

8 以下の説明に従って、必要な DDM をすべてシステムファイル FDDM に移行します。

DDM のシステムファイル FDDM への移行

FDDM モードで使用できるすべての DDM は、システムファイル FDDM に含まれている必要があります。特に、Natural に付属するライブラリ SYSEXDDM のサンプル DDM は、システムファイル FDDM で使用できることが必要です。

DDM を FDDM システムファイルに移行する場合は、以下のようなさまざまな方法を選択できます。

- オブジェクトハンドラを使用できます。これは、FDDM システムファイルをサポートし、DDM を FDDM システムファイルに移行可能にします。DDM は Natural ライブラリからアンロードでき、アクティブな Natural セッションの FDDM システムファイルに保存できます。

  重要: 開発環境を完全に移行するには、オブジェクトハンドラを使用することをお勧めします。

- また、SYSMAIN ユーティリティのコピー機能または移動機能を使用して DDM を移行することもできます。この場合、古い環境を再度使用するために、最初に FDDM パラメータを非アクティブ化する必要があります。

これらの代替策についての詳細は次に説明します。

注意: INPL ユーティリティは、FDDM モードがアクティブでない場合は Natural ライブラリに、または FDDM モードがアクティブな場合はシステムファイル FDDM に、DDM をロードします。ロードされた INPL ファイルが両方のモードで動作するように設計されている場合、このことで多少影響を受ける可能性があります。このためには、DDM が Natural ライブラリおよび FDDM システムファイルで使用できることが必要です。

手順 3.2. オブジェクトハンドラを使用して DDM をシステムファイル FDDM に移行するには

1 上述の説明に従って、FDDM モードをアクティブ化します。

2 修正されたパラメータファイル（パラメータ FDDM のパスが定義されているパラメータファイル）を使用して、Natural を起動します。

3 ダイレクトコマンド SYSOBJH を発行して、オブジェクトハンドラを呼び出します。

次の手順は、オブジェクトハンドラウィザードを使用することを前提としています。

4 メインメニューで、**Unload** 機能をマークして ENTER キーを押します。

5 結果の画面で、オプション［**Unload objects into Natural work file(s)**］をマークして ENTER キーを押します。

6 結果の画面で、オプション［**Set additional options**］をマークして ENTER キーを押します。

7 結果の画面で、オプション［**Use FDDM file for processing DDMs**］を非アクティブ化し、ENTER キーを押して前の画面に戻ります。

これにより、（移行されるDDMが含まれる）古い環境がアクティブになります。このオプションを非アクティブ化しないと、移行する DDM にアクセスできません。

8 ENTER キーを繰り返し押して、アンロードするオブジェクトタイプを選択する画面を表示します。

オプション［**Natural library objects only**］は、デフォルトで選択されています。このオプションは次の手順で必要です。

9 ENTER キーを押します。

10 表示される画面で、フィールド［**Library**］と［**Object name**］にアスタリスク（*）を入力します。さらに、フィールド［**More detailed specification of objects**］にマークを付けます。ENTER キーを押します。

11 結果の画面で、オプション［**Error messages**］および［**Shared resources**］を非アクティブ化します。［**Natural types**］フィールドで「"V"」と入力し、ENTER キーを押します。

12 ENTER キーを押して、処理するコマンドを表示します。

13 ENTER キーを押して、アンロード機能を開始します。

14 オブジェクトがアンロードされたら、メインメニューに戻ります。

15 メインメニューで、**Load** 機能をマークして ENTER キーを押します。

16 結果の画面で、オプション［**Load objects from Natural work file(s)**］をマークして ENTER キーを押します。

17 結果の画面で、オプション［**Set additional options**］をマークして ENTER キーを押します。

18 結果の画面で、オプション［ **Use FDDM file for processing DDMs**］をアクティブ化します。

これにより、FDDM システムファイルが含まれる新しい環境がアクティブになります。

注意: DDM は同じ名前で異なるライブラリに存在することができます。FDDM システムファイルのDDMが上書きされないようにして同じ名前のDDMを検出するには、［**Do not replace**］オプションを使用して DDM をロードすることをお勧めします。このオプションは、オプション［**Use FDDM file for processing DDMs**］と同じページにあります。

19 ENTER キーを押して前の画面に戻ります。

20 ENTER キーを繰り返し押して、ロードするオブジェクトタイプを選択する画面を表示します。

オプション［**Load all option from the work file**］は、デフォルトで選択されています。このオプションは次の手順で必要です。

21 ENTER キーを押します。

これで、処理するコマンドが表示されます。

22 ENTER キーを押してオブジェクトをロードします。

システムファイル FDDM が使用されているかどうかをチェックする

すべての DDM をシステムファイル FDDM に移行したら、FDDM が使用されているかどうかを確認できます。

手順 3.3. FDDM が使用されているかどうかを確認するには

1 Natural を開始します。

2 システムコマンド SYSPROF を発行します。

3 FDDM ファイルが表示されている場合、Natural はこのシステムファイルに保存されている DDM のみにアクセスします。

FDDM ファイルが表示されない場合、または予期されるファイルが表示されない場合は、セッションに使用されるパラメータファイルを改訂します。

重要な情報および警告

Natural開発者は、すべてのオブジェクトに対して読み取り、書き込み、および削除を行う権限を持っている必要があります。

エンドユーザーは、生成されたプログラムの読み取り権限のみを必要とします（場合によっては、ソースの読み取り権限も必要です）。

オペレーティングシステムユーティリティを使用してNaturalファイルにアクセスしないでください。これらのユーティリティは、Naturalディレクトリ情報を変更したり、破壊したりする可能性があります。

コードページの競合が発生することがあるため、外部のエディタは使用しないことをお勧めします。競合が発生すると、必ずではありませんが、ソースコードが破損するおそれがあります。

プライベートデータファイルはディレクトリ FUSER、FNAT および FDDM に保存しないでください。Natural により予期しない方法で削除または変更される可能性があります。

UNIX アプリケーションの作業ディレクトリとして、ディレクトリ FNAT、FUSER および FDDM のいずれも使用しないでください。Naturalシステムコマンドの発行時に問題を引き起こす可能性があります。

Natural からアクセスするオブジェクトのファイル名（8.3 フォーマットのファイル名を含むパス）は、255 バイトを超えてはなりません。

ファイル FILEDIR.SAG

ファイル *FILEDIR.SAG* は最大 60000 個のオブジェクトをサポートします。オブジェクトのプログラミングモード（ストラクチャードまたはレポーティング）や内部的に変換されたオブジェクト名など、Naturalによって使用される内部ライブラリ情報が含まれています。これらの内部オブジェクト名は、Naturalオブジェクトを保存する際に、以下の名前でディスクに自動的に作成されます。

- 8 文字を超える名前（DDM の場合もあります）
- Naturalではサポートされているものの、オペレーティングシステムではサポートされていない特殊文字を含む名前。

内部オブジェクト名は一意であり、現在のオブジェクト名の省略形と任意の数値で構成されます。現在のオブジェクト名と対応する内部オブジェクト名の両方が *FILEDIR.SAG* に記載されます。

オブジェクトが正しいディレクトリにある場合でも、このライブラリ情報が *FILEDIR.SAG* に含まれた後でのみ Natural で使用できます。Natural 内で作成されたオブジェクトの場合、ライブラリ情報は自動的に組み込まれます。その他のオブジェクトでは、`SYSMAIN` ユーティリティの **Import** 機能を使用する必要があります。

ユーティリティ `FTOUCH` は、Natural の外部で *FILEDIR.SAG* の更新に使用できます。

ポータブルな Natural システムファイル

Natural ライブラリ内のディレクトリファイル *FILEDIR.SAG* および Natural エラーメッセージファイルは、プラットフォームに依存しないポータブル名フォーマットで作成されます。例えば、異なる Windows、UNIX、OpenVMSの各プラットフォーム間で、オペレーティングシステムのコマンドを使用してライブラリをコピーするだけで `FUSER` ライブラリを交換できます。

`FNAT` システムファイルは、Naturalのインストールに付属し、バージョン固有かつプラットフォーム固有です。したがって、異なるプラットフォーム間で `FNAT` システムファイルを共有することはお勧めしません。特に、Windows プラットフォーム上の `FNAT` システムファイルには、一部の UNIX または OpenVMS プラットフォームの `FNAT` システムファイルとまったく異なる一連のユーティリティが含まれています。

異なるプラットフォーム間で `FUSER` システムファイルを共有することが可能になりましたが、実行する場合は慎重に処理する必要があります。これは、Naturalのロックメカニズムはマシン境界を越えるものではなく、したがって、異なるプラットフォーム上の2つの Natural セッションで、予期しない結果を伴って同じオブジェクトを同時に修正する可能性があるためです。

以下では次のトピックについて説明します。

言語依存オブジェクト

移植対象のアプリケーションがシステム変数 *LANGUAGE を使用する場合は、次の情報に注意する必要があります。

ほとんどすべての Natural オブジェクトは、大文字のみを含む名前でシステムファイルに保存されます。例外は、言語依存オブジェクト（特定の言語用に作成されたオブジェクト）です。言語依存オブジェクトの名前には、小文字を含めることができます。Windows は大文字と小文字が厳格には区別されない（case-preserving）オペレーティングシステムであるため（一方 UNIX は大文字と小文字が区別される（case-sensitive）オペレーティングシステム）、UNIX で作成された名前が Windows で競合を起こしたり、UNIX で開発されたアプリケーションが Windows で予期しない結果を招くことがあります。

注意: OpenVMS は Windows と同様に動作します。大文字と小文字は区別されません。ただし、ファイル名は常に大文字で作成されます。

例

コマンド SAVE PGM& は、オブジェクト名に言語識別子を含むオブジェクトを作成します。結果のオブジェクト名は、次の *LANGUAGE の設定によって異なります。

*LANGUAGE の設定	次の名前のオブジェクトが作成されます
33	PGMX（大文字の X を使用）
59	PGMx（小文字の x を使用）

UNIX（*PGMX.NGP* および *PGMx.NGP*）で作成された異なるオブジェクトは、ファイル *FILEDIR.SAG* 内で名前 PGMx.および PGMX のエントリを取得します。これら 2 つのオブジェクトには、Natural が実行される環境に応じて異なる処理が行われます。

- Natural for UNIX で PGMX を実行する場合、ファイル *PGMX.NGP* はバッファプールにロードされ、実行されます。
- Natural for Windows で PGMX を実行する場合、ファイル *PGMX.NGP* または *PGMx.NGP* のいずれかがバッファプールにロードされ、実行されます。これは、Windows がこれらの 2 つのオブジェクトを区別しておらず、1つの同じオブジェクトとして扱うためです。したがって、FUSER またはそのような FUSER のコピーを共有するアプリケーションが異なって動作する可能性があります。

非ポータブルメッセージファイルを 64 ビットプラットフォームに移行する

64 ビットプラットフォームで作成されていない、非ポータブルフォーマットの古いメッセージファイルは読み取れません。

アプリケーションを32ビットプラットフォームから64ビットプラットフォームに移行する場合は、最初に古いメッセージファイルをポータブルフォーマットに変換する必要があります。これを行うには、SYSERRユーティリティのエクスポート機能およびインポート機能を使用します。まず、メッセージファイルをテキストファイルにエクスポートし、次にテキストファイルをNaturalにインポートして新しいメッセージファイルを生成します。これにより、Windows、UNIX、OpenVMSで読み取り可能なポータブルメッセージファイルが作成されます。エクスポート機能およびインポート機能の詳細については、『*ツールおよびユーティリティ*』ドキュメントの「*メッセージファイルとテキストファイルの生成*」を参照してください。

セマフォを使用したシステムファイルへのアクセスの同期

セマフォは、Naturalシステムファイルへのアクセスを同期するために使用されます。これには追加のオペレーティングシステムリソースが必要なため、カーネルパラメータ `SEMMNI` および `SEMMNS` を、アクセスされるシステムファイル数に合わせて増やすことを検討する必要があります。

セマフォを使用すると、複数のユーザーがシステムファイル `FNAT` および `FUSER` を扱う権限を持つことができます。セマフォ ID は、ロックファイル（**.LCK*）とともに保存されます。さらに Natural セッションが開始されると、バッファプールはセマフォ ID と対応するロックファイルを探して同期します。

ロックファイルが存在しない場合は、新しいセマフォ ID とロックファイルが生成されます。これは、同期が不可能であることを意味します。

注意: 1 つのリソースのみを削除することはできません。セマフォ ID と対応するロックファイルを必ず削除する必要があります。

NFS を使用した Natural ライブラリの保存

Natural ライブラリを保存するために NFS（Network File System）を使用する場合、Natural ライブラリを保存するディレクトリをネットワークのファイルサーバーから NFS を経由してマウントすると、障害が発生することがあります。

これは、Natural オブジェクトの更新中に各ライブラリに保存された *FILEDIR.SAG* ファイルをロックする必要があるためです。

NFS のロックが、関与しているプラットフォーム間で不適合があったり、正しく設定されていないと、Naturalは要求が処理されるまでに終了できない状態でハングすることがあります。この要求は、通常、関連するシステムのコンソールに表示されるか、システムに依存する別のログファイルに記録されます。

ニュークリアスがハングしたり、終了できない障害を回避するには、Naturalライブラリをローカルディスクにだけ保存します。

4 ワークファイル

ワークファイルとはデータを書き込むことができるファイルで、ワークファイルからデータをNaturalプログラムで読み取ることができます。これらは、データの中間ストレージおよびプログラム間のデータ交換に使用されます。Natural ステートメント READ WORK FILE および WRITE WORK FILE、または UPLOAD PC FILE および DOWNLOAD PC FILE を使用して、ワークファイルとの間でデータを転送できます。

ワークファイルの定義

コンフィグレーションユーティリティまたはDEFINE WORK FILE ステートメントを使用して、最大 32 個のワークファイルの名前（パスを含む）を割り当てることができます。

使用できるワークファイルの最大数は、パラメータ WORK の設定によって異なります。

名前とパスが割り当てられていないワークファイルを使用するプログラムを実行すると、Naturalが自動的にファイル名を作成し、ワークファイルをローカルコンフィグレーションファイルで指定された一時ディレクトリに書き込みます。このようなファイルの名前は、指定されたワークファイル番号と、オペレーティングシステムによって割り当てられた任意の番号で構成されます。ワークファイル名の生成は、固有の名前の生成を試みるアルゴリズムに基づいています。Naturalパラメータ TMPSORTUNIQ により、命名規則が異なる場合があります。ワークファイル名がNaturalの外部から参照されている場合は、ファイルの識別時に問題が発生しないよう、名前を明示的に指定することをお勧めします。

以下では次のトピックについて説明します。

- コンフィグレーションユーティリティでのワークファイル名の定義
- 環境変数でのワークファイル名の定義
- アプリケーションプログラミングインターフェイスを使用したワークファイル名の定義

コンフィグレーションユーティリティでのワークファイル名の定義

コンフィグレーションユーティリティでは、パラメータファイルの **Work Files** カテゴリにワークファイル名が割り当てられます。上記のパラメータ WORK および TMPSORTUNIQ は、このカテゴリにも含まれます。詳細については、『*コンフィグレーションユーティリティ*』ドキュメントの「*ワークファイルの割り当て*」を参照してください。

ヒント: "Work Files" を検索し、ワークファイルの割り当てを見つけます。詳細については、『*コンフィグレーションユーティリティ*』ドキュメントの「*パラメータの検索*」を参照してください。

環境変数でのワークファイル名の定義

以下では次のトピックについて説明します。

- 全般的な情報
- 環境変数のデリミタ

全般的な情報

ワークファイルは UNIX 環境変数を使用して定義することもできます。パラメータファイルでワークファイル名を定義した後は、パラメータファイルを変更せずにワークファイル名を設定できます。例えば、パラメータファイル（または DEFINE WORK FILE ステートメント）のワークファイルに次の名前を指定する場合：

```
$NATURAL/$myfile
```

および、お使いのオペレーティングシステムで以下の設定を想定した場合：

```
set NATURAL=/usr/natural
set myfile=sub/test
```

これにより、次のファイル名に展開されます。

```
usr/natural/sub/test
```

注意: シェルによってチルダ文字（~）の解釈はさまざまなため、この文字は Natural では解釈されません。

環境変数のデリミタ

環境変数の名前は特殊文字で区切られます。左側のデリミタは変数の左側にあり、右側のデリミタは右側にあります。

例えば、文字列 $TEMP は TEMP という名前の環境変数を識別します。$ はともに、左側および右側のデリミタとして使用されます。

有効なデリミタ：

デリミタのタイプ	有効なデリミタ
左側のデリミタ	$
右側のデリミタ	/ 。

アプリケーションプログラミングインターフェイスを使用したワークファイル名の定義

また、ライブラリ SYSEXT のアプリケーションプログラミングインターフェイス USR1050N を使用して、ワークファイルを定義することもできます。

ワークファイルフォーマット

ワークファイルのフォーマットは、定義されているワークファイルタイプによって異なります。さまざまなワークファイルフォーマットを使用できます。Naturalは、ファイル名とその拡張子を確認することでフォーマットを認識します。

```
file-name.extension
```

ここで、*file-name* には最大 8 文字を使用でき、*extension* には最大 3 文字を使用できます。

ワークファイルフォーマットは次のとおりです。

- バイナリフォーマット
- ASCII フォーマット
- Entire Connection フォーマット
- ポータブルフォーマット
- Unformatted フォーマット
- CSV フォーマット

『*Unicode およびコードページのサポート*』ドキュメントの「*ワークファイルと出力ファイル*」も参照してください。

バイナリフォーマット

使用可能なタイプ：SAG

Software AGに固有のこのフォーマットは、すべてのデータタイプで使用できるため、推奨されるフォーマットです。ただし、異なるエンディアンモードのプラットフォーム間で移行することはできません。

書き込まれる各レコードの先頭の2バイトには、レコードの長さが格納されます。長さ自体は、プラットフォーム固有のフォーマットで書き込まれます。

ワークファイルのバイナリフォーマットを定義するには、ピリオドと拡張子 *SAG*（*<file-name>.SAG* など）を含むファイル名を使用します。

ASCII フォーマット

使用可能なタイプ：ASCII および圧縮 ASCII。

各書き込みレコードは改行（LF）で終了するため、ASCII フォーマットは英数字データにのみ推奨されます。

ワークファイルの ASCII フォーマットを定義するには、*SAG* および *NCD* を除く任意の拡張子とピリオドを含むファイル名（*<file-name>.<ext>* など）、または、ピリオドを含み拡張子を持たないファイル名（*<file-name>* など）を入力します。

Entire Connection フォーマット

使用可能なタイプ：Entire Connection および Transfer。

ワークファイルは、2 つの異なる方法でアクセスできます。

- UNIX 上でローカルにアクセスする方法。この目的には、ワークファイルタイプの Entire Connection が使用されます。
- Entire Connection を使用したデータ転送を介してアクセスする方法。この目的には、ワークファイルタイプ Transfer が使用されます。データは Entire Connection に送信され、PC にデータが書き込まれます。

製品 Entire Connection には 2 つのファイル（実際のデータを含むデータファイルと、データファイル内のデータに関するフォーマット情報を含むフォーマットファイル）が使用されます。

Natural は、タイプ Entire Connection に対応するフォーマットファイルを自動的に生成します。フォーマットファイルはデータファイルと同じ名前ですが、拡張子が *NCF* です。拡張子 *NCF* を持つフォーマットファイルの詳細については、『Entire Connection』ドキュメントを参照してください。

タイプ Transfer を使用する場合、製品 Entire Connection によりフォーマットファイルが生成されます（オプション **Create format file** がユーザープロパティで非アクティブ化されていない場合。詳細については、『Entire Connection』ドキュメントを参照してください）。

ワークファイルの Entire Connection フォーマットを定義するには、拡張子 *NCD* とピリオドを含むファイル名（*<file-name>.NCD* など）を入力します。

Entire Connection フォーマットのワークファイルをローカルディスクに対して直接読み取り／書き込みできます。

「*拡張子 NCD のワークファイルに関する特別な考慮事項*」も参照してください。

注意:

1. `READ WORK FILE` ステートメントの `RECORD` オプションは、フォーマット Entire Connection のワークファイルの読み取りには使用できません。

2. オペランドフォーマット U（Unicode）は、ワークファイルタイプ Entire Connection および Transfer ではサポートされていません。これらのワークファイルタイプで U が使用されている場合は、ランタイム時にエラーメッセージが表示されます。

ポータブルフォーマット

使用可能なタイプ：Portable

タイプ Portable は、ワークファイルが異なるマシンに転送されるときに、ワークファイルの自動エンディアン変換を実行します。例えば、PC（リトルエンディアン）で書かれたワークファイルを RS6000 または HP マシン（ビッグエンディアン）で正しく読み取ることができます。エンディアン変換は、フィールドフォーマット I2、I4、F4、F8、および U にのみ適用されます。浮動小数点のフォーマットは IEEE とみなされます。ただし、IEEE の浮動小数点表記はハードウェアシステムによって若干異なります。原則として、これらの違いは通常はワークファイルに書き込まれない無限大表現および NaN 表現にのみ適用されます。不明な点がある場合は、ハードウェアの説明を確認してください。

ファイルは常にマシン固有の表現で書き込まれるため、変換は、表現の異なるマシンによってファイルが読み取られた場合にのみ実行されます。これにより、可能な限り高速なパフォーマンスが維持されます。

上記の変換を除き、このフォーマットで変換は行われません。

ダイナミック変数に `READ WORK FILE` ステートメントを使用すると、変数のサイズが現在のレコードの長さに変更されます。

Unformatted フォーマット

使用可能なタイプ：Unformatted

タイプ Unformatted は、1 つのダイナミック変数と 1 つのレコードのみを含むファイル全体の読み取りまたは書き込みを行います（例えば、データベースから読み取られたビデオを保存する場合）。フォーマット情報は挿入されません。すべてがそのまま書き込まれ、読み取られます。

CSV フォーマット

使用可能なタイプ：CSV（コンマ区切りの値）

注意: ワークファイルタイプ CSV を使用する場合は、`CATALOG` または `STOW` コマンドを使用してソースを再カタログする必要があります。Natural バージョン 4 の生成プログラムでは、ワークファイルタイプ CSV を使用できません。

Natural フィールドは、次のように CSV ワークファイルに保存されます。

1. 最初の手順では、内部フィールドデータが読み取り可能なフォーマットに変換されます。

- 内部のNaturalデータフォーマットB（バイナリ）、O（オブジェクトハンドル）、G（GUIハンドル）、C（属性制御）のフィールドデータは、フィールド変換なしでレコードにコピーされます。データはそのまま取得されます。
- 内部の Natural データフォーマット A（英数字）のフィールドデータは、指定されたワークファイルのコードページに変換されます（『*コンフィグレーションユーティリティ*』ドキュメントの「*ワークファイル*」を参照）。コンフィグレーションユーティリティでワークファイルのコードページが指定されていない場合は、パラメータ `CP` で定義されたデフォルトのコードページが使用されます。変換は行われません。

  内部のNaturalデータフォーマットU（Unicode）のフィールドデータは、指定されたワークファイルのコードページに変換されます（『*コンフィグレーションユーティリティ*』ドキュメントの「*ワークファイル*」を参照）。ワークファイルのコードページが指定されていない場合は、パラメータ `CP` で定義されたデフォルトのコードページに変換されます。
- 内部のNaturalフォーマットD（日付）とT（時刻）の値は、英数字出力フォーマットに変換されます。ユーザーが指定した日時フォーマットが使用されるように、`DTFORM` パラメータが評価されます。
- 数値タイプの内部フィールド値は、英数字出力フォーマットに変換されます。

2. 2 番目の手順では、読み取り可能なフォーマットのフィールドデータが CSV ワークファイルレコードにコピーされます。ワークファイルのフィールドは、指定されたセパレータ文字で区切られます。フィールドに特殊文字が含まれている場合、フィールドは二重引用符で区切られます。各書き込み記録は改行（CR/LF）で終了します。

Natural フィールド名を持つヘッダーをワークファイルに書き込むよう定義した場合（『*コンフィグレーションユーティリティ*』マニュアルの「*ワークファイルの割り当て*」を参照）、以下が適用されます。

- `WRITE WORK FILE` ステートメントでは、最初に書き込まれたレコードのフィールド名を含むヘッダー行がワークファイルの最初の行に格納されます。後続の CSV レコードに異なる数のフィールドが含まれている場合は、ヘッダー行が後続の CSV レコードに対応していない可能性があります。
- `READ WORK FILE` ステートメントでは、CSV ワークファイルの最初の行がヘッダー行とみなされます。したがって、最初の行はスキップされます（つまり、最初の行のレコードデータは返されません）。

拡張子 NCD のワークファイルに関する特別な考慮事項

拡張子 *NCD* を持つファイルが Entire Connection によって作成され、`READ WORK FILE` ステートメントを介して Natural に読み込まれる場合、セッションプロパティで Entire Connection オプション **Keep trailing blanks** をアクティブ化する必要があります。詳細については、『Entire Connection』ドキュメントを参照してください。

注意: Entire Connection を使用して NCD ファイルを作成し、オブジェクトハンドラを使用してこのファイルをロードすると、ソースコントロールレコードが見つからないことを示すエラーが表示される場合があります。これを回避するには、NCD ファイルを作成するときにオプション **Keep trailing blanks** がアクティブであることを確認してください。

次の考慮事項は、Entire Connection フォーマットのワークファイルに適用されます。

- NCD ファイルが `READ WORK FILE` ステートメントで読み込まれ、対応する NCF フォーマットファイルが利用できないか、または無効な情報を含んでいる場合、NCD ファイルは ASCII ワークファイルとみなされます。
- `APPEND` 属性を使用して NCD ファイルにデータを追加する場合、古いデータと新しいデータのレコードレイアウト（NCF フォーマットファイルに書き込まれるフィールドフォーマットおよび長さの情報）が一致している必要があります。レコードレイアウトが異なる場合は、ランタイム中にエラーが発生します。
- Entire Connection で処理できる `WRITE WORK FILE VARIABLE` のワークファイルレコードの最大サイズは 32767 バイトです。
- 拡張子 *NCD* の「古い」ワークファイルがある場合は、拡張子を変更する必要があります。
- 次の各プロファイルパラメータは、読み取りおよび書き込み操作の両方で同じ値に設定する必要があります。

  `DC`（小数点文字）
  `IA`（入力割り当て文字）
  `ID`（入力デリミタ文字）

- メインフレームコンピュータ上の浮動小数点変数に使用できる値の範囲は、他のプラットフォームとは異なることに注意してください。メインフレーム上の F4 変数と F8 変数に設定可能な値の範囲は次のとおりです。

  $\pm 5.4 * 10^{-79} \sim \pm 7.2 * 10^{75}$

  F4 変数について、その他の大半のプラットフォームで設定可能な値範囲は次のとおりです。

$\pm 1.17 * 10^{-38} \sim \pm 3.40 * 10^{38}$

F8変数について、その他の大半のプラットフォームで設定可能な値の範囲は次のとおりです。

$\pm 2.22 * 10^{-308} \sim \pm 1.79 * 10^{308}$

- Entire Connection でデータ転送中に DBMS コールが発行され、その数が画面 I/O 間で許可される DBMS コールの制限を超える場合（プロファイルパラメータ MADIO で指定）、Natural エラーメッセージが返されます。このエラーを回避するために、ライブラリ SYSEXT にアプリケーションプログラミングインターフェイス USR1068N が用意されています。USR1068 はデータベースコールのカウンタをゼロ（0）にリセットします。データ転送中にDBMSコールが発行されるたびに呼び出す必要があります。

ワークファイルタイプ Transfer の使用

ローカルアクセスを使用すると（つまり、データ転送が行われません）、Entire Connection フォーマットのワークファイルをローカルディスクに対して直接読み取り／書き込みできます。ただし、データ転送を使用して Entire Connection フォーマットのワークファイルにアクセスすることもできます。どちらの方法も同時に使用できますが、ワークファイル番号が異なる場合のみ使用できます。

データ転送（タイプ Transfer）を使用してアクセスするワークファイルは、Entire Connection フォーマット（NCD）である必要があります。

データ転送では、Natural ステートメント READ WORK FILE および WRITE WORK FILE はローカルディスクとの間で読み取りや書き込みを行わず、Entire Connection を実行する PC にデータを転送します。その後、Entire Connectionによって、PCのディスクとの間で読み取り／書き込み操作が実行されます。

ワークファイル番号を使用するには、コンフィグレーションユーティリティでプロファイルパラメータ ECPMOD を ON に設定する必要があります。この場合、ワークファイル名を割り当てる必要はありません。これは、Entire Connection がファイル名の入力を求めるためです。

5 Natural バッファプール

全般的な情報

Natural バッファプールは、同じコンピュータ上のオブジェクトにアクセスする複数の Natural プロセス間で Natural オブジェクトを共有するために使用されます。これは、コンパイルした Natural プログラムを実行に備えて配置するストレージエリアです。Natural ユーザーからの Naturalオブジェクト要求に応じて、プログラムはバッファプールにまたはバッファプールから移動されます。

Natural ではリエントラントな Natural オブジェクトコードが生成されるため、Natural プログラムの1つのコピーを複数のユーザーが同時に実行できます。この目的のため、各オブジェクトは、呼び出し元が呼び出すたびにロードされるのではなく、システムファイルから Naturalバッファプールに 1 回だけロードされます。

以下では次のトピックについて説明します。

- バッファプール内のオブジェクト
- UNIX での調整
- 複数のバッファプール
- バッファプールにオブジェクトを保存する
- 読み取り専用バッファプール
- 制限

バッファプール内のオブジェクト

バッファプール内のオブジェクトは、プログラムやマップなど、あらゆる実行可能なオブジェクトです。ローカルデータエリア、パラメータデータエリア、およびコピーコードの実行可能なオブジェクトは、コンパイル目的でのみバッファプールに配置されます。

Naturalオブジェクトがバッファプールにロードされると、ディレクトリエントリと呼ばれるコントロールブロックがそのオブジェクトに割り当てられます。このコントロールブロックには、オブジェクトの名前、オブジェクトが属するライブラリやアプリケーション、オブジェクトの取得元のデータベース ID および Natural システムファイル番号などの情報と、特定の統計情報（プログラムを同時に実行しているユーザーの数など）が含まれます。

UNIX での調整

リソース共有では、バッファプールへのアクセスをすべてのユーザー間で調整する必要があります。これを実現するには、いくつかのシステムリソースが必要です。例えば、UNIX オペレーティングシステム上の共有メモリは、オブジェクトとその管理情報を格納するために使用されます。これらのオブジェクトへのアクセスを同期するには、一連のセマフォを使用します。使用可能な共有メモリの量とセマフォの数はオペレーティングシステムでスタティックに設定されます。そのため、システムパラメータの変更やインストール用のオペレーティングシステムカーネルの再作成が必要になる場合があります。これらのトピックの詳細はシステムに依存し、お使いの UNIX コンピュータのインストールドキュメントに記載されています。

複数のバッファプール

個々の要件に応じて、同じコンピュータ上で同じNaturalバージョンの異なるバッファプールを同時に実行することができます。

バッファプールにオブジェクトを保存する

ユーザーがプログラムを実行すると、バッファプールマネージャへの呼び出しが行われます。ディレクトリエントリが検索され、プログラムがすでにバッファプールにロードされているかどうかが確認されます。まだバッファプールに存在しない場合は、適切なライブラリからコピーが取得され、バッファプールにロードされます。

Naturalオブジェクトがバッファプールにロードされると、このプログラムを識別するために新しいディレクトリエントリが定義され、また新しくロードするオブジェクトを収容するために、現在実行中ではない1つまたは複数の他のNaturalオブジェクトがバッファプールから削除されることがあります。

この目的のために、バッファプールは、現在どのユーザーがどのオブジェクトを使用しているかを記録し、すべてのオブジェクトを解放することなくユーザーがNaturalを終了する状況を検出します。他のアプリケーションに属する新しいオブジェクトに対応できるよう、使用されていないオブジェクトや古いオブジェクトをダイナミックに削除します。

読み取り専用バッファプール

読み取り専用バッファプールとは、読み取りアクセスのみを許可する特殊なバッファプールです。オブジェクトが読み取り専用のバッファプールにない場合、Naturalはエラー82（オブジェクトが見つかりません）を発行します。システムファイルで見つからないオブジェクトの取得は試行されないため、システムファイルおよびバッファプール上のすべてのロック操作はスキップされます。アカウントデータは収集されません。読み取り専用バッファプールにアクセスできるユーザー数に制限はありません。

読み取り専用バッファプールは、コンフィグレーションユーティリティで定義されます（下記の「*バッファプールの設定*」も参照）。バッファプールが読み取り専用バッファプールとして定義されている場合、ユーザーの最大数に対して定義された値は無視されます。

このユーティリティ `NATBPSRV` は、読み取り専用バッファプールにセマフォを割り当てません。ただし、これはローカルコンフィグレーションファイル（インストールの割り当て）内で定義されたNaturalパラメータファイルの場所にある *<bufferpool-name>.PRL* という名前のファイルのプリロードリストを想定しています。例えば、読み取り専用バッファプールの名前が"ROBP"である場合、ファイル名は *ROBP.PRL* である必要があります。

プリロードリストは、Natural ユーティリティ `CRTPRL` を使用して生成できます。このユーティリティはバッファプールの内容を抽出し、バッファプールの既存のプリロードデータとマージします。

PRL ファイルのプリロードリストには、コンマで区切られた次の形式のデータを持つレコードが含まれています。

```
database-ID、file-number、library、object-name、kind、type
```

ファイル内のキーワードは、NATBPMON ユーティリティの DIR コマンドで示されるキーワードと同じ意味を持ちます。

ディレクトリを記述するレコード（オブジェクトが *FILEDIR.SAG* の一部であることを示すオブジェクトの種類 D）を除き、値はすべてのキーワードに割り当てる必要があります。例：

Keywords	NATBPSRV でバッファプールに以下をロードします
222,111,MY_LIB,PGM1,G,P	データベース 222 およびファイル番号 111 にあるライブラリ MY_LIB からのプログラム PGM1のオブジェクトコード。
222,113,*,*,D	*LIBDIR.SAG* は、FNAT=222,113 に存在します。
222,111,MY_LIB,*,D	*FILEDIR.SAG* は FUSER=222,111 に存在するライブラリ MY_LIB にあります。

読み取り専用バッファプールには、アプリケーションを詳細に把握する必要があるという短所があります（存在しないオブジェクトはロードできないため）。つまり、アプリケーションが必要とするすべてのオブジェクトをプリロードリストで指定する必要があります。まれに、アプリケーションに必要な完全なオブジェクトセットを事前に決定できます。

セカンダリ読み取り／書き込みバッファプール

Naturalはプライマリバッファプールとして読み取り専用バッファプールで実行できます。このようなバッファプールは変更できません。読み取り専用バッファプールに存在しないオブジェクトはロードできません。オブジェクトが読み取り専用のバッファプールにない場合、Naturalはエラー82（オブジェクトが見つかりません）を発行します。これを回避するには、Naturalを実行時にセカンダリ標準バッファプール（読み取り／書き込みアクセス可能）に接続し、そこで不明オブジェクトをアクティブにします。プライマリバッファプール内のオブジェクトの検索コールに失敗した場合、セカンダリバッファプールはバックアップバッファプールとして動作します。ダイナミックパラメータ BPID2 はセカンダリバッファプールを識別します。

読み取り専用バッファプール以外に、セカンダリバッファプールに接続できるユーザーの最大数は決まっており、セマフォによるオブジェクトのロックは、セカンダリバッファプールにアクセスするたびに実行されます。

読み取り専用バッファプールのプリロードリストは、ユーティリティ CRTPRL を使用して、セカンダリ読み取り／書き込みバッファプールの内容を読み取り専用バッファプールのプリロードリストにマージすることにより、更新／拡張できます。

代替の読み取り専用バッファプール

読み取り専用バッファプールの場合、コンフィグレーションユーティリティで代替バッファプールの名前を定義できます（下記の「*バッファプールの設定*」も参照）。

読み取り専用バッファプールでのみ使用可能な NATBPMON ユーティリティの SWAP コマンドを使用して、読み取り専用バッファプールを 「廃止」 としてタグ付けできます。廃止バッファプールに接続された Natural セッションはすべてこのバッファプールから切断され、代替バッファプールに接続されます。ただし、代替バッファプールも読み取り専用バッファプールである場合に限ります。Natural で新しいオブジェクトをロードしようとしたり（例えば CALLNAT または RETURN ステートメントを実行したとき）、スタックに配置されたコマンドを Natural で解釈しようとすると、別のバッファプールへの切り替えが発生します。廃止としてタグ付けされたバッファプールの IPC リソース（共有メモリセグメント）は、NATBPMON ユーティリティの SWAP コマンドを発行した後で削除できます。この機能を使用すると、Naturalセッションを停止することなく、更新されたコンテンツを含む別の読み取り専用バッファプールでバッファとその内容を交換できます。

既知の問題：NATBPMON ユーティリティの IPCRM コマンドは、読み取り専用バッファプールに関連付けられたセマフォを削除しようとするエラーをレポートします。

CRTPRL ユーティリティを使用したプリロードリストの作成

ライブラリ SYSBPM にある Natural ユーティリティ CRTPRL は、読み取り専用バッファプールのプリロードリストを作成するために使用します。

このユーティリティは、プリロードリストのベースとしてソースバッファプールの内容を使用し、プリロードリストが読み取り専用（ターゲット）バッファプールにすでに存在するかどうかをチェックします。

- プリロードリストが存在する場合、プリロードリストにある既存のデータはソースバッファプールのデータとマージされ、プリロードリストは新しい内容で保存されます。
- プリロードリストが存在しない場合は、ソースバッファプールの内容を使用して作成されます。

結果のプリロードリストの内容により、読み取り専用バッファプールの内容が決定します。プリロードリストは、対応するオブジェクトを読み取り専用バッファプールにロードするユーティリティ NATBPSRV によって読み取られます。

制限

Natural バッファプールを使用する場合は、最低限の制限のみを考慮する必要があります。

- Natural セッションがハングアップした場合は、UNIX コマンド `kill -KILL`（および `kill -9`）、端末コマンド `break` または中断用キーを使用して終了しないでください。

  このセッションでバッファプール内部データ構造の変更を実行している場合、更新が完全に完了していない段階で中断する可能性があります。バッファプールの内部データ構造が矛盾している場合は、悪影響を与えることがあります。

  代わりに、UNIX コマンド `kill -TERM`（または `kill -15`）を使用してハングアップセッションを終了します。

  注意: これは、Natural ニュークリアスがバッファプールルーチンを実行している場合にのみ発生します。

- すべてのリソースは、1 つの Natural バッファプールのすべてのユーザー間で共有する必要があります。プロセスのグループメンバーシップは、バッファプールのアクセス権を付与するために使用されます。つまり、共有メモリはすべてのグループメンバーが変更できますが、他のユーザーは変更できません。同様のことがセマフォにも当てはまります。

  注意: 同じ Natural バッファプールのすべてのユーザーは、UNIX オペレーティングシステム上の同じユーザーグループに属している必要があります。

バッファプールの設定

バッファプールの割り当ては、ローカルコンフィグレーションファイルに格納されます。バッファプールを設定するには、コンフィグレーションユーティリティを使用してローカルコンフィグレーションファイルに特定の値を指定する必要があります。これらの値のリストについては、『*コンフィグレーションユーティリティ*』ドキュメントの「*バッファプール割り当て*」を参照してください。

バッファプールを作成するためのユーティリティ NATBPSRV の使用

バッファプールは、ユーティリティ `NATBPSRV` を使用して作成されます。

注意: ユーティリティ `NATBPSRV` は、他のバッファプールユーザーの作業に損傷を与える可能性があるため、アクセスできる Natural ユーザーを限定する必要があります。

NATBPSRV は、バッファプールに必要なリソースを割り当て、バッファプールに使用される恒久的な通信機能（共有メモリおよびセマフォ）を作成します。リソースと機能に必要な仕様は、コンフィグレーションユーティリティを使用して作成します（「バッファプールの設定」を参照）。

NATBPSRV ユーティリティは、起動手順 *natstart.bsh*。

デフォルトでは、バッファプール NATBP が起動します。別のバッファプールを開始する場合は、次の NATBPSRV コマンド行オプションを使用して名前を指定します。

```
NATBPSRV BP = buffer-pool-name
```

NATBPSRV により、同じ名前のバッファプールがすでにアクティブになっていることがバッファプールの作成プロセスで検出された場合、すでにアクティブなバッファプールが削除されます。削除に失敗した場合、NATBPSRV により、適切なエラーメッセージが表示され、終了します。

NATBPSRV エラーメッセージ

NATBPSRV は、作成するバッファプールが読み取り専用バッファプールであることを意図している場合は、次のエラーメッセージを発行することがあります。

Unable to attach to buffer pool. Return code ... received from bp_init.

説明	プリロードリストで説明されているオブジェクトをロードするには、ユーザーとして NATBPSRV を以前に作成されたバッファプールに関連付けます。関連付けのプロセスに失敗しました。
処置	Software AG 技術サポートに連絡してください。

Unable to get parameter path.

説明	Natural のパラメータファイルを識別するローカルコンフィグレーションファイルで定義されたパスを確立できませんでした。
処置	Software AG 技術サポートに連絡してください。

File ... is not accessible.

説明	プリロードリストにアクセスできないか、存在しません。
処置	アクセス権を改訂するか、プリロードリストを作成します。

Unable to open file ...

説明　　プリロードリストを読み取れません。

処置　　プリロードリストを再度作成します。

Skipped erroneous record: '...'. Buffer pool may not operate correctly.

説明　　プリロードリストに無効なレコードが見つかりました。レコードがスキップされ、ロードプロセスが続行されます。オブジェクトが見つからないために、アプリケーションでエラーが発生する可能性があります。

処置　　レコードが手動で作成されている場合は修正するか、Software AG 技術サポートに連絡してください。

Unable to retrieve LIBDIR.SAG in FNAT(...,...). Application will not run.

説明　　*LIBDIR.SAG* が見つかりませんでした。`FNAT(...,...)` に依存するアプリケーションは実行されません。

処置　　レコードが手動で作成されている場合は修正するか、Software AG 技術サポートに連絡してください。

Buffer pool manager returned with error code ... . Buffer pool is not operational.

説明　　*FILEDIR.SAG* バッファプールにロードすることができませんでした。バッファプールが小さすぎて *FILEDIR.SAG* を保持できないか、*FILEDIR.SAG* が破損しています。上記のメッセージは、どの *FILEDIR.SAG* が問題の原因であるかを示しています。

処置　　レコードが手動で作成されている場合は修正するか、Software AG 技術サポートに連絡してください。

Buffer pool manager returned with error code ... . Error ... occurred.

説明　　バッファプールへのオブジェクトのロード中にエラーが発生しました。

処置　　通常、バッファプールのサイズが小さすぎることが原因です。サイズを大きくして、操作を繰り返します。問題が解決しない場合は、Software AG 技術サポートに連絡してください。

Object ... in library ... on system file (...,...) not found. Application may not run.

説明　　処理されたプリロードレコードが、見つからなかったオブジェクトを参照しています。これは通常、アプリケーションが変更され、対応するプリロードリストが更新されていない場合に発生します。

処置　　問題のプリロードレコードを除外／改訂します

Preload executed. Buffer pool is ready to run.

説明　すべてのプリロードレコードが処理されました。バッファプールはロック解除されており、Natural でそのバッファプールにアクセスできます。

バッファプールのモニタリング

バッファプールモニタは、バッファプールの動作を監視するために使用されます。バッファプールモニタを使用して、コンピュータ上でNaturalを停止する必要がある場合にバッファプールをシャットダウンすることもできます。

バッファプールモニタは、Natural バッファプールの現在の状態に関する情報を収集します。

同じコンピュータ上で複数のバッファプールがアクティブになっており、複数のバッファプールにロードされているオブジェクトがNaturalプロセスによって変更された場合、そのオブジェクトは、変更する Natural プロセスが付加されるバッファプールからのみ削除されます。

バッファプールモニタの使用方法の詳細については、「*バッファプールモニタの使用（NATBPMON）*」を参照してください。

トラブルシューティング

このセクションでは、Naturalバッファプールの使用時に発生する可能性のある問題とその解決方法について説明します。

ここでは、UNIX コマンド `ipcs` および `adb` について理解していることを前提としています。

一般的なコマンド出力の例と、実行中の間違いについての説明を以下に示します。

問題 1

NaturalまたはNaturalバッファプールモニタ（NATBPMONユーティリティ）を起動できません。

例

以下の例では、Natural管理者またはユーザーが直面する可能性のある最も一般的な問題を示しています。これらの問題は Natural または Natural バッファプールモニタを起動する際、バッファプールがアクティブではない場合に発生します。

- 以下のコマンドで Natural を起動しようとします。

```
natural bp = sag
```

次のメッセージが表示されます。

```
Natural Startup Error:    16
Unable to open Buffer Pool,
Buffer Pool error:   "unexpected system call error occurred "  (20)
Global shared memory could not be attached.:   shmkey = 11111111

Operating System Error 2 - No such file or directory
```

- 次のコマンドで Natural バッファプールモニタの起動を試みます。

```
natbpmon bp = sag
```

NATBPMON プロンプトで WHO コマンドを入力すると、次のメッセージが表示されます。

```
Buffer Pool error:   unexpected system call error occurred  (20)
Global shared memory could not be attached.:   shmkey = 11111111
Operating System Error 2 - No such file or directory
```

解決策

1. 「*バッファプールを作成するためのユーティリティ **NATBPSRV** の使用*」の説明に従って、バッファプールサービスを開始します。
2. UNIX コマンド ipcs を使用して、必要なセマフォおよび共有メモリの存在を確認します。

```
ipcs -m -s
```

この出力結果は次のとおりです。

```
IPC status from /dev/kmem as of Mon 23-MAY-2005 12:03:24.30
T     ID     KEY        MODE       OWNER    GROUP
Shared Memory:
m    807 0x4e425031 --rw-rw----      sag  natural
Semaphores:
s     85 0x4e425031 --ra-ra----      sag  natural
```

注意: 上記の出力は、Natural バッファプールに属さないメモリセグメントおよびセマフォを除外するために編集されました。

割り当てたキーで共有メモリセグメントまたはセマフォセットが見当たらない場合、バッファプールは開始されませんでした。

問題 2

Natural バッファプールと Natural ユーティリティが同じ Natural バージョンではありません。

例

ユーティリティでバッファプールを使用しようとすると、ユーティリティとバッファプールのバージョンが同一であるかチェックされます。バージョンが異なる場合、アクセスは拒否され、エラーメッセージが出力されます。

- Natural を起動しようとすると、次のメッセージが表示されます。

```
Natural Startup Error 16: Unable to open buffer pool.
Buffer pool error: "Buffer pool does not correspond with your version of ↩
Natural"(25).
Internal version of buffer pool is 0 but requested internal version is 1. ↩
```

- Natural バッファプールモニタを起動しようとすると、次のメッセージが表示されます。

```
Buffer pool error: Buffer pool does not correspond with your version of Natural ↩
(25).
Internal version of buffer pool is 0 but requested internal version is 1.
```

解決策

使用している Natural バージョンがバッファプールバージョン番号に対応しており、内部バッファプールバージョン（BP バージョン）も正しいことを確認します。Natural と同じバージョンでバッファプールを再スタートしますが、他のユーザーがアクティブでないことを確認してください。

⚠ **重要:** 内部バッファプールバージョン番号（**BP** バージョン）は、サービスパックのリリース（製品バージョン番号の3 桁目）によって異なる可能性があります。例えば、Natural バージョン *vrs* で開始されたバッファプールは Natural バージョン *vr(s+1)* では使用できず、その反対も同じです。

バッファプールをシャットダウンして再起動する

通常、バッファプールをシャットダウンして再起動する必要はありません。これは、バッファプール内の重大な内部エラーによりバッファプールが使用不可能になった場合（発生することはまれです）、またはバッファプール構造を定義するパラメータが古くなったためにバッファプールが使用不可になった場合にのみ必要です。

NATBPMON ユーティリティがまだバッファプールにアクセスできる場合は、次の手順に従います。

1. NATBPMON ユーティリティの SHUTDOWN コマンドを使用してバッファプールをシャットダウンします。

   SHUTDOWN コマンドが実行されると、新しいユーザーはバッファプールへのアクセスを拒否されます。

   ヒント: アクティブなバッファプールユーザーは、NATBPMON ユーティリティの WHO コマンドおよび STATUS コマンドを発行することで監視できます。

2. 最後のユーザーがバッファプールへのアクセスを停止したら、NATBPMON ユーティリティの IPCRM コマンドを発行してバッファプールリソースを削除できます。
3. バッファプールを再起動するには、十分な権限のあるアカウントからファイル *natstart.bsh* を呼び出します。

スーパーユーザー権限がある場合は、SHUTDOWN コマンドの FORCE オプションを使用できます。

1. NATBPMON ユーティリティの SHUTDOWN FORCE *grace-period* コマンドを使用してバッファプールをシャットダウンします。

   このコマンドは、オプションを指定しない SHUTDOWN コマンドと同様に、新しいユーザーによるバッファプールへのアクセスを拒否します。ただし、終了シグナル SIGTERM はすべてのアクティブな Natural セッションに送信され、バッファプールから強制的にログオフされます。

   オプションのパラメータ *grace-period* を省略すると、このコマンドはすべてのアクティブセッションがシャットダウン処理を実行するまで待機した後、NATBPMON ユーティリティの IPCRM コマンドを実行します。

   オプションのパラメータ *grace-period* が指定されている場合、NATBPMON は、バッファプールにログオンしているセッションのクローズダウンステータスに関係なく、IPCRM コマンドを実行する前に、指定された秒数待機します。したがって、猶予時間として定義された値は、セッションが時間内に終了できるよう十分に長くする必要があります。

   注意: SHUTDOWN FORCE 0 は SHUTDOWN FORCE と同様です（パラメータ *grace-period* を指定しない）。

2. SHUTDOWN FORCE コマンドを正常に実行した後にバッファプールを再起動するには、十分な権限のあるアカウントからファイル *natstart.bsh* を呼び出します。

NATBPMON ユーティリティがバッファプールのクリーンシャットダウンを実行できない場合は、オペレーティングシステムのコマンドを使用してバッファプールを削除する必要があります:

1. UNIX コマンド ipcs を使用して、バッファプールの共有メモリおよびセマフォのステータスを検出します。

```
ipcs -a -m
```

ipcs -m -a コマンドの出力列 **NATTCH** には、現在共有メモリセグメントに接続されているプロセスの数が表示されます。例えば、次のようになります。

```
IPC status from /dev/kmem as of Mon May 23 12:15:38.39 2002
T         ID     KEY        ...  OWNER  GROUP  ... NATTCH    SEGSZ
Shared Memory:
m        707 0x4e425031  ... sag    natural ...      7   153600
```

2. 共有メモリに接続されているプロセスの数に、Natural ニュークリアスまたは現在実行中の NATBPMON ユーティリティが含まれていると思われます。これらのプロセスを実行しているユーザーに通知し、自分で終了するように依頼するか、ps コマンドを使用してプロセス ID を検出した後、UNIX コマンド kill を使用してセッションを終了します。
3. 重要な作業にバッファプールを使用しているユーザーがいないことを確認したら、UNIX コマンド ipcrm を使用してリソースを削除できます。例えば、次のようになります。

   ```
   ipcrm -M 0x4e425031 -S 0x4e425031
   ```

   -M および -S オプションに指定される値は、バッファプールの起動に使用されるパラメータファイル内で指定された値である必要があります。

   UNIX コマンド ipcrm を使用して共有メモリおよびセマフォを削除する場合は注意してください。誤って間違ったリソースを削除してしまった場合、コンピュータ上で実行されている他のソフトウェア製品に重大な影響を与える可能性があります。
4. 削除の結果は、UNIX コマンド ipcs を再度使用して確認できます。

   メモリセグメントまたはメッセージキューの一部がまだ表示されている場合は、他のソフトウェアに属しているか、他のプロセスが依然として接続しているために削除のためのマークが付けられている可能性があります。

共有メモリおよびセマフォを削除した後にバッファプールを起動できない場合は、コンピュータの再起動または Software AG サポートへの問い合わせを検討する必要があります。

6 バッファプールモニタ（NATBPMON）の使用

「*Natural バッファプール*」も参照してください。ここでは、バッファプールに関する全般的な情報と、バッファプールの起動方法について説明しています。

注意: このユーティリティを使用すると、バッファプールの他のユーザーの作業に有害な影響を与える可能性があるため、通常、このユーティリティにアクセスできるユーザーを限定する必要があります。

NATBPMON ユーティリティの呼び出し

NATBPMON ユーティリティは、デフォルトのバッファプール NATBP または別の既存のバッファプールのいずれかに対して呼び出すことができます。

手順 6.1. NATBPMON ユーティリティを呼び出すには

- デフォルトのバッファプール NATBP を使用する場合は、コマンド行に次のコマンドを入力します。

```
NATBPMON
```

または:

別のバッファプールを使用する場合は、コマンド行に次のコマンドを入力します。

```
NATBPMON BP=buffer-pool-name
```

次のプロンプトが表示されます。

```
NATBPMON>
```

NATBPMON コマンド

NATBPMON プロンプトでは、次のコマンドを入力できます。

Command	説明
CLEAR	これは ZERO コマンドと同様です。
CORPSES	コープスのリストを表示します。コープスとは、削除されたものの、削除が行われた時点でバッファプールで使用されていたオブジェクトです。このオブジェクトは使用されなくなると、自動的にコープスのリストから消えます。 **注意:** DIR コマンドとともに表示される列 **cusr** は、オブジェクトが使用されているかどうかを示します。

Command	説明
DELETE {*pattern*|[*]}	オブジェクトをバッファプールから削除します。アスタリスク（*）を使用することで、すべてのオブジェクトをバッファプールから削除できます。パターンを使用してオブジェクトのコレクションを指定します。これは、ワイルドカードを含むファイルのクラスを指定できる最新のオペレーティングシステムと同様です。詳細については、「*パターンの指定*」を参照してください。
DIR {*pattern*|[*]}	バッファプール内のすべてのオブジェクトを含むディレクトリを表示します。詳細については、セクション「*パターンの指定*」および「*バッファプール内のオブジェクトの表示*」を参照してください。
DUMP	エラー解析に使用します。 **重要:** Software AG サポートから要求された場合を除き、このコマンドを使用しないでください。
EXIT	NATBPMON ユーティリティを終了します。
FIN	NATBPMON ユーティリティを終了します。これは EXIT コマンドと同様です。
HELP	NATBPMON ユーティリティで使用可能なすべてのコマンドのリストを表示します。
IPCRM	バッファプールに割り当てられたリソースを解放します。このコマンドは、アクティブなユーザーがいない場合に、SHUTDOWN コマンドの後にのみ使用する必要があります。
KILL *n*	指定したバッファプールユーザーを強制終了します。*n*は「強制終了」するユーザーの名前です。この番号は、WHO コマンドで表示されるインデックス番号に対応します。
PARAM	バッファプール設定を表示します。詳細については、「*バッファプールの設定の表示*」を参照してください。
QUIT	NATBPMON ユーティリティを終了します。これは EXIT コマンドと同様です。
SHUTDOWN [FORCE [*grace-period*]]	オプション FORCE を指定しない：バッファプールをシャットダウンします。このコマンドが発行されると、新しいプロセスでバッファプールを使用できなくなります。NATBPMON ユーティリティは、シャットダウンステータスが「保留中」のバッファプールで実行できます。この場合、NATBPMON ユーティリティのすべてのコマンドを使用できます。すべてのユーザーがバッファプールの使用を停止したら、IPCRM コマンドを使用してバッファプールのリソースを削除できます。 このオプション FORCE では、スーパーユーザー権限で NATBPMON を実行する必要があります。SUDO または SU でパスワードが確認され、NATBPMON への制御が与えられると、新しいセッションはすべてログインできなくなり、終了シグナル SIGTERM がすべてのアクティブな Natural セッションに送信されます。続いて、NATBPMON は、バッファプールによって使用される IPC リソースがシステムから削除されるまで、パラメータ *grace-period* で定義された秒数待機します。オプションのパラメータ *grace-period* が省略されている（または 0 に設定されている）場合、NATBPMON はすべてのプロセスがクリーンアップ処理を実行するまで待機します。このプロセスは緊急停止とみなされる場合があります。スーパーユーザー権限なしで実行された場合、アクションは実行されず、コマンドを実行する権限がないことを示すメッセージが送信されます。「*バッファプールをシャットダウンして再起動する*」も参照してください。

Command	説明
	注意: シャットダウン後にバッファプールを起動するには、NATBPSRV ユーティリティを使用します。
STATUS	バッファプールに関する統計情報が表示されます。詳細については、「*バッファプールに関する統計情報*」を参照してください。
SWAP	読み取り専用バッファプールでのみ使用できます。読み取り専用バッファプールを「廃止」としてタグ付けします。このようなバッファプールに接続されたすべての Natural セッションは、そのバッファプールから切断され、代替のバッファプールに接続されます。
WHO	バッファプールを使用しているすべてのユーザーのリストを表示します。NATBPMON ユーティリティが各バッファプールユーザーに自動的に割り当てる番号（インデックス）およびユーザー ID、端末 ID、バッファプール（tid）を使用したプロセスのプロセス ID の統計が表示されます。
WRITE	ディスクにバッファプールオブジェクトを書き込みます。インデックスとファイル名を指定するよう求められます。 **注意:** DIR コマンドで表示される列「indx」には、インデックス番号が表示されます。
ZERO	STATUS コマンドで表示されるすべてのカウンタを 0 にリセットします。

バッファプール内のオブジェクトを表示する

DIR コマンドはオブジェクトのリストを表示します。このリストには次の情報が表示されます。

列	説明	
indx	NATBPMON ユーティリティがバッファプールにロードされたときにオブジェクトに自動的に割り当てられる番号。	
cusr	バッファプール内のオブジェクトを使用している現在のユーザー数。	
pusr	バッファプール内で同時にアクティブ化されたオブジェクトの最大数。	
nusg	バッファプールでオブジェクトがアクティブ化された回数。	
g	システムファイルからバッファプールにオブジェクトをロードしているかどうかを示します。次のいずれかの値になります。	
	0	オブジェクトがロードされていません。
	1	オブジェクトがロードされています。
size	バッファプール内のオブジェクトのサイズ（バイト単位）を指定します。	
gpv	生成プログラムのバージョン番号です。	
key	オブジェクトに関する次の情報を示します。	
	D	データベース ID。
	F	ファイル番号。

列	説明	
	L	オブジェクトが存在するライブラリです。
	N	オブジェクトの名前です。数字とアットマーク（@）は、現在ロードされているライブラリの *FILEDIR.SAG* を示します。
	K	オブジェクトの種類（G = 生成されたオブジェクトモジュール、S = ソース、D = *FILEDIR.SAG* の一部）。
	T	オブジェクトタイプ（K フィールドに D が表示されている場合は空白）。

DIR コマンドが発行されると、プール内のすべてのオブジェクトは、次のような表記で表示されます。

```
indx:    index of the element
 cusr:    current number of concurrent users
 pusr:    peak number of concurrent users
 nusg:    number of usages
 g   :    set if object is generating
 gpv :    version of generated program

indx | cusr | pusr |  nusg  |  g  |  size  |  gpv  |  key
-----+------+------+--------+-----+--------+-------+-------------------------------------------------
 1   |  0   |  1   |    4   |  0  |   920  |       |  (D=99 F=101 L="DEMO" N="SEL-MAP" K='G' T='M')
 2   |  1   |  7   |    2   |  0  |  3096  |       |  (D=99 F=101 L="DEMO" N="EMWND" K='G' T='P')
 3   |  4   |  9   |    4   |  0  |   604  |       |  (D=99 F=101 L="DEMO" N="HDR" K='G' T='P')
 4   |  2   |  3   |    7   |  0  |   412  |       |  (D=99 F=101 L="RPA" N="MMUPROG1" K='G' T='P')
 5   |  0   |  1   |    5   |  0  |   372  |       |  (D=99 F=101 L="RPA" N="MMUPROG2" K='G' T='P')
 6   |  0   |  5   |    4   |  0  |   372  |       |  (D=99 F=101 L="RPA" N="MMUPROG3" K='G' T='P')
```

パターンの指定

パターンは、コマンド DIR および DELETE を使用して指定できます。このセクションの例は、DIR コマンドに該当します。

一部のオブジェクトを選択するには、一致するパターン表現を指定して、特定のキーフィールドの値を制限できます。

指定したフィールドの許容フィールド値を制限するには、次のパターン式を使用する必要があります。

```
name=expression
```

コンマで区切って複数のパターンを指定できます。

キー全体を受け入れるには、指定されたパターンがすべて対応するフィールドと一致する必要があります。

式にはワイルドカード文字 "*" および "?" を使用できます。

文字 "*" は一連の文字の有無に関係なく一致し、ワイルドカード文字 "?" はただ 1 つの個別の文字と一致します。

例

上記のサンプルでタイプ P のすべてのオブジェクトを選択するには、次のコマンドを使用します。

```
DIR T=P
```

デモライブラリ内のすべてのプログラムを選択するには、次のコマンドを使用します。

```
DIR T=P, L=DEMO
```

名前に M を含むすべてのオブジェクトを選択するには、次のコマンドを使用します。

```
DIR N=*M*
```

バッファプール設定の表示

PARAM コマンドにより、次の設定が表示されます。

```
Active since ..............: 27-JAN-2016 17:30:34, Version 8.3(837) BP version 1
Last time cleared .........: 27-JAN-2016 17:30:34

Bpid ......................: NATBP
Readonly ..................: no
Shmkey ....................: 0x18371389
Semkey ....................: 0x18371389
Memsize ...................:   20971508
Maxusers ..................:         50
```

Bpid	バッファプール ID
Readonly	読み取りアクセスのみを許可する特殊なバッファプールであるかどうかを示します。
Shmkey	バッファプールの作成またはバッファプールへの接続に使用される一意の名前です。
Semkey	バッファプールメモリへのアクセスを同期するために使用される一意の名前です。
Memsize	使用可能な共有メモリのサイズです。
Maxusers	バッファプールへ同時にアクセス可能なユーザーの最大数。

『*コンフィグレーションユーティリティ*』ドキュメントの「*バッファプール割り当て*」を参照してください。

バッファプールに関する統計情報

STATUS コマンドにより、次の統計が表示されます。

```
Active since ...........: 18-JAN-2016 11:57:32, Version 8.3(837) BP version 1
Last time cleared ......: 18-JAN-2016 11:57:32
Bpid ...................: NATBP
Allocated memory (b) ...:      19142488 Max users ............:            50
Smallest allocation ....:            32 Current users ........:             1
Largest allocation .....:       4707272 Peak users ...........:            27
Free memory (b) ........:       1829032 Dead users purged ....:            15
Smallest free ..........:           136
Largest free ...........:         52952

Dormant objects ........:          1378 Smallest object (b) ..:           108
Active objects .........:             0 Largest object (b) ...:         83890
Generating objects .....:             0 Total object sizes ...:      13577572
Obsolete objects .......:             0

Attempted locates ......:     197948090 Stored objects .......:             0
Attempted fast locates .:     103794754 Loaded objects .......:       2323293
Successful fast locates.:     100344415 Activated objects ....:     190326501
Percent ................:         96.68 Aborted loads ........:        328152

Dormant objects purged .:         63731 Peak parallel activations:         10
Object reusage factor ..:         81.92
```

全般的な情報	
Active since	バッファプールが開始された日時、バッファプールのバージョン番号と内部バージョン（BP バージョン）
Last time cleared	バッファプールが最後にクリアされた日時。
Bpid	バッファプール ID
メモリの割り当て	
Allocated memory (bytes)	すべての割り当て済みメモリの合計
Smallest allocatio	割り当て済みメモリの最小容量
Largest allocation	割り当て済みメモリの最大容量
Free memory (bytes)	すべての空きメモリの合計
Smallest free	連続する空きメモリの最小容量
Largest free	連続する空きメモリの最大容量
ユーザー統計	

Max. users	バッファプールへ同時にアクセス可能なユーザーの最大数。『*コンフィグレーションユーティリティ*』ドキュメントの「*バッファプール割り当て*」を参照してください。
Current users	現在バッファプールを使用しているユーザー数。
Peak users	バッファプールを使用しているユーザーの最大数。
Dead users purged	バッファプールから削除された非アクティブなユーザーの数。この数値は0（ゼロ）に近い値であることが必要です。この数値の増分は、バッファプールユーザーのエントリ（Naturalセッション）が無条件にキャンセルまたは強制終了されたことを示しています。このようなユーザーのエントリがバッファプールマネージャによって識別されるたびにこの数値が増加し、キャンセルされたセッションによってクリーンアップが実行され、バッファプールに残っている残差を除去します。
オブジェクトの使用統計	
Dormant objects	使用可能であるが非アクティブなオブジェクトの数です。これらのオブジェクトはバッファプールにありますが、使用されていません。これらは後で使用でき、バッファプールユーザーに使用を要求されるとすぐにアクティブオブジェクトになります。
Active objects	アクティブオブジェクトの数。これらのオブジェクトは現在1人以上のバッファプールユーザーによって使用されています。
Generating objects	現在バッファプールにロードされているオブジェクトの数。これらのオブジェクトは、ロード操作が完了するとすぐに使用可能になります。
Obsolete objects	バッファプールから削除されるものの、まだ使用されているオブジェクトの数です。これらのオブジェクトは、CORPSES コマンドを使用して表示できます。廃止オブジェクトは、このオブジェクトをアクティブ化したすべてのユーザーがこのオブジェクトを解放するとすぐにバッファプールから削除されます。実稼働環境では、この数値は0（ゼロ）にする必要があります。ゼロ以外の値は、NATBPMON の DELETE コマンドを使用してオブジェクトが削除されたか、新しいオブジェクトが作成されたために古くなったことを示します（例えば、CATALOG コマンドのため）。
オブジェクトサイズの統計	
Smallest object (bytes)	バッファプール内で最小のオブジェクトのサイズ。
Largest object (bytes)	バッファプール内で最大のオブジェクトのサイズ。
Total object sizes	バッファプール内のすべてのオブジェクトの合計サイズ。
統計情報の検索	
Attempted locates	オブジェクトの検索に成功した回数と失敗した回数。この数値は、バッファプールマネージャがバッファプール内のオブジェクトの検索を要求した回数を示します。
Attempted fast locates	既知のスロットでのアクティブ化の試行回数。これは、オブジェクトの以前の場所がわかっていた場合のオブジェクトのアクティブ化の回数です。再度アクティブ化されたときに、おそらくバッファプール内の同じ場所にオブジェクトが見つかります。
Successful fast locates	成功した高速ロケートの回数。
Percent	成功した高速ロケートの割合。

オブジェクトのロード統計	
Stored objects	バッファプールに格納されているオブジェクトの数。これは、バッファプールに格納され、システムファイルからロードされなかったオブジェクトの数です。
Loaded objects	システムファイルにロードされたオブジェクトの数。バッファプールにオブジェクトが見つからない場合は、毎回システムファイルからロードされます。この数値は、オブジェクトがバッファプールにロードされるたびに増加します。
Activated objects	アクティブ化されたオブジェクトの数。アクティブ化とは、バッファプール内にあるオブジェクトを、バッファプールユーザーが「使用中」としてマークするプロセスです。
Aborted loads	バッファプール内のメモリ不足のため、またはバッファプールにオブジェクトをロードするときにエラーが発生したために中断されたロード操作の回数です。この数値には顕著なばらつきがないことが必要です。
全般的なロード統計	
Dormant objects purged	新しくロードされたオブジェクトのための空き領域を確保するためにバッファプールから削除された、未使用のオブジェクトの数です。
Peak parallel activations	バッファプール内のいずれかのオブジェクトの並行アクティベーションの最大数です。
Object reusage factor	オブジェクトが再アクティブ化された平均回数です。この数値は、バッファプールにロードされたオブジェクト数に対するオブジェクトアクティベーション数の比率です。

7 バッチモードでの Natural

この章では、Natural をバッチモードで実行する場合の特殊な考慮事項について説明します。

バッチモードとは？

Natural では、2 つの処理モードを区別しています。

- （Natural スタジオを使用する）対話式モード
- バッチモード

2つのモードの主な違いは、対話式モードでは、コマンドとデータがユーザーによりキーボードから入力され、出力が画面に表示されることです。バッチモードでは、ユーザーが介入することなく、入力がファイルから読み取られ、出力がファイルに書き込まれます。

Naturalがバッチジョブとして実行されている場合、バッチジョブを実行したユーザーがNaturalを操作する必要ありません。バッチジョブは、順番の決まった入力データを受け取って順番に実行されるプログラムで構成されます。

バッチモードは、大量のデータを定期的に処理する場合に便利です。

バッチモードでの Natural セッションの開始

バッチモードは、パラメータ BATCHMODE を使用してアクティブ化されます。

手順 7.1. バッチモードでNatural セッションを開始するには

1 以下に示すように、ダイナミックパラメータ BATCHMODE で Natural を起動します。

```
natural BATCHMODE
```

上記の呼び出し（BATCHMODE パラメータのみが指定されている場合）では、必要な入出力チャネルがコンフィグレーションユーティリティで定義済みであるとみなされます。入出力チャネルについて詳しくは、このセクションで後述する「*バッチモードでの Natural の使用*」を参照してください）。パラメータファイル内のバッチモードに関連するプロファイルパラメータの詳細については、『コンフィグレーションユーティリティ』ドキュメントの「バッチモード」を参照してください。

また、必要な入出力チャネルをダイナミックパラメータとして上記の呼び出しに追加することもできます。これについては、このセクションの「*バッチモードのサンプルセッション*」で後述します。上記の呼び出しでダイナミックパラメータとして指定された入出力チャネルは、パラメータファイル内のチャネル定義を上書きします。

2 出力チャンネルとして定義されているファイルをチェックします。終了時には、このファイルには、セッションが正常に終了したことを示すメッセージが含まれている必要があります。

バッチモードでの Natural セッションの終了

バッチモードの Natural セッションは、セッション中に次のいずれかを検出すると終了します。

- バッチ入力ファイルのシステムコマンド FIN、または
- 実行中の Natural プログラム内の TERMINATE ステートメント。

注意: バッチ入力ファイルで入力終了条件が発生すると、バッチセッションも終了します。この場合、出力チャネルとして定義されたファイルに、予期しない終了を示すメッセージが含まれています。

バッチモードでの Natural の使用

バッチモードで Natural セッションを開始するには、ダイナミックパラメータ BATCHMODE を指定する必要があります。さらに、入出力チャネルは以下で説明するように定義する必要があります。

重要: バッチモードでは、入力チャネル CMSYNIN および／または CMOBJIN、ならびに出力チャネル CMPRINT が常に必要です。

以下では次のトピックについて説明します。

- 入出力チャネル
- 入力および出力ファイルのコードページ

入出力チャネル

バッチモードには次のパラメータを使用できます。

パラメータ	説明
CMSYNIN	Natural プログラムの実行中に INPUT ステートメントによって読み込まれる Natural コマンドとデータ（オプション）を含むバッチ入力ファイルを定義します。
CMOBJIN	INPUT ステートメントで読み取るデータを含むバッチ入力ファイルを定義します。このデータは、該当する RUN または EXECUTE コマンドの直後に、パラメータ CMSYNIN で定義されたファイルに配置することもできます。
CMPRINT	Natural プログラムの DISPLAY、PRINT および WRITE ステートメントから得られた出力のバッチ出力ファイルを定義します。

パラメータ	説明
CMPRT*nn*	セッション中に実行される任意の Natural プログラムによって参照される追加レポートの出力ファイルを定義します。*nn* は、DISPLAY、PRINT、または WRITE ステートメントで使用されているレポート番号に対応する 01～31 の 2 桁の 10 進数です。
CMWRK*nn*	セッション中に実行される任意の Natural プログラムによって参照されるワークファイルを定義します。*nn* は、READ WORK FILEまたは WRITE WORK FILE ステートメントで使用されている番号に対応する 01～32 の 2 桁の 10 進数である必要があります。
NATLOG	パラメータ CMPRINTで定義されたバッチ出力ファイルに書き込むことができなかったメッセージを記録するために使用します。バッチモードで NATLOG を有効にすることをお勧めします。

入力および出力ファイルのコードページ

以下のパラメータは、入力ファイルのエンコードおよび出力ファイルのエンコードに使用されるコードページを指定するために使用されます。

パラメータ	説明
CPSYNIN	コマンドのバッチ入力ファイルのエンコードに使用するコードページを指定します。このファイルは、パラメータ CMSYNIN で定義されます。
CPOBJIN	データのバッチ入力ファイルのエンコードに使用するコードページを指定します。このファイルは、パラメータ CMOBJIN で定義されます。
CPPRINT	バッチ出力ファイルのエンコードに使用するコードページを指定します。このファイルは、パラメータ CMPRINT で定義されます。

CMSYNIN および CMOBJIN に対するエンコード：

- 入力ファイル CMSYNIN または CMOBJIN のいずれかにコードページが指定されている場合、入力ファイルのデータはこのコードページを使用してエンコードされているとみなされます。
- 入力ファイル CMSYNIN または CMOBJIN のいずれかにコードページが指定されていない場合、入力ファイル内のデータは、Natural パラメータ CP で指定されたデフォルトのコードページを使用してエンコードされているとみなされます。
- Natural パラメータ CP でコードページが指定されていない場合、入力ファイルのデータは現在のシステムコードページを使用してエンコードされているとみなされます。

CMPRINT に対するエンコード：

- 出力ファイル CMPRINT にコードページが指定されている場合、出力データはこのコードページを使用してエンコードされます。
- 出力ファイル CMPRINT にコードページが指定されていない場合、出力データは、Natural パラメータ CP で指定されたデフォルトのコードページを使用してエンコードされます。
- Natural パラメータ CP でコードページが指定されていない場合、出力データは現在のシステムコードページを使用してエンコードされます。

エンコード／デコードが失敗した場合（例えば、ファイルのエンコードに使用されるコードページに含まれていない文字が CMPRINT に書き込まれた場合）、バッチジョブはスタートアップエラー42（バッチモードドライバエラー）で終了し、エンコード／デコードエラーが発生したファイルが示されます。

特に、これらの各パラメータでは、コードページとして UTF-8 を指定することができることに注意してください。これにより、UTF-8 でエンコードされた Unicode データの読み取りと書き込みが可能になります。

バッチモードのサンプルセッション

この例では、バッチモードで Natural を起動する方法を説明します。単純な Natural プログラムが実行され、データ項目がバッチ入力ファイルから取得されます。この項目が INPUT ステートメントで処理された後、続く DISPLAY ステートメントによってデータがバッチ出力ファイルに書き込まれます。続いて、Natural が終了します。

この例では、ライブラリ SYSEXBAT に保存されているプログラム RECCONT を使用しています。

注意: このライブラリに保存されているオブジェクトの詳細については、ライブラリ SYSEXBAT 内のテキスト A-README を参照してください。

サンプルセッションが、次のコールによって呼び出されます。

```
natural BATCHMODE CMSYNIN=cmd.txt CMOBJIN=data.txt CMPRINT=out.txt NATLOG=ALL
```

注意: このコールでは、すべてのファイルが現在のディレクトリにあり、出力がこのディレクトリに書き込まれることを想定しています。ファイルが別のディレクトリにある場合、または出力を別のディレクトリに書き込む場合は、パスを指定する必要があります。

上記の呼び出しのパラメータを以下で説明します。

BATCHMODE
パラメータ BATCHMODE はバッチモードを有効にし、システム変数の値 *DEVICE を BATCH に設定します。

CMSYNIN=cmd.txt
バッチ入力ファイル *cmd.txt* は、ファイルシステムに保存されるテキストファイルです。このファイルの内容を以下に示します。ライブラリ SYSEXBAT へのログオン、Natural プログラム RECCONT の実行、および Natural セッションの終了のための Natural システムコマンドが含まれています。

```
LOGON SYSEXBAT
EXECUTE RECCONT
FIN
```

Natural プログラム RECCONT の内容は次のとおりです。

```
DEFINE DATA
LOCAL
  1 #firstname   (A10)
  1 #lastname    (A10)
END-DEFINE
INPUT (IP=OFF AD=M) #firstname #lastname
DISPLAY #firstname #lastname
END
```

CMOBJIN=data.txt

RECCONT プログラム内の INPUT ステートメントは、バッチ入力ファイル *data.txt* で定義されたデータを使用します。これは、ファイルシステムに保存されているテキストファイルです。このファイルの内容を以下に示します。

```
Ben %
Smith
```

注意: パーセント文字（%）は、情報が次行に続くことを示します。

CMPRINT=out.txt

プログラム RECCONT 内の DISPLAY ステートメントは、ファイルシステムで作成されたバッチ出力ファイル *out.txt* にデータを書き込みます。このファイルの内容を以下に示します。

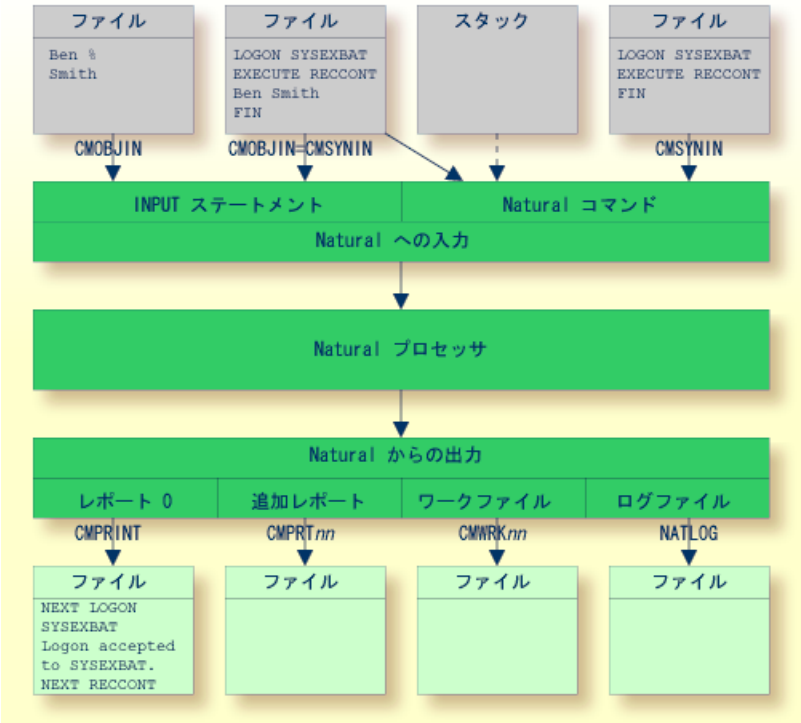
```
NEXT LOGON SYSEXBAT
Logon accepted to library SYSEXBAT.
NEXT EXECUTE RECCONT



DATA Ben %
DATA Smith
Page     1                                                  25.04.05  13:39:09

#FIRSTNAME #LASTNAME
---------- ----------

Ben        Smith
NEXT FIN
NAT9995 Natural session terminated normally.
```

NATLOG=ALL

上記のコールでサンプルセッションを呼び出すと、（バッチ入出力ファイルファイルの名前も含めて）すべてのタイプのメッセージを含むログファイルが作成されます。通常、ログ

ファイルは、ローカルコンフィグレーションファイルに定義されている Natural の一時ディレクトリに作成されます。NATLOG パラメータの説明を参照してください。

次の図は、バッチモードでNaturalが入力を読み取り出力を書き込むさまざまな方法を示しています。



上図に示すように、次のいずれかの手順に従います。

- CMOBJIN および CMSYNIN
  バッチ入力には異なるファイルが使用されます。あるファイルには Natural コマンドが含まれ、別のファイルにはデータが含まれます。

```
natural BATCHMODE CMSYNIN=cmd.txt CMOBJIN=data.txt CMPRINT=out.txt
```

- CMSYNIN
  バッチ入力には1つのファイルが使用されます。これには Natural コマンドとデータの両方が含まれています。

```
natural BATCHMODE CMSYNIN=data.txt CMOBJIN=data.txt CMPRINT=out.txt
```

注意: 使用されるバッチ入力ファイルは1つのみですが、パラメータ CMSYNIN と CMOBJIN の両方を指定する必要があります。両方のパラメータが同じファイルを参照する必要があります。

- CMOBJIN およびSTACK
  バッチ入力には 1 つのファイルが使用されます。これにはデータが含まれています。Natural コマンドは、プロファイルパラメータ STACK を使用して指定します。

```
natural BATCHMODE CMOBJIN=data.txt STACK="(LOGON SYSEXBAT; RECCONT;FIN)"
```

バッチモード検出

システム変数*DEVICEは、Naturalがバッチモードまたは対話式モードのどちらで実行されているかを示します。

モード	説明
バッチモード	*DEVICE には値 BATCH が含まれます。この値は、パラメータ BATCHMODE によって設定されます。
対話式モード	*DEVICEには BATCH 以外の値が含まれています。ほとんどの場合、値 VIDEO が含まれています。

例：

```
IF *DEVICE = "BATCH" THEN
   WRITE 'This is the background task'
ELSE
   WRITE 'This is the interactive session'
END-IF
```

バッチモード制限

Naturalがバッチモードで実行されている場合は、一部の機能が使用できなくなるか、無効になります。

- 対話式入力または対話式出力はできません。
- INPUT ステートメントのデータのみを処理できます。
- 端末データベース SAGtermcap はサポートされていません。そのため、別の文字セットに使用される端末機能 TCS はサポートされていません。別の文字セットを使用するには、代わりに環境変数 NATTCHARSET を使用してください。
- CMPRINT で定義されたバッチ出力ファイルには、色およびビデオ属性は書き込まれません。
- 充填文字は INPUT ステートメント内には表示されません。

- 特定のNaturalシステムコマンドはバッチモードで実行できないため、無視されます。『システムコマンド』ドキュメントには、この制限が適用される各システムコマンドに対応する注記が記載されています。

バッチモードシミュレーション

前述のバッチモードに加えて、バッチモードのシミュレートも可能です。ただし、バッチモードのシミュレーションではなく、バッチモードを使用することをお勧めします。バッチモードには、バッチモードシミュレーションと比較して次の利点があります。

- キーワードデリミタモードがサポートされており、簡単なデータ入力が可能です。
- 構成可能なフォーマット済みの出力処理。
- 拡張エラー処理。
- 起動およびシャットダウンの高速化。
- プログラムの実行の高速化。

入力チャネルがファイルにリダイレクトされる場合、Natural は入力コマンドとデータをキーボードからではなく、このファイルから読み取ります。端末で行うのとまったく同じ方法でデータを指定する必要があります。例えば、入力フィールドが2つあるの場合、2つ目のフィールドに配置するには、最初のフィールドの末尾を空白で満たす必要があります。キーワードデリミタモードはサポートされていません。キーワードデリミタモードを使用するには、バッチモードシミュレーションの代わりにバッチモードを使用します。

出力チャネルがファイルにリダイレクトされると、Naturalは画面に表示されるすべての出力をこのファイルに書き込みます。制御シーケンスもファイルに書き込まれるため、ファイルを読み取れなくなります。フォーマット済みの出力を取得するには、バッチモードシミュレーションではなくバッチモードを使用します。

Naturalの起動時にダイナミックパラメータ `BATCH` を使用して、システム変数 `*DEVICE` を値 `BATCH` に設定します。この値は Natural プログラムで確認できます。

例：入力チャネルのリダイレクト

```
natural BATCH < input-file-name
```

Naturalは、この入力ファイルからすべての入力操作を受け取ります（この入力ファイルの例を以下に示します）。

例：入出力チャネルのリダイレクト

```
natural BATCH < input-file-name > output-file-name
```

Naturalレポートのみを保持し、他のすべての出力を非表示にする場合（NULLデバイスへの出力の書き込み）、プロファイルパラメータ `MAINPR` を有効なプリンタ番号に設定し、パラメータ

ファイル内の対応する論理プリンタ（デバイス）に実行可能ファイルを割り当て、次のように指定します。

```
natural BATCH < input-file-name > /dev/null
```

Naturalレポートはすべて実行可能ファイルに書き込まれますが、画面出力は省略されます。入力ファイルは、Natural で入力がまったく必要ない場合でも指定する必要があります。この場合、NULL デバイスも使用できます。

サンプル入力ファイル

```
dlist program *^M
fin^M
```

バッチモードシミュレーションの入力ファイルには、対話式セッションで実行する同じキー入力が含まれている必要があります。

上記のサンプル入力ファイルでは、次のキー入力が使用されます。

d	［**Direct Command**］ウィンドウを開きます。
プログラムのリストを表示します。*	すべてのプログラムのリストの表示に使用される Natural システムコマンドを実行します。
^M	CTRL+M キーの組み合わせ（改行）の省略表記です。ENTER キーをシミュレートします。
FIN	Natural セッションの終了に使用される Natural システムコマンドを実行します。
^M	CTRL+M キーの組み合わせ（改行）の省略表記です。ENTER キーをシミュレートします。

8 NATCONV.INI を使用した異なる文字セットのサポート

コンフィグレーションファイル *NATCONV.INI* の設定が A フォーマットに適用されます。U フォーマットの場合は、ICU ライブラリが使用されます。

この章では、Natural がどのように文字セットをサポートしているかについて説明します。

さまざまな文字セットのサポートが重要なのはなぜですか？

さまざまな文字セットを使用した複数言語のサポートは、国際化に対する Natural のアプローチを表しています。次を使用する場合に役立ちます。

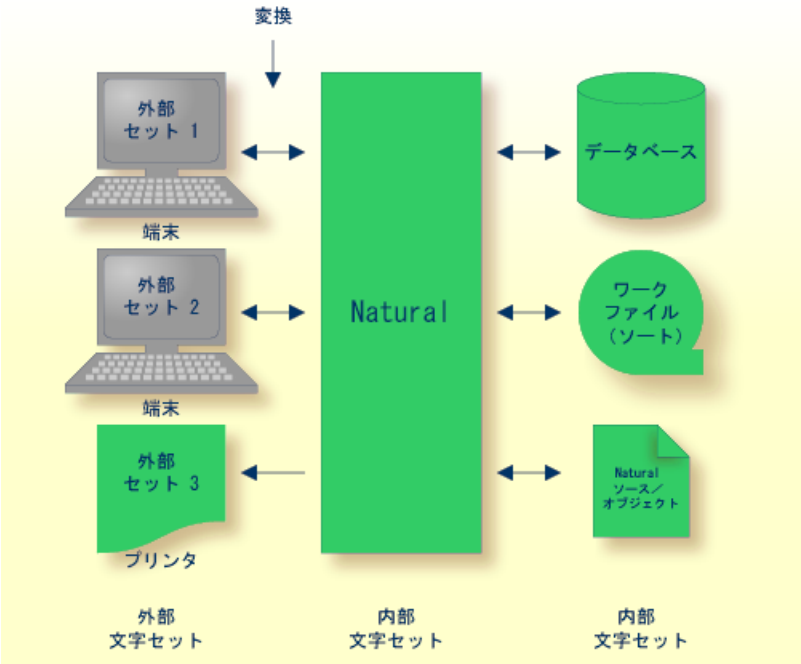
- すべて同じ Natural 環境と通信するさまざまな文字セットを持つ端末およびプリンタ。
- 1 つのデータベースを共有し、異なるプラットフォームに配置された複数の Natural 環境。
- 言語固有の文字の大文字／小文字変換。
- Natural の識別子、オブジェクト名、およびライブラリ名に使用される言語固有の文字。
- マスク定義と比較したオペランド内の言語固有の文字（『*プログラミングガイド*』の「*MASK オプション*」を参照）。

サポートされている文字セット

Natural では、最下位の 7 ビットの ASCII 文字セットに従ったあらゆるシングルバイト文字セットをサポートしています。

Natural では、内部文字といくつかの外部文字セットを区別します。内部文字セットは、Natural 自体によって使用されます。

次の図に示すように、内部文字セットと外部文字セット間の変換は、端末からの入力後、端末またはプリンタへの出力前に実行されます。ワークファイル I/O、データベース I/O、および Natural オブジェクトの読み取り／書き込みに使用できる外部文字セットへの変換はできません。



内部文字セット

デフォルトでは、Naturalは内部文字セットISO8859_1を使用します。デフォルトの文字セットが要件を満たしていない場合は、Naturalまたはその他の標準の文字セットで提供されている定義済みの文字セットのいずれかを選択できます。

注意: 同じデータベースを共有している内部文字セットが異なるコンピュータを実行した場合、またはこれらのコンピュータ間でデータやNaturalオブジェクトを交換しようとした場合に、問題が発生する可能性があります。

外部文字セット

任意の端末およびプリンタに外部文字セットを定義できます。

端末の場合、文字セットの名前は端末データベースの TCS エントリによって定義されます。次に例を示します。

```
:TCS = usascii:
```

また、すべての TCS 設定を上書きする UNIX 環境変数 `$NATTCHARSET` を使用することもできます。

TCS エントリと論理 `NATTCHARSET`（環境変数 `$NATTCHARSET` で設定）のどちらも定義されていない場合、端末 I/O 中に変換は実行されません。

プリンタの場合、プリンタプロファイルで外部文字セット名を定義できます。これは、グローバルコンフィグレーションファイルの一部です。『コンフィグレーションユーティリティ』ドキュ

メントの「*コンフィグレーションファイルのパラメータの概要*」内の「*プリンタプロファイル*」を参照してください。

異なる文字セットを使用する方法

言語固有の文字をサポートするために Natural で使用されるすべてのチェック、変換、および分類テーブルが、コンフィグレーションファイル *NATCONV.INI* 内に存在します。デフォルトでは、このファイルは Natural の *etc* ディレクトリにあります。

NATCONV.INI を変更して、ローカルまたはアプリケーション固有の文字セットをサポートできます。

標準アプリケーションでは *NATCONV.INI* を変更する必要はありません。変更すると、特に Natural オブジェクトとデータベースデータがすでに存在する場合に、重大な不整合が発生する可能性があるため、変更しないでください。

次のいずれかのことを行う場合は、変更が必要です。

- デフォルト以外の内部文字セットを使用する場合、
- *NATCONV.INI* でサポートされていない文字セットを含む端末またはプリンタを使用する場合、
- 識別子で特定の文字の使用を許可または禁止する場合、
- `MASK` オプションの評価にローカル文字をサポートする場合。

NATCONV.INI を変更する場合は十分に検討し、慎重に実行する必要があります。そうでない場合、特定が困難な問題が発生する可能性があります。

NATCONV.INI はセクションとサブセクションに分かれています。次のセクションが定義されています。

セクション	説明
`CHARACTERSET-DEFINITION`	このセクションでは、内部文字セットの名前を定義します。デフォルトは ISO8859_1 です。 別の文字セットを選択する場合は、この文字セットのサブセクションを以下のセクションに含める必要があります。
`CHARACTERSET-TRANSLATION`	このセクションでは、内部文字セットと外部文字セット間の変換に必要なテーブルについて説明します。 例えば、`:TCS = ASCII_GERMAN:` の `SAGtermcap` にエントリを持つ端末を使用し、ISO 8859_1 が内部文字セットとして使用されている場合、このセクションには次の 2 つのサブセクションが含まれている必要があります。

セクション	説明
	■ [ISO8859_1->ASCII_GERMAN] ■ [ASCII_GERMAN->ISO8859_1]
CASE-TRANSLATION	このセクションには、次のいずれかが指定された場合に実行される大文字から小文字への変換に必要なテーブルが含まれています。 ■ 端末コマンド %U、 ■ フィールド属性 AD=T、 ■ ステートメント EXAMINE TRANSLATE。 この変換は、内部文字セット内で行われます。内部文字セットが ISO 8859_5 などの場合、このセクションには次の 2 つのサブセクションが含まれている必要があります。 ■ [ISO8859_5->UPPER] ■ [ISO8859_5->LOWER]
IDENTIFIER-VALIDATION	このセクションには、識別子（ソースプログラムのユーザー定義変数）、オブジェクト名、およびライブラリ名の検証に必要なテーブルが含まれています。定義されたそれぞれの内部文字セットのサブセクションを含んでいます。 特殊文字"#"（データベース変数以外の場合）、"+"（アプリケーション独立変数の場合）、"@"（SQL および Adabas の空値／長さインジケータの場合）、および"&"（ダイナミックソース生成の場合）は、このセクションで再定義できます。さらに、識別子、オブジェクト名、およびライブラリ名で有効な最初の文字と後続の文字のセットを変更できます。 **注意:** x7f（10 進数で 127）より大きい値を持つオブジェクト名の有効な文字セットを拡張する場合、オブジェクトのソート順序（LIST * コマンド時など）が数値順にならないことがあります。
CHARACTER-CLASSIFICATION	このセクションには、文字の分類に必要なテーブルが含まれています。例えば、MASK オプションを評価するときに使用されます。定義されたそれぞれの内部文字セットのサブセクションを含んでいます。

セクション CHARACTERSET-DEFINITION と各サブセクションには、文字の変換方法と、どの文字がどの属性に関連しているかを示す行が含まれています。これらの行は次のように表されます。

```
line          ::=  key = value
key           ::=  name_key | range_key
name_key      ::=  keyword{ CHARS }
keyword       ::=  INTERNAL-CHARACTERSET | NON-DB-VARI | DYNAMIC-SOURCE |
                     GLOBAL-VARI | FIRST-CHAR | SUBSEQUENT-CHAR |
                     LIB-FIRST-CHAR | LIB-SUBSEQUENT-CHAR | ALTERNATE-CARET
                     ISASCII | ISALPHA | ISALNUM | ISDIGIT | ISXDIGIT |
                     ISLOWER | ISUPPER | ISCNTRL | ISPRINT | ISPUNCT |
                     ISGRAPH | ISSPACE
```

```
range_key     ::=  hexnum | hexnum-hexnum
value         ::=  val {, val }
val           ::=  hexnum | hexnum-hexnum
hexnum        ::=  xhexdigithexdigit | xhexdigithexdigit
```

注意:

1. range_key 変数が左側に指定されている場合、右側に指定されている値の数は、キー範囲に指定されている値の数に対応している必要があります。ただし、右側に指定されている値が 1 つのみの場合は、キー範囲の各要素に割り当てられます。
2. name_key 変数が左側に指定され、対応する文字コードのリストが 1 行に収まらない場合は、name_key = を再度指定することで次の行に続けることができます。行の先頭に空白またはタブを使用しないでください。

有効な行の例：

x00-x1f = x00	x00 と x1f との間のすべての文字が x00 に変換されます。
x00-x7f = x00-x7f	x00 と x7f との間の文字は変換されません。
x00-x08 = x00,x01-x07,x00	文字 x00 および x08 が x00 に変換され、x01 と x07 との間の文字は変換されません。
ISALPHA = x41-x5a,x61-x7a,xc0-xd6,xd8 ISALPHA = xd9-xf6,xf8-xff	この 2 行で指定されたすべての文字に ISALPHA 属性が割り当てられます。

無効な行の例：

x41 = 'A'	文字はすべて 16 進形式で指定する必要があります。
0x00-0x1f = 0x00	16 進値は、次のいずれかの方法で指定する必要があります。 x*digitdigit* X*digitdigit*
x00-x0f = x00,x01	指定された値の数が、キー範囲の要素の数に対応していません。

9 Natural の出口コード

Natural の出口コードには次の 2 つのタイプがあります。

- スタートアップエラー。出口コード 0 および 1 は成功を示し、その他の出口コードはすべてエラーを示します。
- TERMINATE ステートメントによって生成されたエラー。出口コードは 0～255 です。

Natural スタートアップエラー

Natural を起動すると、次の出口コードが発生する場合があります。

2	SAGtermcap または環境変数 NATTCHARSET で指定された端末コントロール文字列（TCS）機能。
3	文字変換テーブルの初期化に失敗しました。
4	文字変換ファイル *NATCONV.INI* にエラーがあります。
5	グローバルコンフィグレーションファイル *NATCONF.CFG* からデータベース割り当てを読み込めません。
6	FNAT(*dbid*,*fnr*) または FUSER(*dbid*,*fnr*) が見つかりません。コンフィグレーションファイルを確認してください。
7	LFILE テーブルを初期化できません。
8	現在は使用されていません。
9	現在は使用されていません。
10	現在は使用されていません。
11	現在は使用されていません。
12	指定されたパラメータファイルを読み込めません。パラメータファイルを確認してください。
13	パラメータファイル NATPARM を読み取れません。
14	ストレージマネージャの初期化に失敗しました。
15	STDIN からの読み込み中にエンドオブファイル（EOF）になりました。
16	バッファプールを開けません。Natural システム管理者に連絡してください。
17	*NATURAL.INI* ファイルからバッファプール割り当てを読み込めません。
18	FDIC 割り当てが正しくありません。
19	FNAT 割り当てが正しくありません。
20	FSEC 割り当てが正しくありません。
21	FUSER 割り当てが正しくありません。
22	Natural ログインモジュールをロードできません。
23	ローカルデータにメモリを割り当てることができません。USIZE パラメータおよび／または SSIZE パラメータの値を小さくしてください。
24	Natural ディスプレイモジュールをロードできません。
25,26	共有可能なイメージまたは DLL のロードエラー。
27	ログインがキャンセルされました。Natural が終了します。

28	Natural の起動時にセキュリティ違反が発生しました。Natural が終了します。
29	Natural の起動時にセキュリティ違反が発生しました。ログイン失敗回数が多すぎるため、ログインが中止されました。
30	Natural システムエラーメッセージが発生しました。
31	NAT0866 Natural ニュークリアスが Natural Security ニュークリアスではありません。
32	パスワードの確認に失敗しました。
33	ロックマネージャでは、セマフォを作成／初期化できません。
34	指定された FNAT/FUSER.にアクセス可能なライブラリがないか、ライブラリが存在しません。システムファイル割り当てと FNAT および FUSER（ディレクトリとファイル）のファイル属性を確認します。
35	WFC I/O 端末ドライバの内部エラー。
36	内部 XVT エラー。
38	ランタイムコンテキストの作成に失敗しました。
39	NATDIR 環境変数および／または NATVERS 環境変数が見つかりません。NATDIR 環境変数を設定した場合は、無効な文字や空白文字が含まれていないことを確認してください。NATVERS には Natural バージョンのみが含まれている必要があります。パスには有効なドライブ ID が含まれている必要があります。
40	Natural zmodem エラー。
41	古いパラメータモジュールとはデータベースタイプが異なるエントリがあるため、TF テーブルの作成に失敗しました。Natural コンフィギュレーションユーティリティでパラメータモジュールを確認してください。
42	バッチモードドライバエラー。
43	画面ウィンドウのサイズが小さすぎます。
44	SQL シグナルハンドラを終了します。
45	アドオン製品をロードできません。
46	FNAT ライブラリ SYSLIB にアクセスできません。権限が不十分か、ファイル保護違反です。
47	*NATURAL.INI* ファイルから PARM_PATH エントリを読むことができないか、ディレクトリにアクセスできません。
48	*NATURAL.INI* ファイルから CONFIG_NAME エントリを読むことができないか、ファイルにアクセスできません。
49	*NATURAL.INI* ファイルから NATTCAP エントリを読むことができないか、ファイルにアクセスできません。
50	*NATURAL.INI* ファイルから NATCONV エントリを読むことができないか、ファイルにアクセスできません。
51	*NATURAL.INI* ファイルから TMP_PATH エントリを処理できません。パス「*path*」にアクセスできません。
52	*NATURAL.INI* ファイルから PROFILE_PATH エントリを読むことができないか、ディレクトリにアクセスできません。
53	ローカルコンフィグレーションファイル *NATURAL.INI* を開けません。
54	NATOSDEP の *NATCONF.CFG* を読むことができません。

55	NATEXTLIB の *NATURAL.INI* を読むことができません。
57	未使用。
59	認識できないオプション「*option*」が指定されました。
60	内部テーブルを初期化できる十分なメモリがありません。
61	バッチエラーが発生しましたが、CC=ON パラメータにより処理が続行されました。
62	アクティブなリポジトリを持つ複数の Natural セッションは許可されていません。
63	アクティブなリポジトリを持つ Natural セッションがすでに実行されています。
64	FNAT の *LIBDIR.SAG* を開けませんでした。有無の確認とアクセス保護の確認を行います。
65	この Natural セッションに割り当てられた FNAT が古くなっています。
72	これは、Naturalの評価版です..... まで有効です
73	Natural ... のこの評価コピーのテスト期間が期限切れになりました。... まで有効でした
77	FDDM 割り当てが正しくありません。
85	コンテキスト初期化中の Natural ランタイムスタートアップエラー。
86	無効なコードページ [*name*] が指定されました。
87	シグナルハンドラの初期化に失敗しました。
88	バッファプール使用状況が矛盾しています。
91	Natural Web I/O インターフェイスへの接続がありません。
105	ダイナミックパラメータ ITERM の値は、ON または OFF です。
106	初期化中のエラーで終了します。

10 Entire System Server インターフェイスの設定

Entire System Server 製品を使用する場合は、Entire System Server インターフェイスが必要です。Entire System Server インターフェイスは Natural の一部であり、追加のインストールは必要ありません。

また、Natural はライブラリ SYSNPE および SYSNPR を提供します。

SYSNPE は Entire System Server のオンラインチュートリアルです。Entire System Server ユーザーのための起動ヘルプとして提供されています。Entire System Server の詳細については、『Entire System Server』ドキュメントを参照してください。

ライブラリ SYSNPR には、Entire System Server DDM のデータベース ID を変更するために使用されるプログラム CHANGEDB が含まれています。

必要条件

Entire System Server インターフェイスにより、Entire Net-Work を介して z/OS、z/VSE、および BS2000 上の Entire System Server にアクセスできます。Entire System Server インターフェイスを完全にサポートするには、メインフレームプラットフォームで Entire Net-Work バージョン 5.8.1 以上が必要です。

有効化

標準の Natural コンフィグレーション設定を使用している場合、Entire System Server インターフェイスはアクティブではありません。Entire System Server インタイフェイスデータベース（Natural プロファイルパラメータ ESXDB）の値は、デフォルトで 0 に設定されています。Entire System Server インターフェイスを使用するには、コンフィグレーションユーティリティを使用して、パラメータ ESXDB の値を 148 に設定する必要があります。

コンフィグレーションユーティリティでは、パラメータ ESXDB はパラメータファイルのパラメータグループ **Product Configuration** で割り当てられます。

ヒント: "ESXDB" を検索して、このパラメータを見つけます。詳細については、『コンフィグレーションユーティリティ』ドキュメントの「*パラメータの検索*」を参照してください。

ESXDB は Entire System Server の DDM に使用されるデータベース ID を指定します。この DBID は、Entire System Server リクエストのターゲット DBID を指定するものではありませんが、カタログ化された Entire System Server DDM に使用される DBID を Natural に伝えます。効果的な Entire System Server のターゲット DBID は、すべての Entire System Server DDM の一部である NODE フィールドで指定します。

⚠ **重要:** ESXDB の値を 148 に変更して、Entire System Server インターフェイスのサポートにより Natural を実行します。Entire System Server の DDM はすべて DBID 148 でカタログされます。

Natural を再起動した後、メインフレームで実行されている Entire System Server ノードに Entire Net-Work を介してアクセスできます。

Entire System Server インターフェイスのカスタマイズでは、Entire System Server の DDM のみ変更がサポートされます。

Entire System Server の DDM に使用されるデータベース ID の変更

ライブラリ SYSNPR には、すべての Entire System Server DDM のデータベース ID を変更するために使用されるプログラム CHANGEDB が含まれています。ライブラリ SYSNPE には、すべての Entire System Server DDM があります。プログラム CHANGEDB に新しい DBID 値として入力されたデータベース ID は、コンフィグレーションユーティリティの Entire System Server インターフェイスデータベースパラメータ（ESXDB）の値としても指定する必要があります。

11 SQL データベースアクセスの調整

デフォルトでは、Natural SQL ドライバは最近使用した 16 の Natural ステートメントのテーブルを管理します。このテーブルのすべてのステートメントは準備完了とマークされます。これは、ステートメントがデータベースシステムによってコンパイルされることなくすぐに実行できることを示します。

パフォーマンスを最大限まで高めるために、ダイナミックパラメータ SQLRELCMD および SQLMAXSTMT が提供されています。これらのパラメータは、SQL ドライバのステートメントテーブルの処理を設定するものです。これらのパラメータはプロファイルパラメータではないことに注意してください。

SQLRELCMD

このパラメータは、コマンドをSQL ステートメントテーブルからいつ解放するかを決定します。

設定可能値：

- ENDGP（デフォルト）：生成プログラムが終了すると、ステートメントテーブルにあるこのプログラムのすべてのステートメントがテーブルから削除されます。
- NEVER：テーブルからステートメントは削除されません。

SQLMAXSTMT

このパラメータでは、ステートメントテーブルのサイズを決定します。

設定可能値：

- 1～64（デフォルト：16）

SQLMAXSTMT パラメータを設定する場合は、次の点に注意してください。

- テーブル内に準備完了済みのステートメントを多く維持すると、リソースの消費量が増加する可能性があります。
- ステートメントテーブルのサイズが、ターゲットデータベース内のダイナミック SQL ステートメントの制限を超えると、アプリケーションに SQL エラーが発生します。
- これは、SQLMAXSTMT の最適化による利点が実際にあるかどうかに依存します。
- 一般的に、PREPARE ステートメントの数が最小になるとバッチタイプのアプリケーションのパフォーマンスは向上しますが、ターゲットデータベースのリソースの消費が増加するため、オンラインアプリケーションのパフォーマンスが低下する可能性があります。

例

上記のパラメータをダイナミックに設定するには、Natural の起動時に入力します。

```
natural sqlrelcmd=never sqlmaxstmt=40
```

すると、Natural は、ステートメントテーブルサイズを 40 で開始し、Natural が終了した場合にのみクリアされます。

12 ソートキー計算のユーザー出口 - NATUSKnn

一部の言語には、ソートプログラムまたはデータベースシステムで正しいアルファベット順にソートできない文字が含まれています。システム関数 SORTKEY を使用すると、このような「正しくソートされない」文字をアルファベット順で「正しくソートされる」別の文字に変換できます。

Natural プログラム内で SORTKEY 関数を使用すると、ユーザー出口 NATUSK*nn* が呼び出されます。*nn* は現在の言語コード（システム変数 *LANGUAGE の現在の値）です。

NATUSK*nn* ユーザー出口は、CALL インターフェイスを使用して C プログラミング言語で書き込むことができます。SORTKEY で指定した文字列はユーザー出口に渡されます。ユーザー出口は、この文字列の「正しくソートされない」文字を対応する「正しくソートされる」文字に変換するようにプログラミングされている必要があります。続けて、変換された文字列が Natural プログラムで使用され、さらに処理が実行されます。

注意: 変換テーブルが指定されていません。

NATUSK*nn* は CALL インターフェイスを使用して呼び出されます。C 関数のパラメータには、次の値があります。

パラメータ	Contents
1	引数の数。
2	オペランドへのポインタの配列。
3	各オペランドのフィールド情報の配列。

Natural システム関数 #OP1=SORTKEY(#OP2) を使用する場合、ソースオペランドはインデックス 0 の配列にあり、ターゲットオペランド（#OP1）はインデックス 1 の配列にあります。

ユーザー出口のサンプル *natusk01.c* はソースフォームで提供されます。これは英語に適用され、文字列内のすべての英語の小文字を大文字に変換します。サンプルは <*install*-

dir>/*natural*/*samples*/*sysexuex* に用意されています。ここには他のユーザー出口も含まれています。

この例のソースコードには、SORTKEY の特定のユーザー出口を書き込むために必要なすべてのコメントが含まれています。

リンケージ規則およびロード規則については、CALL ステートメントを参照してください。

13 異常終了（アベンド）の処理

シグナル SIGTERM が取得されます。SIGTERM のシグナルハンドラは、現在使用されているすべてのリソースを解放し、Natural をスムーズに終了します。

SIGBUS、SIGSEGV および SIGILL の Natural のデフォルトシグナルハンドラは、コマンド gcore（または Solaris で gcore 機能を提供する同じ名前のスクリプト）を使用でき、Natural を実行しているユーザーがこのコマンドの実行権限を持っている場合にのみインストールされます。これらのシグナルハンドラのいずれかが制御を取得すると、起動時に有効であったハンドラが復元され、gcore が実行され、シグナル SIGTERM が検出されたときと同様の動作が試行されます。

注意: オペレーティングシステムが AIX の場合、機能 coredump が呼び出されます。