THE POWER TO KNOW.

UNIX版SAS® 9.4

第4版

SAS® ドキュメント

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2014. *SAS® 9.4 Companion for UNIX Environments, Fourth Edition*. Cary, NC: SAS Institute Inc.

SAS® 9.4 Companion for UNIX Environments, Fourth Edition

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

August 2014

目次

このドキュメントについて

SAS 言語の構文規則

SAS 言語の構文規則の概要

SAS では、SAS 言語要素の構文ドキュメントに共通の規則を使用しています。これらの規則により、SAS 構文の構成要素を簡単に識別できます。規則は、次の項目に分類されます。

- 構文の構成要素
- スタイル規則
- 特殊文字
- SAS ライブラリと外部ファイルの参照

構文の構成要素

言語要素の多くでは、その構文の構成要素はキーワードと引数から構成されます。キーワードのみ必要な言語要素もあります。また、キーワードに等号(=)が続く言語要素もあります。複数の引数を含む構文で区切り記号を使用する場合と使用しない場合を説明するために、引数の構文の形式が複数示されています。

キーワード
プログラムの作成ときに使用する SAS 言語要素名です。キーワードはリテラルであり、通常、構文の先頭の単語です。CALL ルーチンでは、最初の 2 つの単語がキーワードです。

これらの例の SAS 構文では、キーワードには太字が使用されています。

CHAR (*string, position*)

CALL RANBIN (*seed, n, p, x*);

ALTER (*alter-password*)

BEST *w.*

REMOVE <*data-set-name*>

この例では、CALL ルーチンの最初の 2 つの単語がキーワードです。

CALL RANBIN(*seed*, *n*, *p*, *x*)

引数なしで 1 つのキーワードから構成される SAS ステートメント構文もあります。

DO;
... SAS code ...

END;

2 つのキーワード値のいずれか 1 つの指定が必要なシステムオプションもあります。

DUPLEX | **NODUPLEX**

プロシジャステートメントによっては、ステートメント構文中に複数のキーワードが含まれます。

CREATE <UNIQUE> **INDEX** *index-name* **ON** *table-name* (*column-1* <, *column-2*, …>)

引数

数値定数、文字定数、変数、式のいずれかです。引数は、キーワードに続くか、キーワードの後ろの等号に続きます。SAS では、引数を使用して、言語要素を処理します。引数が必須の場合もオプションの場合もあります。構文では、オプションの引数は山かっこ(< >)で囲まれます。

この例では、*string* と *position* がキーワード CHAR に続きます。これらの引数は、CHAR 関数の必須引数です。

CHAR (*string, position*)

引数ごとに値が指定されます。この例の SAS コードでは、引数 *string* の値は 'summer'、引数 *position* の値は 4 です。

```
x=char('summer', 4);
```

この例では、*string* および *substring* は必須引数ですが、*modifiers* と *startpos* はオプションです。

FIND(*string, substring* <,*modifiers*> <,*startpos*>

argument(s)

引数は必ず 1 つ必要であり、複数の引数が許可されます。引数の間はスペースで区切ります。カンマ(,)などの区切り記号は、引数間に必要ありません。

たとえば、MISSING ステートメントは、この形式で複数の引数を含みます。

MISSING *character(s)*;

<LITERAL_ARGUMENT> *argument-1* <<LITERAL_ARGUMENT> *argument-2 ...* >

引数は必ず 1 つ必要であり、リテラル引数がこの引数に関連付けられます。リテラルと引数のペアは複数指定できます。リテラルと引数の間に区切り記号は必要ありません。省略記号(...)は、追加のリテラルと引数が許可されることを示します。

たとえば、BY ステートメントはこの引数を含みます。

BY <DESCENDING> *variable-1* <<DESCENDING> *variable-2* …>;

argument-1 <option(s)> <argument-2 <option(s)> ...>

引数は必ず 1 つ必要であり、1 つ以上のオプションがこの引数に関連付けられます。複数の引数と関連するオプションを指定できます。引数とオプションの間に区切り記号は必要ありません。省略記号(...)は、追加の引数と関連するオプションが許可されることを示します。

たとえば、FORMAT プロシジャの PICTURE ステートメントは、この形式で複数の引数を含みます。

PICTURE name <(*format-option(s)*)>
<*value-range-set-1* <(*picture-1-option(s)*)>
<*value-range-set-2* <(*picture-2-option(s)*)> …>>;

argument-1=value-1 <argument-2=value-2 ...>
引数には値を割り当てる必要があり、複数の引数を指定できます。省略記号(...)は、追加の引数が許可されることを示します。引数間に区切り記号は必要ありません。

たとえば、LABEL ステートメントは、この形式で複数の引数を含みます。

LABEL *variable-1=label-1 <variable-2=label-2 …>*;

argument-1 <, argument-2, ...>
引数は必ず 1 つ必要であり、カンマまたは別の区切り記号で区切って複数の引数を指定できます。省略記号(...)は、カンマで区切られた引数が続くことを示します。SAS ドキュメントでは両方の形式が使用されます。

次に、この形式で指定された複数の引数の例を示します。

AUTHPROVIDERDOMAIN (*provider-1:domain-1 <, provider-2:domain-2, …>*

INTO :*macro-variable-specification-1 <*, :*macro-variable-specification-2, …>*

注: 通常、SAS ドキュメントのサンプルコードは、小文字の固定幅フォントを使用して表記されます。コードの作成には、大文字も、小文字も、大文字と小文字の両方も使用できます。

スタイル規則

SAS 構文の説明に使用されるスタイル規則には、大文字太字、大文字、斜体の規則も含まれます。

大文字太字
関数名やステートメント名などの SAS キーワードを示します。この例では、キーワード ERROR の表記には大文字太字が使用されています。

ERROR <*message*>;

大文字
リテラルの引数を示します。

この CMPMODEL=システムオプションの例では、BOTH、CATALOG、XML がリテラルです。

CMPMODEL=BOTH | CATALOG | XML |

斜体
ユーザー指定の引数または値を示します。斜体表記の項目は、ユーザー指定値であり、次のいずれかを表します。

- 非リテラル引数。この LINK ステートメントの例では、引数 *label* はユーザー指定値のため、斜体で表示されます。

  LINK *label*;

- 引数に割り当てられる非リテラル値。

  この FORMAT ステートメントの例では、引数 DEFAULT に変数の *default-format* が割り当てられます。

  FORMAT *variable(s) <format* > <DEFAULT = *default-format*>;

特殊文字

SAS 言語要素の構文には、次の特殊文字も使用されます。

=
等号は、一部の言語要素(システムオプションなど)のリテラル値を示します。

この MAPS システムオプションの例では、等号により MAPS の値が設定されます。

MAPS = *location-of-maps*

< >
山かっこはオプションの引数を示します。必須引数は山かっこで囲みません。

この CAT 関数の例では、少なくとも項目が 1 つ必要です。

CAT (*item-1* <, *item-2*, …>)

|
縦棒は、値グループから 1 つの値を選択できることを示します。縦棒で区切られている値は、相互排他です。

この CMPMODEL=システムオプションの例では、引数を 1 つのみ選択できます。

CMPMODEL=BOTH | CATALOG | XML

...
省略記号は、引数の繰り返しが可能なことを示します。引数と省略記号が山かっこで囲まれている場合、その引数はオプションです。繰り返される引数には、その引数の前や後ろに、区切り記号を入れる必要があります。

この CAT 関数の例では、複数の *item* 引数が許可され、カンマで区切る必要があります。

CAT (*item-1* <, *item-2*, …>)

'*value*'または"*value*"
一重引用符や二重引用符付きの引数は、その値にも一重引用符または二重引用符を付ける必要があることを示します。

この FOOTNOTE ステートメントの例では、引数 *text* に引用符が付けられています。

FOOTNOTE <*n*> <*ods-format-options* '*text*' | "*text*">;

;
セミコロンは、ステートメントまたは CALL ルーチンの終わりを示します。

この例では、各ステートメントがセミコロンで終了しています。

```
data namegame;
length color name $8;
color = 'black';
name = 'jack';
game = trim(color) || name;
run;
```

SAS ライブラリおよび外部ファイルの参照

多くの SAS ステートメントなどの言語要素では、SAS ライブラリと外部ファイルを参照します。論理名(ライブラリ参照名またはファイル参照名)から参照を作成するのか、引用符付きの物理ファイル名を使用するかを選択できます。論理名を使用する場合、通常、参照の作成に SAS ステートメント(LIBNAME または FILENAME)を使用するのか、動作環境のコントロール言語を使用するのかを選択します。複数の方法を使用して、SAS ライブラリと外部ファイルを参照できます。動作環境によっては使用できない方法があります。

SASドキュメントでは、外部ファイルを使用する例には斜体のフレーズ *file-specification* を使用します。また、SAS ライブラリを使用する例には斜体フレーズ *SAS-library* を引用符で囲んで使用します。

```
infile file-specification obs = 100;
libname libref 'SAS-library';
```

UNIX 版 SAS 9.4 の新機能

概要

次のカテゴリ別に、UNIX 版 SAS の変更領域を示します。

デフォルトの更新

LRECL=オプションのデフォルトは現在 32,767

LRECL=のデフォルト値が 256 から 32,767 に変更されました。固定長レコードを使用する場合(RECFM=F)、LRECL=のデフォルト値は 256 です。

MEMSIZE システムオプションのデフォルトは現在 2G

SAS 9.4 では、MEMSIZE システムオプションのデフォルト値が 512M から 2G に変更されました。さらに、呼び出し中に**-MEMSIZE MAX** を指定すると、MEMSIZE の値が、ページサイズと、プロセスに使用可能なメモリ制限の両方に基づいて適宜調整されます。

SORTSIZE システムオプションのデフォルトは現在 1G

SORTSIZE=システムオプションのデフォルト値が 256M から 1G に変更されました。SORT プロシジャに対する SORTSIZE=オプションのデフォルト値は、SORTSIZE=システムオプションの値に基づきます。

SAS コマンド: SETENV | UNSETENV

SETENV | UNSETENV コマンドが新しく追加されました。SETENV コマンドを使用すると、変数名または変数値、あるいはその両方を指定して環境変数を定義できます。UNSETENV コマンドを使用すると、変数名を指定して環境変数を削除できます。

SAS プロシジャ: BMDP

SAS 9.4 のメンテナンスリリース 2 では、BMDP プロシジャが廃止されました。BMDP プロシジャを呼び出した場合、SAS では、BMDP ソフトウェアの実行は試行されません。ただし、SAS による BMDP ファイルへの変換および BMDP ファイルからの変換を可能にする BMDP エンジンは、現在も使用可能です。

SAS ステートメント

FILE および FILENAME ステートメント: PERMISSION=オプション

SAS 9.4 のメンテナンスリリース 2 では、FILE および FILENAME ステートメントに対して新しいオプションが使用可能です。PERMISSION=オプションでは、指定ファイル参照名に対して読み取り、書き込み、および実行権限を指定できます。また、設定した権限の適用先を、自分、ファイルのグループ所有者、他のユーザーにするかどうかも指定します。

FILENAME ステートメント: アクセス方法

FILENAME ステートメントには、次の新しいアクセス方法があります。

DATAURL
: これにより、ユーザー指定テキストからデータを読み取れます。.

HADOOP
: 構成ファイルに場所が指定されている Hadoop 分散ファイルシステム(HDFS)上のファイルにアクセスできます。

ZIP
: ZIP ファイルにアクセスできます。

SAS9.4 のメンテナンスリリース 1 では、次のアクセス方法が新しく追加されました。

ACTIVEMQ
: SAS プログラムで、HTTP プロトコルによる ActiveMQ メッセージブローカとのメッセージの送受信を可能にします。

JMS
: SAS プログラムで、JMS API 準拠メッセージサービスとのメッセージの送受信を可能にします。

SAS 9.4 のメンテナンスリリース 1 では、ロックダウン状態の SAS Server に対する処理制限が実装されました。SAS 9.4 のメンテナンスリリース 2 では、SAS がロックダウン状態の場合、FILENAME ステートメントアクセス方法の EMAIL、FTP、HADOOP、SOCKET、URL がアクセス不能(無効)になります。ただし、SAS Server 管理者は、これらのアクセス方法の 1 つ以上を再有効化して、SAS がロックダウン状態の場合もアクセス可能にできます。

SAS システムオプション

ALIGNSASIOFILES システムオプション

新しい ALIGNSASIOFILES システムオプションでは、SAS データセットでデータのページを配置して、パフォーマンスを向上させます。

FILELOCKWAIT=システムオプション

新しい FILELOCKWAIT システムオプションでは、ロックされたファイルが使用可能になるまで SAS が待機する秒数が設定されます。

HOSTINFOLONG システムオプション

新しい HOSTINFOLONG システムオプションでは、SAS の開始時に、追加の動作環境情報を SAS ログに書き込むよう指定されます。

OPLIST システムオプションの拡張

SAS 9.4 のメンテナンスリリース 2 では、OPLIST システムオプションで、SAS の起動時に指定されるパスワード値はいずれも自動的にマスクされます。SAS ログには、マスク値のみが表示されます。

RTRACE システムオプション引数

SAS セッション中に読み込みまたはロードされるリソースのリストを生成する RTRACE システムオプションには、VER と呼ばれる新しい引数があります。この引数では、SAS が読み込みまたはロードする各モジュールのバージョン番号やその他のトレース情報が書き込まれます。

RTRACELOC システムオプションの拡張

SAS 9.4 のメンテナンスリリース 2 では、RTRACELOC システムオプションで生成されるファイル名を拡張して、プロセス ID、日付、およびシステム時間を含められます。これらの値をファイル名に含めるには、それぞれ%p、%d、%t を含めます(たとえば、mytrace.%d.%t.%p など)。

システムパフォーマンス

SAS 9.4 のメンテナンスリリース 1 では、システムパフォーマンスを測定するための iotest.sh ツールのドキュメントが追加されました。“UNIX でのパフォーマンスに関する注意点”という新しい章では、このツールの使用法が説明されます。

ドキュメントの拡張

ファイル拡張子区切り文字の処理方法

ファイル拡張子の処理方法についてのセクションが追加されました。このセクションでは、ピリオドを区切り文字として使用する方法を説明します。

システムオプション内のスペース

システムオプションの値にスペースが含まれる場合は、**`-bufsize='3 k';`**などのように、値を引用符で囲む必要があります。

さまざまな配布先に複数のユーティリティファイルを配布

SORT プロシジャでは、1 つのスレッドプロシジャによって書き込まれた複数のユーティリティファイルをさまざまな場所に配布できます。UTILLOC オプションに対して指定された各場所によって、ユーティリティファイルを配布可能な場所が 1 つずつ識別されます。複数の場所が指定された場合は、ユーティリティファイルの必要時に、SAS アプリケーションによってその場所が順番に使用されます。

UNIX での環境変数

SAS 9.4 のメンテナンスリリース 2 では、UNIX 環境で使用される環境変数を説明するために新しい章が追加されました。この章には、SASV9_CONFIG、SASV9_OPTIONS、および PATHENCODING 環境変数が含まれています。

SAS 9.4 のメンテナンスリリース 2 では、SAS プログラムで提供するいずれのパスも、PATHENCODING 環境変数と SAS セッションエンコーディングのどちらでも識別される文字を含める必要があります。具体的には、英語(LANG=EN)を使用する SAS セッションで UTF-8 の PATHENCODING 値を指定するには、UTF-8 または SAS_U8 の SAS セッションエンコーディングを指定する必要があります。

ロックダウン状態のサーバーに対する処理制限

SAS 9.4 のメンテナンスリリース 1 では、ロックダウン状態の SAS Server に対する処理制限が新しく追加されました。クライアント/サーバー環境で実行中の場合(たとえば、SAS Enterprise Guide の実行中など)、SAS Server 管理者は、SAS クライアントによるディレクトリとファイルのセットへのアクセスを許可する環境を作成できます。その他すべてのディレクトリとファイルは、アクセス不可能になるかロックダウンされます。

SAS Server がロックダウン状態にある場合、次の SAS 関数と CALL ルーチンは使用できません。

ADDR	PEEK
ADDRLONG	PEEKC
CALL MODULE	PEEKCLONG
CALL POKE	PEEKLONG
CALL POKELONG	

注: このドキュメントでは、CALL MODULE ルーチンと PEEKLONG 関数のみがリストされます。CALL MODULE ルーチンと PEEKLONG 関数には、UNIX に固有のプロパティがあります。

EBCDIC データへのアクセス

SAS 9.4 のメンテナンスリリース 2 では、付録 4 “ASCII システムでの EBCDIC データの使用” (495 ページ)という新しい付録が追加されました。この付録では、EBCDIC と ASCII のデータ表現についての背景が提供されます。この付録には、ASCII マシンで EBCDIC を使用するための各種方法の例が含まれます。

ユーザー補助

この製品のユーザー補助の詳細については、SAS ウィンドウ環境のユーザー補助機能(support.sas.com) を参照してください。

推奨資料

- *Base SAS プロシジャガイド*
- *SAS データセットオプション: リファレンス*
- *SAS 出力形式と入力形式: リファレンス*
- *SAS 関数と CALL ルーチン: リファレンス*
- *SAS 言語リファレンス: 解説編*
- *SAS マクロ言語: リファレンス*
- *SAS ファイルの移動とアクセス*
- *SAS 各国語サポート(NLS): リファレンスガイド*
- *SAS Output Delivery System: ユーザーガイド*
- *SAS ステートメント: リファレンス*
- *SAS システムオプション: リファレンス*

SAS 刊行物の総一覧については、support.sas.com/bookstore にてご確認ください。必要な書籍についてのご質問は、下記までお寄せください。

SAS Books
SAS Campus Drive
Cary, NC 27513-2414
電話: 1-800-727-3228
ファクシミリ: 1-919-677-8166
メール: sasbook@sas.com
Web アドレス: support.sas.com/bookstore

1 部

UNIX 版 SAS を実行する

1 章

入門ガイド: UNIX 版 SAS

UNIX 環境で SAS セッションを開始する

SAS を起動する

SAS セッションは`!SASROOT` ディレクトリ内のリンクを使用して起動されます。(`!SASROOT` ディレクトリ はサイトまたはコンピュータでの SAS のインストールディレクトリまたはフォルダの名前を表す用語です)。UNIX 管理者は操作環境のためのコマンドリストにこのリンクを追加することができます。`!SASROOT` ディレクトリの詳細については、“!SASROOT ディレクトリについて” (445 ページ)を参照してください。

サイトにて SAS を起動するコマンドに関してはシステム管理者にお問い合わせください。多くのサイトでは SAS を起動するコマンドは `sas` です。しかしお使いのサイトでは SAS インストールプロセス中に別のコマンドが定義されることもあります。このドキュメントでは SAS は `sas` コマンドにて起動されるものと仮定します。

注: SAS セッションを起動する前に、SAS セッションへの割り込みや終了するさまざまな方法を確認してください。詳細については、“UNIX 環境で SAS セッションを終了または割り込む” (26 ページ)を参照してください。さらに、SAS セッションを停止することができない場合は、システム管理者に連絡してください。

SAS 起動スクリプト

SAS は `!SASROOT/bin` ディレクトリにあるスクリプトにより起動されます。SAS 起動スクリプトはインストールされている各言語にあわせて作成されています。起動スクリプトはインストールされた言語の言語コードを使った名前を付けられています。たとえば、`sas_en` は英語版の SAS を起動します。すべての場所に全言語がインストールされています。

SAS 設定の詳細については、UNIX 環境のインストールドキュメントを参照してください。

SAS 構成ファイル

SAS はインストールされている各言語のための個別の構成ファイルを作成します。特定の言語の構成ファイル各言語に対して **!SASROOT/nls/<language>/sasv9.cfg** のフォームを持っています。言語とは独立したもう一つの構成ファイルは **!SASROOT/sasv9.cfg** です。**!SASROOT/nls/<language>/**にあるマスタ構成ファイルが、**!SASROOT/nls/<language>/**にある言語固有のファイルに加えて、すべての言語に使用されます。必要に応じてこれらの構成ファイルを変更することができます。SAS 構成ファイルをカスタマイズする方法についての詳細は、“構成ファイルと Autoexec ファイルを使用し、SAS セッションをカスタマイズする” (21 ページ)を参照してください。

SAS コマンドの構文

SAS コマンドの一般のフォームを次に示します。

sas <*-option1 ...-option-n*> <*filename*>

sas –sysin *filename*

SAS コマンドにこれらの引数を使用することができます。

-option1 ... -option-n
セッションや X コマンド行オプションを設定するための SAS システムオプションを定義します。詳細については、“UNIX 版に固有の SAS システムオプション” (360 ページ) および “X コマンド行オプション” (13 ページ)を参照してください。オプションを省略した場合は(コマンド行または構成ファイルにて)、SAS の(またはサイト特有の)デフォルトオプションが適用されます。

filename
実行する SAS プログラムを含むファイル名を指定します。SAS コマンドにてファイル名を指定することでバッチ SAS セッションを起動します。ファイル名を省略して対話形式セッションを開始します。

ファイル名がカレントディレクトリにない場合、フルパス名を指定します。フルパス名が与えられない場合、拡張子.sas と見なされます。

注: オプションが *filename* を認識しない場合、コマンドは実行されません。この場合、**-sysin *filename*** が必要です。

例: 対話式 SAS セッションの起動

SAS システムオプションを指定しないで対話式 SAS セッションを起動するには、次を入力します。

```
sas
```

実行モードはデフォルト設定によって異なります。詳細については、“UNIX 環境で SAS を実行する方法の選択” (7 ページ)を参照してください。

SAS の起動時に WORK および MEMSIZE システムオプションを指定するには、次のコマンドを入力します。

```
sas -work /saswork -memsize 4G
```

SAS が開始しない場合

SAS が開始しない理由はいくつかあります。よくある理由の一部を次に示します。

- 存在しない autoexec ファイルを指定した場合、SAS は開始しません。物理ファイルが存在しないことを示すエラーメッセージが SAS ログに表示されます。
- 構成ファイルが見つからない場合、SAS は開始しません。通常このエラーはインストールに問題があることを示します。
- Work ディレクトリが見つからない場合、SAS は開始しません。WORK システムオプションを使用して Work ディレクトリを指定できます。
- **`./sas –nodms –stimerr`** (stimer のスペルミス)などのスペルミスのある無効なオプションを指定した場合、SAS は開始しません。

SAS が開始しない場合、SAS ログは失敗を説明するエラーメッセージを記録します。しかし、SAS ログが初期化される前に SAS が発行したエラーメッセージはコンソールログに書き込まれます。エラー処理を効率的にするための追加として、エラー時には標準出力に加えて、情報が書き込まれます。

システムにパッチが正しく適用されていない場合、SAS は NLS 拡張エラーなどのエラーを発生します。これと他のタイプのエラーメッセージはインストールが検索のルールを正しく設定していなかったことを意味します。

UNIX では、STDOUT fileref がコンソールログの場所を指定します。

フォアグラウンド処理またはバックグラウンド処理で SAS を実行する

UNIX はマルチプロセスオペレーティングシステムで、同時に複数のプロセスを実行することができます。たとえば、1 つのプロセスをフォアグラウンドで実行させながら、3 つのプロセスをバックグラウンドで実行させることが可能です。

プロンプトを待つ間に、フォアグラウンドのプロセスが実行されます。つまり、現在のコマンドを実行している間に、追加のコマンドを実行させることはできません。コマンドを入力した後で、シェルがコマンドを実行するプロセスを開始します。システムがコマンドを実行させた後、シェルがプロンプトを表示し追加コマンドを入力することができます。フォアグラウンドプロセスとして SAS を実行している例を次に示します。

```
sas
```

フォアグラウンドにて実行することで、標準入力と出力にアクセスすることが可能です。

バックグラウンドのプロセスはシェルとは独立して実行されます。コマンドを入力した後で、シェルがコマンドを実行するプロセスを開始し、システムプロンプトを発行します。最初のコマンドの実行を待たずに、他のコマンドを入力したり、他のバックグラウンドプロセスを開始したりすることが可能です。バックグラウンドプロセスを実行するために使用されるコマンドの例を次に示します。

```
sas&
```

注: C シェルと Korn シェルは、フォアグラウンドで実行、バックグラウンドで実行、そして停止、の 3 つの内いずれか 1 つの状態にジョブを割り当てることが可能なコマンドを持っています。SAS を–nodms モードにて実行した場合、入力を待たずにプロセスが停止します。dms モードでは、シェルが標準出力と入力の設定を維持します。

UNIX 環境で SAS を実行する方法の選択

SAS ウィンドウ環境モード、対話型ラインモード、そしてバッチモードにて SAS を実行することが可能です。

- “ウィンドウ環境の SAS の起動” (8 ページ)
- “UNIX 環境の対話型ラインモード” (9 ページ)
- “UNIX 環境のバッチモード” (10 ページ)

サイトでのデフォルトのインターフェースまたは動作モードについて UNIX システム管理者に確認します。

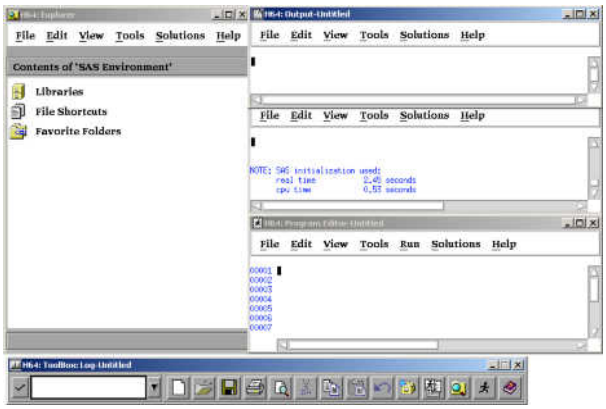
UNIX 環境の SAS ウィンドウ環境

SAS ウィンドウ環境について

SAS ウィンドウ

画面を通じて、キーボード、マウス、メニュー、アイコンを使用して SAS と対話します。ウィンドウ環境は **Explorer**、**Program Editor**、**Output**、**Log**、そして **Results** ウィンドウを含むさまざまなウィンドウ環境を含みます。次のウィンドウは **Explorer**、**Output**、**Log**、そして **Program Editor** を示します。**ツールボックス**ウィンドウも表示されます。

画面 1.1 *SAS ウィンドウ環境でのウィンドウ*



SAS セッションはデフォルトのウィンドウ環境インターフェースになります。(構成ファイルを使用してデフォルトを変更することが可能です。) ウィンドウ環境を使用したい場合は、SAS セッションをフォアグラウンドプロセスまたは SAS コマンド行にアンパサンド (&)を追加することでバックグラウンドプロセスとして起動することができます。これらの SAS コマンドの例については、“フォアグラウンド処理またはバックグラウンド処理で SAS を実行する” (6 ページ) を参照してください。

ウィンドウ環境の使用の詳細については、“SAS ウィンドウ環境の定義” (146 ページ) を参照してください。

注: X Window を使用していない場合、NODMS システムオプションを使用して SAS を対話型ラインモードにて起動させることが可能です。詳細については、"UNIX 環境の対話型ラインモード" (9 ページ)を参照してください。

Explorer ウィンドウについて

Explorer は、データセット、ライブラリ、メンバ、アプリケーション、そして出力などの表示や管理などの基本的な SAS ソフトウェアのタスクを管理するウィンドウ環境です。**SASExplorer** は次の実行が可能な中央のアクセスポイントです。

- グラフィックインターフェースを通した SAS データの操作
- **Program Editor**、**Output**、そして **Log** ウィンドウ (他のウィンドウも含む)へのアクセス
- **Results** ウィンドウにて SAS プロシジャ出力結果の表示
- SAS へのファイルのインポート

Program Editor、Output、Log ウィンドウについて

Program Editor、**Output**、そして **Log** ウィンドウによって、SAS プログラムを編集したり実行したり出力を表示したりすることが可能です。これらのウィンドウの詳細については、オンライン SAS ヘルプとドキュメントを参照してください。

ウィンドウ環境の SAS の起動

SAS セッションが開始するときに、どのウィンドウを開くかを次のコマンドを使用して指定することができます。

- DMS システムオプションを指定することで、**Program Editor**、**Output**、そして **Log** ウィンドウを開くことができます。

```
sas -dms
```

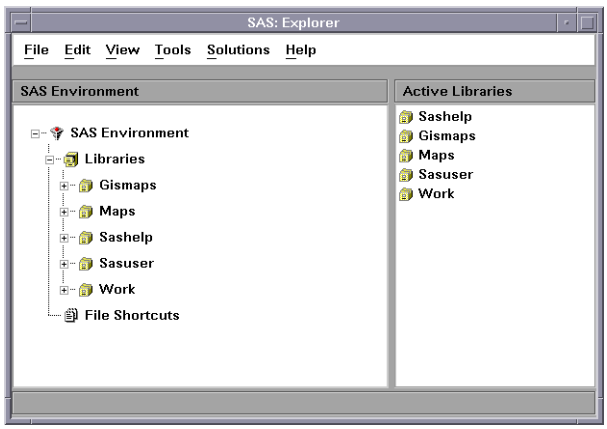
- DMSEXP システムオプションを指定することで、**Program Editor**、**Output**、**Log**、そして **Results** ウインドウ、さらに **Explorer** ウィンドウを開くことができます。

```
sas -dmsexp
```

- EXPLORER システムオプションを指定することで、**Explorer** ウィンドウのみを開くことが出来ます。

```
sas -explorer
```

画面 1.2 *SASExplorer ウィンドウ*



SAS を起動するデフォルトの指定は `sas -dmsexp` です。このコマンドは **Program Editor**、**Output**、**Log**、そして **Results** ウィンドウ、さらに **Explorer** ウィンドウを表示し

ます。**-dmsexp** オプションなしで SAS を起動する場合、**Explorer** ウィンドウは表示されません。

追加の SAS ウィンドウを開く Toolbox が開かれます。Toolbox の詳細については、"SAS ウィンドウ環境の操作" (146 ページ)を参照してください。

ウィンドウ環境の SAS の終了

SAS セッションを終了するには、コマンドウィンドウにて BYE または ENDSAS コマンドを入力するか、終了したい SAS セッションのメニューから **File** ⇨ **Exit** を選択します。

UNIX 環境の対話型ラインモード

対話型ラインモードについて

X Window を使用していない場合、NODMS システムオプションを使用して SAS を対話型ラインモードにて起動させることが可能です。

SAS から発行されたプロンプトに対して応答するために SAS ステートメントを 1 行ずつ入力します。SAS はターミナルから入力されるソースステートメントを逐次読み込みます。次のどれかが起こると、DATA と PROC ステップが実行します。

- RUN、QUIT、または DATALINES ステートメントが入力されます。
- 別の DATA または PROC ステートメントが入力されます。
- ENDSAS ステートメントが入力されます。

対話型ラインモードを使用するには、SAS をフォアグラウンドで実行する必要があります。

対話型ラインモードで SAS を起動する

対話型ラインモードセッションを起動するには、NODMS または NODMSEXP システムオプションにて SAS を起動します。

```
sas -nodms
sas -nodmsexp
```

デフォルトでは、各ステップが実行されるに従い、SASLog とプロシジャ出力(もしあれば)がウィンドウに表示されます。

SAS を対話型ラインモードにて起動したり、それにパラメータを割り当てたりすることが可能です。

```
sas -sysparm 'A B C'
```

A B C の値は SYSPARM マクロ変数に割り当てられています。–nodms オプションを使用して SAS をラインモードで起動する場合、プログラムエディタ内や SAS コマンドプロンプトから **progparm.sas** などのプログラム名を含めることができます。

SAS を起動した後、**1?** プロンプトが表示され、SAS ステートメントを入力することが可能です。各ステートメントを入力後、行番号プロンプトが表示されます。

対話型ラインモードでの SAS の終了

EOF キー、通常は CTRL-D(“コントロールキーの使用” (28 ページ) を参照)を押すか、ENDSAS ステートメントを発行してセッションを終了できます。

```
endsas;
```

すべての SAS ステートメントの実行終了後、セッションが終了します。

UNIX 環境のバッチモード

SAS のバッチモード実行について

SAS をバッチモードで実行するには、SAS 起動コマンドに SAS プログラム名を指定します。フォアグラウンドにてバッチモードで実行したり、SAS コマンドの終わりにアンパサンドを指定することでバックグラウンドで実行したり、**batch**、**at**、**nohup**、または **cron** などの UNIX コマンドを使ってバッチキューにアプリケーションをサブミットして実行したりすることが可能です。(詳細については、**batch**、**at**、**nohup**、**cron** の各コマンドの UNIX man ページを参照してください。) これらの UNIX コマンドのいずれかを使用してユーザーがアプリケーションを開始する場合、システムからユーザーがログオフすると、アプリケーションは実行を終了します。FSEDIT などの対話式プロシジャで始まるステートメントをアプリケーションが含む場合、フォアグラウンドでバッチアプリケーションを実行したり、–noterminal オプションを指定する必要があります。

バッチモードで SAS を起動する

SAS をバッチモードで起動するには、SAS コマンドにファイル名を指定する必要があります。たとえば、実行する SAS ステートメントを含むファイルが **weekly.sas** であり、NODATE および LINESIZE システムオプションを指定したい場合、次のコマンドを入力します。

```
sas weekly.sas -nodate -linesize 90
```

コマンドはフォアグラウンドにてプログラムを実行します。プログラムをバックグラウンドで実行したい場合、コマンドの終わりにアンパサンドを追加します。

```
sas weekly.sas -nodate -linesize 90 &
```

SAS はログとプロシジャ出力を含む .ログファイルと .lst ファイルをカレントディレクトリに作成します。

バッチキューにプログラムをサブミットする

プログラムをバッチキューにサブミットするには、**batch**、**at**、**nohup** または **cron** コマンドを使用します。たとえば、次のようにシェルプロンプトから **weekly.sas** を発行することが可能です。

```
$ at 2am
sas weekly.sas
<control-D>
warning: commands will be executed using /usr/bin/sh
job 8400.a at Wed Mar 16 02:00:00 2011
$
```

プログラムを実行するのに必要な SAS コマンド(たとえば **cmdfile.sh**)を含むファイルを作成する場合、次のコマンドをシェルプロンプトに入力します。

```
at 2am < cmdfile.sh
```

SAS はプログラムと同じ名前を持つファイルに出力を送ります。出力ファイルは拡張子 .lst を持ちます。ログファイルは .log の拡張子を持つファイルに書き込まれます。これらのファイルは両方ともカレントディレクトリに書き込まれます。バッチキューへのジョブのサブミットに関する詳細については、これらのコマンドの UNIX man ページを参照してください。出力の回付の詳細については、“出力の印刷と出力先指定” (90 ページ)を参照してください。

ファイルをバッチモードでサブミットする場合、256 バイトを超える行は切り捨てられます。この 切り捨てに関する明確なメッセージは SAS ログに書き込まれます。

注: FSEDIT プロシジャ、CATALOG プロシジャ、または REPORT プロシジャなどの対話式プロシジャを開始するステートメントがプログラムに含まれている場合、そのプログラムはフォアグラウンドプロセスとして実行するか、–noterminal オプションを使用する必要があります。

UNIX パイプを使用して外部ファイルのデータを書き出す

UNIX パイプを使用して外部ファイルから SAS プログラムへデータを書き込むことが可能です。たとえば、データが **mydata** という外部ファイルに存在し、**myprog.sas** という SAS プログラムが

```
infile stdin;
```

というステートメントを含む場合、このコマンドを発行して **myprog.sas** に外部データ **mydata** からデータを読ませます。

```
cat mydata | sas myprog.sas
```

外部ファイルの使用の詳細については、“外部ファイルとデバイスの使用” (68 ページ)を参照してください。SAS プログラムに外部ファイルからデータを読ませる別の方法については、“Bourne シェルと Korn シェルのファイルディスクリプタ” (78 ページ)を参照してください。

UNIX 環境でリモートホスト上の SAS の実行

リモートホストの SAS の実行について

SAS を対話型モードで起動した場合、SAS をローカルホストで実行するか、SAS をリモートホストで実行してワークステーションで実行中の X サーバーを通してセッションと対話することが可能です。サーバーは X Wndow System に必要な表示サービスを提供します。

ほとんどの場合、サーバー名はコンピューター名から得られます。たとえば、コンピュータの名前が **green** の場合、サーバーの名前は **green:0.0** となります。ほとんどの場合、X サーバーはユーザーのログイン時にはすでに実行されています。サーバーを手動で開始する必要がある場合、X Window System ソフトウェアに付随するドキュメントを参照してください。

リモートホスト上で SAS を実行するには、DISPLAY 環境変数を設定するか、**-display** X コマンド行オプションを指定するかのいずれかでどのディスプレイを使用するかを SAS に指示する必要があります。

リモートホストの SAS の実行ステップ

リモートホスト上で SAS を実行するには、SAS を起動する前に DISPLAY 環境変数を設定するか、SAS コマンド行オプションとして**-display x** を指定するかのいずれかでどのディスプレイを使用するかを SAS に指示する必要があります。そして次のステップに従います。

1. リモートホストで実行しているクライアントがサーバーに接続する権限を持っているか確認してください。ほとんどの X サーバーにおいて、権限はユーザーのホームディレクトリに存在する .Xauthority ファイルを使用して設定されます。さらに、**xhost** コマンドを使用して権限を取り除くことができます。**xhost** クライアントを使用してすべてのリモートホストにサーバーに接続する権利を与えるには、X サーバーを実行しているシステムのシステムプロンプトに次のコマンドを入力します。

   ```
   xhost +
   ```

   システムが **xhost** クライアントへのアクセスを設定しない場合、リモートアクセスに関する情報をシステムドキュメントから参照してください。

   権限の編集や表示の詳細については、UNIX man ページの xauth を参照してください。

2. リモートシステムへログオンするか、またはリモートシェルを使用します。
3. リモートホストで実行している X クライアントを表示するターゲットのサーバーを指定します。次の 2 つのうち 1 つの方法でサーバーを指定できます。

   a. DISPLAY 環境変数を設定します。Bourne シェルおよび Korn シェルでは、DISPLAY 変数を次のように設定できます。

      ```
      DISPLAY=green:0.0
      export DISPLAY
      ```

      Korn シェルではこれら 2 つのコマンドを組み合わせることができます。

      ```
      export DISPLAY=green:0.0
      ```

      C シェルでは、UNIX **setenv** コマンドを使用する必要があります。

      ```
      setenv DISPLAY green:0.0
      ```

      DISPLAY 変数は、そのシステムですべての X クライアントで使用されます。

      注: お使いのシステムのシェルを決定するには、UNIX コマンドプロンプトで **ps** とタイプするか、SHELL 環境変数の値をチェックします。

   b. DISPLAY システムオプションを使用します。例:

      ```
      sas -display green:0.0
      ```

      接続に問題がある場合、たとえば表示名のかわりに IP アドレスを使用して接続をやり直すことが可能です。

      ```
      -display 10.22.1.1:0.0
      ```

      注: このオプションは X Window System のコマンド行オプションであり SAS のものではありません。SAS 構成ファイルや SASV9_OPTIONS 環境変数でこのオプションを指定すると、他のインターフェースを実行中に問題を起こす場合があります。

SAS からの X サーバーへの接続を防ぐ

SAS が X サーバーに接続するのを防ぐには、DISPLAY 環境変数を設定せず、コマンド行にて **-noterminal** の SAS オプションを使用します。**-noterminal** オプションを指定することで SAS セッションを表示しません。バッチモードでグラフを作成するにはこのオプションを指定する必要があります。PROC IMPORT および PROC EXPORT を使用する場合もこのオプションを指定する必要があります。詳細については、*SAS/GRAPH: Reference* の"Running SAS/GRAPH Programs"を参照してください。

接続の問題のトラブルシューティング

SAS がディスプレイと接続できない場合、問題の性質を示すメッセージを印刷して終了します。受け取るメッセージの例を次に示します。

```
ERROR: The connection to the X display server could not be made.
Verify that the X display name is correct, and that you have
access authorization. See the online Help for more information
about connecting to an X display server.
```

SAS セッションを正しく起動したかどうかを確認します。**xhost** クライアント(**xhost +**を入力)または他の方法をしようして表示の権限を変更する必要があります。SAS のセッションをラインモードにて起動した場合、NODMS システムオプションを指定することができます。

SAS を起動できない場合、**xclock** などの別のアプリケーションを実行してください。アプリケーションを実行できない場合、UNIX システム管理者に連絡を取ってください。

X コマンド行オプション

X Window System オプションの指定法

SAS などのいくつかの X クライアントを起動した場合、X Window System に送られるコマンド行オプションを使用することができます。一般に、コマンド行の SAS オプションの後に X Window System オプションを指定します。

サポート対象の X コマンド行オプション

コマンドプロンプトから SAS セッションを起動した場合に、利用できる X コマンド行オプションを次の表で説明します。

-display *host*:*server*.*screen*
SAS セッションを表示したいターミナルの名前や IP アドレスを指定します。たとえば、IP アドレス 10.22.1.1:0.0 を持つ **wizard** が表示ノードである場合、

```
-display wizard:0.0
```

と入力するか

```
-display 10.22.1.1:0.0
```

-name *instance-name*
instance-name で始まる SAS リソースファイルの中のリソースを読み込みます。たとえば、**-name MYSAS** は次のような **MYSAS** で始まるリソースを読み込みます。

```
MYSAS.dmsfont: Cour14

MYSAS.defaultToolbox: True
```

-title *string*
SAS セッションウィンドウのタイトルを指定します。タイトルは最大 64 文字まで含まれます。ウィンドウタイトルは入力された文字サイズ、つまり小文字、大文字または混合にて表示されます。タイトルに複数の言葉を使用するには、その言葉を一重引用符または二重引用符で囲みます。たとえば、**-title MYSAS** は **Explorer** ウィンドウに **MYSAS:Explorer** というタイトルバーを表示します。

-xrm *string*
あらゆるデフォルトを上書きするリソースを指定します。たとえば、次のリソースは SAS を終了する時に、**確認**ダイアログボックスを表示しません。

```
-xrm 'SAS.confirmSASExit: False'
```

サポート対象外の X コマンド行オプション

SAS は次の X コマンド行オプションをサポートしていません。なぜならこれらの機能は SAS に適用されないか、または SAS リソースで提供されているからです。詳細については、“X リソースの概要” (170 ページ)を参照してください。

-geometry
ウィンドウ形状は **SAS.windowHeight**、**SAS.windowWidth**、**SAS.maxWindowHeight**、そして **SAS.maxWindowWidth** リソースにて指定されます。

-background、-bg
これらのオプションは無視されます。

-bordercolor、-bd
これらのオプションは無視されます。ウィンドウ枠の色の指定についての説明は、“ウィンドウ要素の色と属性の定義(CPARMS)” (206 ページ)を参照してください。

-borderwidth、-bw
これらのオプションは無視されます。ウィンドウの幅は SAS にて設定されます。

-foreground、-fg
これらのオプションは無視されます。

-font、-fn
SAS フォントは **SAS.DMSFont**、**SAS.DMSboldFont**、そして **SAS.DMSfontPattern** リソースにて指定されます。

-iconic
このオプションは無視されます。

-reverse、-rv、+rv
これらのオプションは無視されます。色の反転の指定に関する記述は、“ウィンドウ要素の色と属性の定義(CPARMS)” (206 ページ)を参照してください。

-selectionTimeout
タイムアウト長さは **SAS.selectTimeout** リソースにて指定されます。

-synchronous、+synchronous
XSYNC コマンドは SAS と X サーバーの間のシンクロ通信を切り替えます。

-xn1language
このオプションは無視されます。

SAS セッションからオペレーティングシステムコマンドの実行

同期または非同期タスクのどちらを実行するかの指定

非同期または同期にて SAS セッションから UNIX コマンドを実行することが可能です。コマンドを非同期タスクとして実行すると、コマンドは現在実行中のすべてのほかのタスクとは独立して実行されます。コマンドを非同期にて実行するには、SYSTASK ステートメントを使用する必要があります。コマンドを非同期にて実行する詳細については、“SYSTASK ステートメント: UNIX” (351 ページ) を参照してください。

一つまたは複数の UNIX コマンドを同期にて実行する場合、SAS セッションにて作業を続行する前に、これらのコマンドの実行が終了するのを待つ必要があります。UNIX コマンドを同期にて実行するには、CALL SYSTEM ルーチン、%SYSEXEC マクロプログラムステートメント、X ステートメント、そして X コマンドを使用することが可能です。CALL SYSTEM ルーチンは DATA ステップと共に使用することが可能です。%SYSEXEC マクロステートメントはマクロ定義の中で使用することが可能です。そして X ステートメントは DATA ステップとマクロ定義の外で使用することが可能です。あらゆる SAS コマンド行に X コマンドを挿入することが可能です。詳細については、“CALL SYSTEM ルーチン: UNIX” (271 ページ) および “UNIX 環境のマクロステートメント” (297 ページ)を参照してください。

1 つの UNIX コマンドの実行

シングルコマンド

一つのみの UNIX コマンドを実行するには、X コマンド、X ステートメント、CALL SYSTEM ルーチンまたは %SYSEXEC マクロステートメントを次のように挿入することが可能です。

X *コマンド*

X *コマンド*;

CALL SYSTEM ('*コマンド*');

%SYSEXEC *コマンド*;

注: %SYSEXEC マクロステートメントを使用する時、指定する UNIX コマンドにセミコロンを含む場合は、UNIX コマンドをマクロ引用機能に含む必要があります。引用機能の詳細については、*SAS マクロ言語: リファレンス*を参照してください。

例 1: X ステートメントを使用した UNIX コマンドの実行

X ステートメントを使用して次のように `ls`UNIX コマンド(子シェル内にて) 実行することが可能です。

```
x ls -l;
```

例 2: CALL SYSTEM ルーチンを使用した UNIX コマンドの実行

DATA ステップ内で、CALL SYSTEM ルーチンを使用して `cd` コマンドを実行し、SAS セッションのカレントディレクトリを変更できます。

```
data _null_;
call system ('cd /users/smith/report');
run;
```

これで、その SAS セッションの間は、相対(部分)ファイル名の検索は、**/users/smith/report** ディレクトリから始められます。セッションを終了すると、SAS セッションを開始したときのディレクトリがカレントディレクトリとなります。

CALL SYSTEM ルーチンの詳細については、"CALL SYSTEM ルーチン: UNIX" (271 ページ)を参照してください。

SAS による 1 つの UNIX コマンドのプロセス

一つのみのコマンドを指定すると、SAS はそのコマンドが **cd**、**pwd**、**setenv** または **umask** かどうかを確認し、もしそうであればこれらのコマンドと同等の SAS コマンドを実行します。SAS の **cd** および **pwd** コマンドは相対する Bourne シェルのコマンドと同等です。SAS の **setenv** コマンドは C シェルの同名コマンドと同等です。SAS の **umask** コマンドは Bourne、Korn、そして C シェルにてサポートされた数字モードの **umask** コマンドと同等です。現在の SAS セッションの環境に影響を与えるため、これら 4 つのコマンドは SAS に組み込まれています。SAS ソフトウェアにて実行されると、SAS 環境、そして SAS セッションにて開始されたシェルプログラムの環境のみに影響を与えます。SAS セッションにて開始したプログラムシェルの環境には影響を与えません。

コマンドが **cd**、**pwd** または **setenv** でない場合、SAS は指定したコマンドを実行するシェルを開始します。使用されるシェルは SHELL 環境変数によります。コマンドが **umask** で、*mask* を指定していない場合、SAS は現在の SAS セッションが開始したシェルにそのコマンドを送ります。**unmask** コマンドの詳細については、"SAS セッションのファイルの権限の変更" (17 ページ)を参照してください。

いくつかの UNIX コマンドの実行

UNIX コマンドの実行

いくつかの UNIX コマンドを実行するには、X コマンド、X ステートメント、CALL SYSTEM ルーチン、そして%SYSEXEC マクロステートメントを使用することが可能です。

X '*command-1*;...*command-n*'

X '*command-1*;...*command-n*';

CALL SYSTEM ('*command-1*;...*command-n*');

%SYSEXEC *quoting-function*(*command-1*;...*command-n*);

各 UNIX コマンドをセミコロン(;)で分けます。

注: %SYSEXEC マクロステートメントを使用していくつかの UNIX コマンドを実行する場合、コマンドリストはセミコロンをセパレータとして使用するためマクロ引用機能に UNIX コマンドの文字列を含める必要があります。引用機能の詳細については、*SAS マクロ言語: リファレンス*を参照してください。

例: %SYSEXEC マクロを使用したいくつかのコマンドの実行

次のコードは **pwdls** と呼ばれるマクロを定義し実行します。このマクロは **pwd** および **ls -l**UNIX コマンドを実行します。

```
%macro pwdls;
%sysexec %str(pwd;ls -l);
%mend pwdls;
%pwdls;
```

この例では**%str** をマクロ引用機能として使用しています。

SAS でのいくつかの UNIX コマンドのプロセス

一つ以上の UNIX コマンドを指定した場合 (つまりコマンドリストがセミコロンにて分けられている)、SAS はすべてのリストをシェルに送り、**cd**、**pwd**、**setenv** または **umask** コマンドなどをチェックしません。なぜならコマンドを自ら指定したときにチェックを行うからです (セミコロン無しで)。

SAS による **cd**、**pwd**、**setenv** または **umask** コマンドのプロセスの詳細は、"SAS による 1 つの UNIX コマンドのプロセス" (16 ページ)を参照してください。

SAS セッションのファイルの権限の変更

起動時に、SAS セッションは親シェルからファイルの権限を受け継ぎます。作成するすべてのファイルに、これらの権限が受け継がれます。SAS 内のファイル権限を変更したり削除する場合、X ステートメントで次のコマンドを発行します。**umask umask** コマンドは新しい"マスク"をファイルに適用します。つまり、作成する新しいファイルすべてに新しいファイル権限が設定されます。このように、**umask** コマンドは新しいファイルや現在のプロセスのディレクトリへのアクセスを制限することでファイルセキュリティを与えます。

umask のデフォルト値はさまざまです。Secure Linux などのシステムでは強制アクセス制御(MAC)を使用しているため、使用可能な Secure Linux の有無にかかわらず **umask** デフォルトは同じです。デフォルトとして 022 を使用しているシステムもあります。システム管理者は独自のデフォルト値を設定することが可能で、独自の.kshrc、.cshrc または.profile ファイルのデフォルトをチェックしたり変更したりすることが可能です。これらの値はすべてのシェルにて実行されるすべての子プロセスに影響します。現在の SAS セッション中に作成されるそれ以降のファイルは、指定した権限を受け継ぎます。与えられたマスクの元で作成されたファイルの権限は 8 進法表現で計算されます。

注: マスクの値は数字または記号です。このコマンドの詳細については、**umask** の UNIX man ページを参照してください。

さらに、FILE または FILENAME ステートメントで PERMISSION=オプションを使用すると、個々の出力ファイルの権限を制御できます。詳細については、"FILE ステートメント: UNIX" (328 ページ)または"FILENAME ステートメント: UNIX" (331 ページ)を参照してください。

バッチモードで X ステートメントを実行する

SAS プログラムがバッチモードで実行され、オペレーティングシステムがジョブコントロールをサポートしている場合、プログラム内の X ステートメントがターミナルからの入力を必要とするときに、プログラムは中断されます。

at または **batch** コマンドをサブミットして SAS プログラムをバッチキューから実行させる場合、SAS は X ステートメントを次のようにプロセスします。

- X ステートメントがコマンドを指定しない場合、SAS はステートメントを無視します。
- X ステートメントでの UNIX コマンドが入力を求める場合、プログラム終了(標準入力を**/dev/null** に設定する)を受け取ります。
- X ステートメントの UNIX コマンドが標準出力または標準エラーを書き込む場合、行き先を変更済みでない限りその出力をユーザーにメイルします。

SAS レジストリファイルのカスタマイズ

SAS レジストリファイルは SAS セッションに関する情報を保存します。SAS レジストリは SAS の構成データのための中央保存場所です。次のリストはレジストリに保存されているいくつかのデータを示します。

- スタートアップ時に SAS が割り当てるライブラリとファイルのショートカット これらのショートカットはパスワードなどのセキュリティ情報を含みます。
- 指定された使用するプリンタとその印刷設定
- さまざまな SAS 製品の構成データ

SASUSER レジストリファイル (regstry.sas7bitm と呼ばれる)はユーザーのデフォルト値を含みます。これらのレジストリ入力は SAS レジストリエディタまたは PROC REGISTRY を使用してカスタマイズすることが可能です。詳細については、*SAS 言語リファレンス: 解説編*の“The SAS Registry”を参照してください。

注意:

経験のあるユーザー向け 一般にレジストリのカスタマイズは経験のある SAS ユーザーおよびシステム管理者のみが実行します。

システムオプションを使用し、SAS セッションをカスタマイズする

SAS セッションのカスタマイズ法

SAS 環境をいくつかの方法でカスタマイズすることが可能です。一つの方法は SAS システムオプションを通す方法です。SAS セッションのカスタマイズの他の方法の詳細については、“X 環境における SAS カスタマイズの概要” (170 ページ)を参照してください。

SAS システムオプションの指定法

SAS オプションは一つまたはそれ以上の方法にて指定することが可能です。

- 構成ファイル
- SASV9_OPTIONS 環境変数
- SAS コマンド
- OPTIONS ステートメントにて (SAS プログラムまたは autoexec ファイルにて) (autoexec ファイルは SAS が起動するときに自動的に実行される SAS ステートメントを含みます。autoexec ファイルを使って SAS システムオプションを指定したり、頻繁に使用されるデータソースへライブラリ参照名およびファイル参照名を割り当てたりすることが可能です)。
- **System Options** ウィンドウ

CENTER および NOCENTER などのような SAS の初期化に影響を与えないオプションを指定したり変更したりすることがいつでも可能です。

いくつかのオプションは構成ファイル、SASV9_OPTIONS 変数または SAS コマンドのみにて指定することが可能です。これらのオプションは SAS がオペレーティングシステムとハードウェアとインターフェースをどのように初期化するかを決定します。それらは構成オプションとも呼ばれます。SAS セッションを開始した後、これらのオプションを変更することはできません。通常、構成ファイルにて頻繁に変更しないオプションを指定します。1 つのジョブのためにオプションを変更する必要がある場合は、SAS コマンドにてその変更を指定します。

システムオプションのデフォルト値を無効にする

SAS システムオプションのデフォルト値は多くの SAS プログラムにとって適切です。しかし、次の方法で一つまたはそれ以上のデフォルト値をオーバーライドすることが可能です。

構成ファイル
: 現在の構成ファイルの変更 (“SAS 構成ファイルの処理の優先順位” (24 ページ) を参照)または新しい構成ファイルの作成をします。各ファイルの前にハイフンを置くことでファイルの SAS システムオプションを指定します。ON または OFF オプションのためには、適切な設定と一致するキーワードをリストするだけです。値を受け入れるオプションのためには、オプション値の後にそのオプションを指定するキーワードを続くオプションをリストします。すべての SAS システムオプションは構成ファイルに表示させることができます。

  たとえば、構成ファイルは次のオプション指定を含むことがあります。

```
-nocenter

-verbose

-linesize 64
```

SASV9_OPTIONS 環境変数
: SAS を起動する前に SASV9_OPTIONS 環境変数にて SAS システムオプションを指定します。“UNIX 環境で環境変数を定義する” (439 ページ)を参照してください。

  SASV9_OPTIONS 環境変数にて指定する設定は変数が定義されたときに開始する SAS セッションに影響を与えます。

  たとえば、Korn シェルでは次のように使用します。

```
export SASV9_OPTIONS='-fullstimer -nodate'
```

SAS コマンド
: SAS コマンドにて SAS システムオプションを指定します。各オプションの前にハイフンを置きます。

```
sas -option1 -option2...
```

  ON または OFF オプションのためには、適切な設定と一致するキーワードをリストするだけです。値を受け入れるオプションのためには、オプション値の後にそのオプションを指定するキーワードを続くオプションをリストします。次は、その例です。

```
sas -nodate -work mywork
```

  SAS コマンドで指定した設定は、SAS セッションの期間にて有効であるか、セッション内で変更できるオプションの場合は変更がされるまで有効です。すべてのオプションは SAS コマンドにて指定することができます。

SAS セッション内の OPTIONS ステートメント
: SAS セッション中のどの時点でも OPTIONS ステートメントで SAS システムオプションを指定します。このオプションは SAS セッションの間に対して、またはそれが変更されるまで有効です。OPTIONS ステートメントの中でオプションを指定する場

合、名前の前にハイフン (-)を付けません。オプションが引数を持つ場合、オプション名の後に = を使用します。次は、その例です。

```
options nodate linesize=72;
options editcmd='/usr/bin/xterm -e vi';
```

OPTIONS ステートメントの詳細については、“OPTIONS Statement” (*SAS Statements: Reference*)を参照してください。一部のオプションが SAS コマンドで指定できない場合もあります。

autoexec ファイルでの OPTION ステートメント
: autoexec ファイル内の OPTIONS ステートメントにて SAS システムオプションを指定します。autoexec ファイルは、SAS が起動されたときに自動的に実行される SAS ステートメントを含みます。autoexec ファイルを使って SAS システムオプションを指定したり、頻繁に使用されるデータソースへライブラリ参照名およびファイル参照名を割り当てたりすることが可能です)。たとえば、autoexec ファイルは次のステートメントを含むこともあります。

```
options nodate pagesize=80;
filename rpt '/users/myid/data/report';
```

System Options ウィンドウ
: **System Options** ウィンドウにて SAS システムオプションを変更します。

一般に、OPTIONS ステートメントまたは **System Options** ウィンドウにて指定されるファイル名やパス名に引用符を使って囲みます。そうでない場合は引用符マークを使用しないでください。例外は個別のオプションにて説明されています。指定するファイル名やパス名を短くするには、表 2.3 (52 ページ)でリストされている省略語を使用することができます。

システムオプションの値にスペースが含まれる場合

システムオプションの値にスペースが含まれる場合は、コマンド行や構成ファイル内ではその値を引用符で囲む必要があります。次に、正しい構文例を示します。

```
-bufsize '3 k';

-bottommargin '2 in';
```

システムオプションの値にスペースが含まれない場合は、その値を引用符で囲む必要はありません。

```
-bufsize 3k;

-bottommargin 2in;
```

複数指定されているシステムオプションの処理方法

SAS コマンド、構成ファイルまたは SASV9_OPTIONS 環境変数などで同じシステムオプションが 1 回以上設定されている場合は、最新の設定の値のみを SAS は使用します。他の設定は無視されます。たとえば、次の SAS コマンドで DMS オプションは無視されます。

```
sas -dms -nodms
```

次の構成ファイルでも DMS オプションは無視されます。

```
-dms
-linesize 80
-nodms
```

デフォルトでは、HELPLOC、MAPS、MSG、SAMPLOC、SASAUTOS、SASHELP の各システムオプションを 1 回以上指定した場合、最新の指定値を SAS は使用します。これらのオプションのいずれかですでに指定されたパス名に更なるパス名を追加する場合は、APPEND または INSERT システムオプションを使用する必要があります。詳細については、“APPEND システムオプション: UNIX” (365 ページ) および “INSERT システムオプション: UNIX” (391 ページ)を参照してください。

複数の場所で指定されているシステムオプションの処理方法

複数の場所でのシステムオプション設定

同じシステムオプションが複数個所で設定されている場合は、最新の設定の値のみを SAS は使用します。次の場所が優先順にリストアップされています。たとえば、**システムオプション**ウィンドウや OPTIONS ステートメントで行われた設定はそれ以外の設定を上書きします。ただし、SASV9_OPTIONS 環境変数を使用してシステムオプションを設定した場合、この設定は構成ファイルにある同一システムオプションの設定のみに優先します。

システムオプション処理時の優先順序

システムオプションを処理する場合の優先順序は次のようになります。

1. **System Options** ウィンドウまたは OPTIONS ステートメント(SAS セッションまたはジョブからの)
2. OPTIONS ステートメントを含む autoexec ファイル(SAS 初期化後)。(autoexec ファイルは、SAS が起動されたときに自動的に実行される SAS ステートメントを含みます。autoexec ファイルを使って SAS システムオプションを指定したり、頻繁に使用されるデータソースへライブラリ参照名およびファイル参照名を割り当てたりすることが可能です)。
3. SAS コマンド
4. SASV9_OPTIONS 環境変数
5. 構成ファイル(SAS 初期化前)。詳細については、“SAS 構成ファイルの処理の優先順位” (24 ページ)を参照してください。

たとえば、構成ファイルが NOSTIMER を指定する場合、–FULLSTIMER を指定することで SAS コマンドの設定をオーバーライドすることが可能です。

デフォルトでは、HELPLOC、MAPS、MSG、SAMPLOC、SASAUTOS、SASHELP の各システムオプションを 1 回以上指定した場合、最新の指定値を SAS は使用します。これらのオプションのどれかにてすでに指定されたパス名に更なるパス名を追加する場合は、APPEND または INSERT システムオプションを使用して新しいパス名を追加します。詳細については、“APPEND システムオプション: UNIX” (365 ページ) および “INSERT システムオプション: UNIX” (391 ページ)を参照してください。

構成ファイルと Autoexec ファイルを使用し、SAS セッションをカスタマイズする

SAS セッションのカスタマイズ

SAS 環境をいくつかの方法でカスタマイズすることが可能です。起動時の SAS 環境をカスタマイズには、構成ファイルおよび autoexec ファイルを使用できます。ウィンドウ環

境を使用した SAS セッションのカスタマイズの詳細については、“X 環境における SAS カスタマイズの概要” (170 ページ)を参照してください。

構成および Autoexec ファイルについて

構成および Autoexec ファイルの定義

構成ファイルと autoexec ファイルを使用し、SAS セッションをカスタマイズできます。これらのファイルを使用してシステムオプションの指定をしたり、SAS セッションを開始するたびに SAS ステートメントを自動実行したりすることが可能です。SAS システムオプションは出力先、プログラム実行の効率、そして SAS ファイルおよびライブラリの属性などを含む SAS セッションの多くの側面を設定します。SAS システムオプションの詳細な記述については、*SAS システムオプション: リファレンス*を参照してください。

SAS 9.4 では、構成ファイルは通常 sasv9.cfg という名前で、autoexec ファイルは autoexec.sas という名前です。これらのファイルは通常 SAS がインストールされたディレクトリに存在します。デフォルトでは、このディレクトリは **!SASROOT** ディレクトリです。

カスタマイズした構成ファイルおよび autoexec ファイルをユーザーのホームディレクトリに持つことが可能です。その場合、ユーザーのセッションを開始する際、それらのファイルに指定されたカスタマイズされた設定を SAS は使用します。構成ファイルをプロセスする時に SAS が使用する優先順位の詳細については、“SAS 構成ファイルの処理の優先順位” (24 ページ)を参照してください。

SAS システムオプションは UNIX システム管理者により制限されているため、それらが管理者で一旦設定されると、ユーザーが変更することはできません。システムオプションは全体、グループ、そしてユーザーによって制限することが可能です。詳細については、**テクニカルサポート Web サイト**の UNIX 環境の構成ガイド、および “Restricted Options” (*SAS System Options: Reference* 1 章)を参照してください。

AUTOEXEC システムオプションの使用

AUTOEXEC システムオプションは autoexec ファイルを指定します。autoexec ファイルには、SAS の起動時や別の SAS プロセス開始時に自動的に実行される SAS ステートメントが含まれます。autoexec ファイルには、任意の SAS ステートメントを含めることができます。たとえば、SAS セッションで定期的にアクセスする SAS ライブラリの LIBNAME ステートメントを autoexec ファイルに含めることができます。

SAS は次の場所にて AUTOEXEC システムオプションを探します。SAS は最初に見つけた AUTOEXEC システムファイルを使用します。

- コマンド行
- SASV9_OPTIONS 環境変数
- 構成ファイル

AUTOEXEC と NOAUTOEXEC システムオプションのいずれも検出されない場合、SAS では次の 3 つのディレクトリ内をこの順序で autoexec ファイルを検索します。

1. カレントディレクトリ
2. ホームディレクトリ
3. **!SASROOT** ディレクトリ (詳細については、“!SASROOT ディレクトリ” (445 ページ)を参照してください)。

SAS は、最初に検出した autoexec ファイルを SAS セッションの初期化に使用します。ユーザーセッションの autoexec ファイルの内容を表示する必要がある場合は、SAS の起動時に ECHOAUTO システムオプションを使用します。

autoexec ファイルの挿入および付加

次のシステムオプションを AUTOEXEC システムオプションと使用することで、autoexec ファイル連結することが可能です。“INSERT システムオプション: UNIX” (391 ページ) および “APPEND システムオプション: UNIX” (365 ページ)。autoexec ファイルは常にテキストファイルです。ファイル名に埋め込み空白または特殊文字が含まれる場合、ファイル名を引用符で囲む必要があります。ファイル名に埋め込み空白または特殊文字が含まれない場合は、1 つ以上のファイル名を指定するときに引用符はオプションです。

次の構文を使用して、autoexec ファイルを結合できます。

```
-autoexec "(/path1/autoexec.sas /path2/autoexec.sas /path3/autoexec.sas)"
```

次の構文で、INSERT システムオプションとともに使用できます。

```
-insert autoexec "a.sas" -insert autoexec "b.sas"
```

次の構文で、APPEND システムオプションとともに使用できます。

```
-append autoexec "a.sas" -append autoexec "b.sas"
```

連結された autoexec リストのファイルのどれかが存在しない場合や開くことができない場合(たとえば読み込みアクセスの権限を持たない)、SAS セッションはエラーメッセージをログに発行します。SAS はリスト内のファイルを一切実行せずに終了します。最後の SAS 終了コードは 103 で、これはシステムスタートアップの失敗を意味します。

構成ファイルと自動実行ファイルの相違点

構成ファイルと autoexec ファイルの相違は次に示します。

- 構成ファイルには、SAS システムオプション設定のみを含められます。Autoexec ファイルには、任意の有効な SAS ステートメントを含められます。たとえば、OPTIONS ステートメントを含む autoexec ファイルにてさまざまなシステムオプションのデフォルト値を変更したり、LIBNAME および FILENAME ステートメントを含む autoexec ファイルにて SAS ライブラリや最も使用する外部ファイルのデフォルト値を変更することが可能です。
- 構成ファイルは SAS の初期化前に処理されるのに対し、autoexec ファイルは SAS 初期化のすぐ後、任意のソースステートメントが処理される前に、処理されます。autoexec ファイルの OPTIONS ステートメントは、SAS セッションの最初のステートメントとして OPTIONS ステートメントをサブミットするのと同等です。

構成ファイルの作成

構成ファイルを作成するには、次の手順に従います。

1. テキストエディタを使用して SAS システムオプションを UNIX ファイルに書き込みます。このファイルを sasv9.cfg または.sasv9.cfg ファイルとして保存します。(詳細については、“SAS 構成ファイルの処理の優先順位” (24 ページ)を参照してください。)
2. 一つまたはそれ以上のシステムオプションを各行に指定します。SAS コマンドにてシステムオプションを指定するのと同じ構文を使用します。しかし SAS コマンド自体を含めません。たとえば、構成ファイルは次の行を含むことがあります。

   ```
   -nocenter
   -verbose
   -linesize 64
   -work /users/myid/tmp
   ```

3. 構成ファイルの保存と終了

SAS 構成ファイルの処理の優先順位

SAS のデフォルト構成ファイルは **!SASROOT** ディレクトリに存在します。オンサイトの SAS 担当者がこの構成ファイルを編集することでサイトに最適なオプションを含むことができます。

一つまたはそれ以上の独自の構成ファイルを作成することも可能です。SAS はこれらのファイルから次の順番でオプション設定を読み込みます。

1. **!SASROOT** ディレクトリにある sasv9.cfg ("!SASROOT ディレクトリのコンテンツ" (445 ページ)を参照してください。)
2. **!SASROOT** ディレクトリにある sasv9_local.cfg ("!SASROOT ディレクトリのコンテンツ" (445 ページ)を参照してください。)
3. ホームディレクトリの.sasv9.cfg (先頭のピリオドに注意する)。
4. ホームディレクトリの sasv9.cfg
5. カレントディレクトリの sasv9.cfg
6. 制限されたあらゆる構成ファイル 制限された構成ファイルはサイト管理者によって設定されたシステムオプションを含み、それはユーザーが変更することはできません。オプションは全体、グループごとまたはユーザーごとに制限できます。制限された構成ファイルの詳細については、UNIX 環境の構成ガイドを参照してください。

今後の SAS リリースでは、これらのファイル名はそれぞれ変更されます。

各システムオプションに対して、SAS は最後に検出した設定を使用します。他の設定は無視されます。たとえば、WORKPERMS システムオプションが **!SASROOT** ディレクトリの sasv9.cfg およびカレントディレクトリの sasv9.cfg に指定された場合、SAS はカレントディレクトリの sasv9.cfg にて指定された値を使用します。

SAS が使用する構成ファイルの指定

SAS が使用する構成ファイルを指定すると、"SAS 構成ファイルの処理の優先順位" (24 ページ)でリストアップされた構成ファイルの検索をバイパスします。

注: SAS は存在する制限された構成ファイルをプロセスします。これらのファイルの設定は指定した構成ファイルの設定よりも優先されます。

SASV9_OPTIONS および SASV9_CONFIG を両方設定した場合、SAS は常に SASV9_OPTIONS を使用します。コマンド行にて–config を使用しない場合のみ、SASV9_CONFIG が使用されます。

構成ファイルを指定するには、次のどれかの手順を終了します。

- SAS コマンドにて CONFIG システムオプションにて構成ファイルを指定します。

  ```
  sas -config filename
  ```

- SASV9_OPTIONS 環境変数にて構成ファイルを指定します。"UNIX 環境で環境変数を定義する" (439 ページ)を参照してください。たとえば、Korn シェルでは次のように使用します。

  ```
  export SASV9_OPTIONS='-config filename'
  ```

- SASV9_CONFIG の環境変数を定義します。"UNIX 環境で環境変数を定義する" (439 ページ)を参照してください。たとえば、Korn シェルでは次のように使用します。

```
export SASV9_CONFIG=filename
```

filename は SAS システムオプションを含むファイルの名前です。

SASV9_OPTIONS または SASV9_CONFIG 環境変数にて構成ファイルを指定した場合、NOCONFIG を SAS コマンドで指定することにより SAS がこのファイルを使用するのを防ぐことができます。

SAS が SASV9_OPTIONS を見つけられない場合、次のメッセージが SAS ログに書き込まれます。

```
ERROR: Cannot open [/fullpath/filename]: No such
file or directory.
```

UNIX 環境で SAS ジョブの完了ステータスの特定

SAS ジョブの終了時の終了ステータスは**$STATUS** と C シェルの場合に表示され、**$?** と Bourne および Korn シェルの場合に表示されます。値 0 は通常の終了を意味します。ABORT ステートメントを使用して終了ステータスコードに影響を与えることができます。ABORT ステートメントは 0 から 255 までの範囲であるオプショナル整数引数 *n* を取得します。

注: リターンコードの 0 から 6 とリターンコードの 977 より大きいものは SAS のための使用に予約されています。

次の表は終了ステータスコードの値をまとめています。

表 1.1 *終了ステータスコード値*

条件	終了ステータスコード
すべてのステップは正常に終了しました。	0
SAS System が警告を発行しました。	1
SAS System でエラーが発生しました。	2
ユーザーが ABORT ステートメントを発行しました。	3
ユーザーが ABORT RETURN ステートメントを発行しました。	4
ユーザーが ABORT ABEND ステートメントを発行しました。	5
深刻なエラーにより SAS が初期化できませんでした。	6
ユーザーが ABORT RETURN である- n ステートメントを発行しました。	n
ユーザーが ABORT ABEND である- n ステートメントを発行しました。	n

コマンド行にて ERRORABEND SAS システムオプションを指定し、さらにジョブがエラーとなった場合、終了ステータスコードは 5 となります。

UNIX 終了ステータスコードは 0 から 255 の範囲です。255 より大きい数字はコードが符号付のバイトであると認識するため、予想通りの印刷を行わない可能性があります。

UNIX 環境で SAS セッションを終了または割り込む

SAS の終了法

次の方法のどれか 1 つを使用して SAS セッションを終了します。

- SAS をウィンドウ環境にて使用している場合、**File** ⇨ **Exit** を選択します。
- **`endsas;`**を使用します。
- SAS をウィンドウ環境にて使用している場合、ツールボックスに **`BYE`** と入力します。
- EOF コマンドのコントロールキーシーケンスが CTRL+D であり、SAS を対話型ラインモードで使用している場合は、CTRL+D を使用します。

SAS への割り込みまたは終了の方法

SAS に割り込むまたは終了する

既存の SAS の方法に加えて、SAS は SAS セッションに割り込んだり終了したりする方法を提供します。"SAS の終了法" (26 ページ)にてリストアップされた方法のうち 1 つによって SAS を終了しようとする前に、SAS はこれらの方法を使用することをお勧めします。

次の方法にて SAS に割り込んだり終了したりすることが可能です。

- 割り込み(interrupt)または終了(quit)コントロールキーを押します。割り込みはダイアログボックスを表示しますが、終了は強制的にシャットダウンします。終了(quit)コントロールキーの使用は推奨しません。
- **SAS: Session Management** ウィンドウ
- UNIX**`kill`** コマンドを入力します。存在するすべての他の SAS 終了の方法に失敗した場合にこのコマンドを使用します。デフォルトでは、kill コマンドは **`kill -15`**(SIGTERM)です。

  実行中の SAS プロセスに対して **`kill -9`**UNIX コマンドを使用すると、書き込みや更新アクセスにて開かれているデータセットを破損する可能性があります。

SAS プロセスに割り込む

SAS プロセスに割り込むために使用する方法は SAS の起動方法により異なります。

- SAS を対話型ラインモードまたはフォアグラウンドのバッチモードにて実行している場合、次のどれかの方法にて SAS に割り込むことが可能です。
  - SAS を起動したシェルで割り込むために設定されたコントロールキーシーケンスを押します。ほとんどの場合、このコントロールキーシーケンスは CTRL+C です。環境に対する適切なコントロールキーシーケンスを決定するには、**`stty`** コマンドの man ページを参照してください。
  - **`kill`** コマンドで**`-SIGINT`** オプションを使用します。詳細については、"UNIX の kill コマンドの使用" (29 ページ)を参照してください。

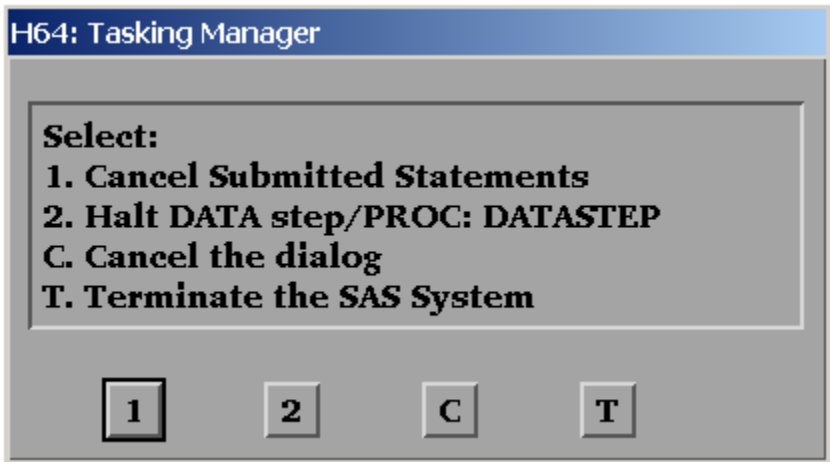
- フォアグラウンドにて SAS ウィンドウ環境を実行させている場合、**SAS: Session Management** ウィンドウの **Interrupt** をクリックします。

  注: -DMS または-DMSEXP オプションにて SAS を起動することによって、SAS Session Manager にアクセスすることが可能です。メニューから **SAS: Session Management** を選択します。

- SAS をバッチモードで実行している場合、SAS プロセスに割り込むにはコントロールキーを使用する必要があります。**SAS: Session Management** ウィンドウは利用できません。

interrupt シグナルはスーパーバイザに割り込みの処理リクエストを送ります。コードでの安全なポイントに到達するまで interrupt シグナルは処理されません。安全なポイントとは interrupt ハンドラが安全に実行するのを可能にするポイントです。スーパーバイザは可能な限り迅速に、ユーザーがどのようなタイプの割り込みアクションを行いたいかをたずねるプロンプトまたはウィンドウを表示して応答します。この間、DATA ステップまたは PROC ステップの通常プロセスは中断されます。

たとえば、DATA ステップまたは PROC ステップに割り込む場合、次に示すものと似たような **Tasking Manager** ウィンドウが表示されます。



次の表はウィンドウでの各オプションを説明します。

表 1.2 *Tasking Manager ウィンドウでのオプション*

オプション	説明	このオプションについて
1	サブミットされたステートメントのキャンセル	このオプションを選択することで現在の DATA ステップまたは PROC ステップを終了します。実行待ちの未処理のソースコードはシステムからクリアされます。対話型ラインモードでは、コマンドプロンプトに戻ります。

オプション	説明	このオプションについて
2	DATA ステップ/PROC の停止: DATASTEP	このオプションを選択すると、DATA ステップまたは PROC ステップに interrupt シグナル信号が送信されます。デフォルトの動作では、DATA ステップまたは PROC ステップが終了し、次のステートメントを実行します。 プロシジャは自らのハンドラに割り込みを処理するように指定する場合があります。この場合、プロシジャは追加の入力を必要とする場合があります。たとえば、SAS webAF は PROC SQL とは異なる割り込みメニューを持ちます。 *注:* リレーショナルデータベースを使用している場合、使用しているリレーショナルデータベースによりますが、interrupt シグナルの処理の方法が異なる場合があります。
C	ダイアログのキャンセル	このオプションを選択すると割り込みをキャンセルし、通常プロセスに戻ります。
T	SAS System の終了	このオプションを選択することで DATA ステップまたは PROC ステップを終了します。実行待ちの未処理のソースコードはシステムからクリアされます。SAS は可能な限り正常に終了します。

SAS プロセスの終了

フォアグラウンドにて SAS ウィンドウ環境を実行させている場合、**SAS: Session Management** ウィンドウの **Terminate** をクリックします。バックグラウンドにて SAS プロセスを対話型ラインモードで実行している場合、コントロールキー使用するか、**kill** コマンドを使用して SAS プロセスを終了します。

SAS: Session Management ウィンドウの **Terminate** をクリックすると、セッションを終了することを確認するダイアログボックスが表示されます。**OK** をクリックすると、SAS セッションと現在実行中のクエリは終了します。**Cancel** をクリックすると、SAS セッションに戻ります。

コントロールキーの使用

コントロールキーでは、割り込み(interrupt)や終了(quit)のキーシーケンスを押すことによりセッションを割り込んだり終了することが可能です。しかし、コントロールキーは SAS プログラムがバックグラウンドにて対話型ラインモードまたはバッチモードにて実行されているときのみに使用することが可能です。コントロールキーを使用してバックグラウンドジョブを停止することはできません。

注: **batch**、**at**、**nohup** または **cron** コマンドにてサブミットしたバッチジョブをコントロールキーを使用して停止することはできません。

コントロールキーはシステムによって異なるため、UNIX の **stty** コマンドを発行してどのキーがどのシグナルを発行するかを確認します。**stty** コマンドは UNIX 動作環境内で大きく異なるため、このコマンドを使用する前に **stty** の UNIX man ページを確認してください。通常は、これらのコマンド形式のいずれかで現在のターミナル設定すべてが出力されます。

```
stty
stty -a
stty everything
```

出力は次に示すものと似た行を含みます。

```
intr = ^C; quit = ^\; erase = ^H;
```

```
kill = ^U; eof = ^D; eol = ^@
```

キャレット(^)は CTRL キーを意味します。この例では、CTRL+C は割り込み(interrupt)キー、CTRL+\は終了(quit)キーです。終了では、より強制的に終了されるため、データの破損を生じる可能性があります。–SIGTERM を使用することで、SAS はシステムを正しく終了します。

SAS Session Manager の使用

ウィンドウ環境で SAS を起動する場合、SAS Session Manager を使用して SAS セッションに割り込んだり終了したりすることが可能です。SAS を開始すると、SAS Session Manager のウィンドウは自動的に最小化します。SAS セッションに割り込んだり終了したりするには、**SAS: Session Management** ウィンドウを開き、**Interrupt** または **Terminate** をクリックします。

SAS セッションを終了したときに非同期の SAS/CONNECT タスクが実行されている場合は、これらのタスクは終了され警告メッセージは表示されません。一般に、ファイルメニューまたはツールボックスから終了する方法が適切です。

注: **Interrupt** をクリックするのは **kill** コマンドで**-SIGINT** オプションを指定するのと同等です。**Terminate** をクリックするのは **kill** コマンドで**-SIGTERM** オプションを指定するのと同等です。

SAS Session Manager の詳細については、“UNIX 環境の SAS Session Manager (motifxsassm)” (149 ページ)を参照してください。

UNIX の kill コマンドの使用

注: 他のすべての方法を試した後でのみ、**kill** コマンドを使用して SAS セッションを終了します。

kill コマンドは、指定に基づき、interrupt シグナルまたは terminate シグナルを SAS に送ります。**kill** コマンドを使って、あらゆるモードで実行している SAS セッションへの割り込みまたは終了することが可能です。**kill** コマンドは SAS セッション内から発行することはできません。別のターミナルまたは別のウィンドウ(お使いのターミナルがその操作を許可する場合)からこのコマンドを発行する必要があります。

kill コマンドの形式は次となります。

kill <*-signal-name*> *pid*

interrupt シグナルを送信するには、**-SIGINT** を指定します。terminate シグナルを送信するには、**-SIGTERM** を指定します。**ps** コマンドとそのオプションを使用して、割り込みまたは終了したい SAS セッションの pid (プロセス識別番号)を特定します。

ps コマンド使用の結果は動作環境により異なります。**ps** コマンドとそのオプションに特定の情報については、お使いの動作環境での UNIX man ページを参照してください。1 つ以上の SAS プロセスを実行している場合、追加するオプションによって、どのプロセスを終了する必要があるかを特定する際に有益です。さらにサーバー (metadata, OLAP など)はプロセス識別番号を起動ディレクトリに残します。**kill** コマンドにてこの番号を使用することができます。

次の表はいくつかの重要な `kill` シグナルを示します。

表 1.3 *主要な kill シグナルの説明*

シグナル	オプション	説明
0	SIGNULL	プロセス識別子へのアクセスを確認します。
1	SIGHUP	SAS を終了させます。
2	SIGINT	SAS にセッションに割り込ませます。SIGINT は SIGQUIT と非常によく似ています。
3	SIGQUIT	SIGTERM よりも強制的にシャットダウンします。コアダンプの発生はしません。
9	SIGKILL	SAS を終了します。SAS 終了のためのすべての方法に失敗した後でのみ、このオプションを使用します。SIGKILL の使用はデータ破損を引き起こします。
15	SIGTERM	SAS を終了させます。

詳細については、`ps` および `kill` コマンドの UNIX man ページを参照してください。

コンソールログ(STDOUT)のメッセージ

SAS ログが利用できない時に、SAS でエラーや警告が発生した場合、SAS が発生するあらゆるメッセージはコンソールログに書き込まれます。通常、SAS ログは SAS 開始の初めと SAS 終了の終わりには利用できません。

-STDIO オプションを使用している場合、ログは stderr に表示され、リストは stdout に表示されます。

SAS Server として実行されているプロセスを終了する

SAS Server として実行しているプロセスを終了する必要がある場合、次のどれかの方法を使用します。

- SAS Metadata Server を使用している場合、SAS Management Console を使用してプロセスを終了します。
- 別の SAS Server を使用している場合、サーバーにインストール済みの UNIX スクリプトを使用してプロセスを終了します。これらのスクリプトを使用してサーバーを起動(または再起動)したり、サーバーがすでに実行されているかどうかを確認することが可能です。これらのスクリプトの詳細については、サイト管理者に連絡してください。

注: サーバーが UNIX スクリプトに応答しない場合、`kill` コマンドを使用してサーバープロセスを終了することができます。詳細については、“UNIX の kill コマンドの使用” (29 ページ)を参照してください。

SAS プロセスとその DBMS プロセスに割り込む

注意:

SAS プロセスの割り込みや DBMS プロセスの割り込みは DBMS にて実行しているすべてのジョブを終了する可能性があります。 SAS または DBMS プロセスの割り込みは、特別な処理です。クエリを構築する際には注意を払ってください。たとえば、SAS が SQL を RDBMS に送ると、SAS では SQL ステートメントを制御できなくなるため、このステートメントに割り込むことはできません。ステートメントは RDBMS にて実行されています。

SAS プロセスに割り込む、現在のクエリが終了する可能性があります。現在のクエリを使用して新しいデータセットを作成している場合、クエリが終了されてもデータセットは作成されます。現在のクエリを使用してデータセットを上書きする場合、クエリが終了させられるとデータセットは上書きされません。多くの場合、そのクエリが完了しなかったという警告は受け取れません。

注: このセクションでは、SAS プロセスとは連続したイベントのことを指します。オペレーティングシステムのプロセスのことではありません。SAS プロセスに割り込んだり終了した場合、オペレーティングシステムのプロセスは実行し続ける場合もあります。

多くの場合(UNIX 環境にて Oracle を使用するなど)、サーバー上のクエリに割り込んだり終了したりすると、次のプロセスが停止します。

- 現在の抽出プロセス。たとえば、WHERE 句を入れ忘れた SQL クエリで SAS に 1 億行を抽出中の場合、割り込みを発行すると、SAS プロセスと DBMS での抽出ステップが停止します。
- サーバーで進行中のクエリのプロセス。たとえば、結果を出力する前に長時間実行する複雑な抽出クエリを持つ場合があります。割り込みを発行することで SAS および DBMS プロセスが停止します。その結果、DBMS サーバーで実行している複雑な抽出クエリは割り込まれ終了されます。
- 更新、削除、挿入処理。たとえば、DBMS の更新、削除または多くの行の挿入を行っている場合 割り込みは SAS および DBMS プロセスを停止します。

2 章

SAS ファイルの使用

UNIX 環境の SAS ファイル、ライブラリ、エンジンについて

SAS ファイル

SAS ファイルについて

ユーザーのデータは、SAS ファイルや、他のソフトウェア製品(データベース管理システムなど)によってフォーマットされたファイルなど、異なる複数の種類のファイルに常駐しています。UNIX 版での SAS ファイルは、特殊な構造を持つ UNIX ファイルです。UNIX 動作環境は SAS のファイルを保存することによってそれを管理しますが、オペレーティングシステムは、SAS によってファイルに組み込まれた構造があるため、それを処理できません。たとえば、ユーザーは `ls` コマンドを使用してファイル名をリストすることはできますが、`vi` エディタを使用してファイルを編集することはできません。SAS ファイルは、永久ファイルまたは一時ファイルとできます。

データセット名の大文字小文字の区別

UNIX 動作環境では、SAS データセット名はすべて小文字で書き込まれます。こうした要件があるため、SAS データセット名はすべて小文字で書き込まれます。

UNIX ユーティリティ `mv` または `cp` を使用して SAS データセット名を大文字または大小文字混合で変更すると、SAS はデータセット名を読み込めなくなります。

UNIX では大小文字が区別されます。したがって、*xxx*.sas7bdat という名前のデータセットと *XXX*.sas7bdat という名前のデータセットは、同一ではありません。これらのデータセットを、まったく別のデータセットとして同一のディレクトリに共存させることもできます。

SAS ライブラリ

SAS ライブラリについて

SAS ファイルは SAS ライブラリに保存されます。SAS ライブラリは、UNIX ディレクトリ内にある SAS ファイルの集まりです。UNIX ディレクトリは SAS ライブラリとして使用できます。(このディレクトリには、外部ファイルと呼ばれる SAS では管理対象外のファイルを含めることもできます。外部ファイルへのアクセス方法については、“外部ファイルとデバイスの使用” (68 ページ) を参照してください。) SAS の Work ライブラリには一時 SAS ファイルが保存されます。これは、ユーザー用に自動的に定義されます。各永久 SAS ファイルについて、ライブラリを指定する必要があります。詳細については、“Work ライブラリ” (58 ページ)を参照してください。

ライブラリ参照名について

SAS ライブラリはライブラリ参照名で識別されます。ライブラリ参照名は、アプリケーション内のディレクトリを参照するための名前です。ライブラリ参照名の割り当て方法の詳細については、“UNIX 環境でのライブラリ参照名の使用による SAS ファイルの参照” (48 ページ)を参照してください。

エンジン

エンジンについて

SAS ファイルおよび SAS ライブラリは、エンジンを介してアクセスします。エンジンは、ライブラリ内のファイルにアクセスするために使用する必要があるルーチンのセットです。SAS では、そうしたファイルタイプに最適なエンジンを使用することにより、ファイルからの読み込みを(ときにはファイルへの書き込みも)可能にしています。一部のファイルタイプについては、使用するエンジンをユーザーが指定する必要があります。その他の場合は、最適なエンジンが自動的に選択されます。SAS データセットの作成に使用されるエンジンにより、そのファイルの形式が決まります。

追加リソース

SAS ファイル、ライブラリおよびエンジンの詳細については、*SAS 言語リファレンス: 解説編*を参照してください。

UNIX 環境に共通する SAS ファイルの種類

SAS データセット

SAS データセットについて

SAS データセットには 2 つのタイプがあります。

- “SAS データファイル(メンバタイプ DATA)” (36 ページ)
- “SAS ビュー(メンバタイプ VIEW)” (37 ページ)

ディスクリプタ情報とデータ値

データセットには、ディスクリプタ情報と、いずれかのエンジンで処理が可能な行および列から成る表として構成されるデータ値が含まれています。ディスクリプタ情報には、データセットタイプ、データセットラベル、データセット内の列の名前およびラベルなどが含まれています。SAS データセットには、1 つ以上の列のインデックスを含めることができます。

SAS データセットは 2 つの形で実装されます。

- データ値およびデータセットのディスクリプタ情報が 1 つのファイルに保存されている場合、その SAS データセットは SAS データファイルと呼ばれます。
- ファイルが、データセットのデータ値とディスクリプタ情報を取得する場所に関する情報を含んでいる場合、その SAS データセットは SAS ビューと呼ばれます。

デフォルトのエンジンは、データファイルまたはデータビューとインデックスを、単一のエンティティと同様に、データセットを処理します。

詳細については、“SAS データファイル(メンバタイプ DATA)” (36 ページ) および “SAS ビュー(メンバタイプ VIEW)” (37 ページ)を参照してください。

SAS データファイル(メンバタイプ DATA)

SAS データファイルはおそらく最も頻繁に使用される SAS ファイルです。このファイルには、`.sas7bdat` という拡張子があります。SAS データファイルは、DATA ステップ

内で、一部の SAS プロシジャによって作成されます。データファイルには 2 つのタイプがあります。

- ネイティブデータファイルは、データ値およびそのディスクリプタ情報を、SAS によりフォーマットされたファイルに保存します。このデータファイルは、SAS の以前のリリースで作成された従来の SAS データセットです。

  デフォルトのエンジンによって作成されるネイティブ SAS データファイルは、インデックス付けが可能です。インデックスとは、データファイルとは別に作成される、またそのデータファイルをインデックスする補助ファイルです。インデックスは、変数またはキーによって、データファイル内のオブザベーションにすばやくアクセスする機能です。UNIX 版では、インデックスは独立したファイルとして保存されますが、SAS データファイルの不可欠な要素として扱われます。

  注意:

  UNIX コマンドを使用してインデックスファイルを削除しないでください。 インデックスファイルを削除すると、SAS データセットが破損することがあります。また、ファイル名を変更したり別のディレクトリに移動したりもしないでください。インデックスの管理には、DATASETS プロシジャを使用します。

- インターフェイスデータファイルは、他のソフトウェアによってフォーマットされていて SAS では読み込みのみが可能なファイルに、データを保存します。詳細については、“UNIX 環境で BMDP、OSIRIS、SPSS ファイルにアクセスする” (61 ページ)を参照してください。

SAS ビュー(メンバタイプ VIEW)

SAS ビューには、データ値とディスクリプタ情報を取得するために必要な情報のみが含まれています。SAS ビューの作成方法によっては、実際のデータは、他の SAS データセットまたは他のベンダのファイルに配置されることがあります。

ビューには 2 つの種類があります。

- ネイティブ SAS ビューは、1 つ以上の SAS データファイルまたは SAS ビューのデータに関する情報を含んでいます。この種類のビューは、SQL プロシジャまたは DATA ステップで作成されます。
- インターフェイス SAS ビューは、他のソフトウェア製品(データベース管理システムなど)によってフォーマットされたデータに関する情報を含んでいます。たとえば、SAS/ACCESS ソフトウェアの ACCESS プロシジャは、インターフェイス SAS ビューを作成します。

SAS カタログ

カタログは、複数のエントリを含めることのできる特殊な SAS ファイルです。1 つの SAS カタログに、さまざまなタイプのエントリを保存できます。たとえば、カタログには、SAS/AF および SAS/FSP ソフトウェアが作成したエントリ、ウィンドウ環境アプリケーション、キー定義、SAS/GRAPH グラフなどを含めることができます。

カタログは CATALOG という SAS メンバタイプを持っています。

プログラムファイル

プログラムファイルは、コンパイル済み DATA ステップで、SAS メンバタイプは PROGRAM になります。詳細については、28 章: “Stored Compiled DATA Step Programs” (*SAS Language Reference: Concepts*)を参照してください。

アクセスディスクリプタファイル

アクセスディスクリプタファイルは、他のソフトウェア製品(Oracle や Sybase のデータベース管理システムなど)によってフォーマットされたデータを記述します。SAS/ACCESS ソフトウェアの ACCESS プロシジャによって作成されたディスクリプタファイルは、ACCESS という SAS メンバタイプを持っています。

UNIX 環境のファイル名拡張子とメンバタイプ

SAS では複数の異なるファイルタイプを区別する必要があるため、ファイルの作成時は自動的に各ファイルに拡張子が割り当てられます。また、各 SAS ファイルはライブラリのメンバであるため、各ファイルにメンバタイプが割り当てられます。

次の表に、ファイル拡張子とそれに対応する SAS メンバタイプを示します。

注意:

SAS ファイルのファイル拡張子は変更しないでください。 ファイル拡張子によって SAS がファイルにアクセスする方法が決まります。拡張子を変更すると、予期しない結果が生じる可能性があります。

表 2.1 *SAS ファイルタイプのファイル拡張子*

SAS 9			
ランダムアクセスファイル	**順次アクセスファイル**	**SAS メンバタイプ**	**説明**
.sas	.sas	.sas	SAS プログラム
.lst	.lst	.lst	プロシジャ出力
.log	.log	.log	SAS ログファイル
.sas7bdat	.sas7sdat	DATA	SAS データファイル
.sas7bndx	.sas7sndx	INDEX	データファイルインデックス(SAS System では独立したファイルとしては扱われない)
.sas7bcat	.sas7scat	CATALOG	SAS カタログ
.sas7bpgm	.sas7spgm	PROGRAM	プログラム(DATA ステップ)
.sas7bvew	.sas7svew	VIEW	SAS ビュー
.sas7bacs	.sas7sacs	ACCESS	アクセスディスクリプタファイル
.sas7baud	.sas7saud	AUDIT	監査ファイル
.sas7bfdb	.sas7sfdb	FDB	連結データベース

SAS 9			
ランダムアクセスファイル	順次アクセスファイル	SAS メンバタイプ	説明
.sas7bmdb	.sas7smdb	MDDB	多次元データベース
.sas7bods	.sas7sods	SASODS	ODS ファイル
.sas7bdmd	.sas7sdmd	DMDB	データマイニングデータベース
.sas7bitm	.sas7sitm	ITEMSTOR	アイテムストアファイル
.sas7butl	.sas7sutl	UTILITY	ユーティリティファイル
.sas7bput	.sas7sput	PUTILITY	永久ユーティリティファイル
.sas7bbak	.sas7sbak	BACKUP	バックアップファイル

UNIX ディレクトリにはさまざまなファイルを保存できますが、用途に応じて個別のディレクトリに保存するほうが実用的だと感じる場合もあります。また、異なる複数のエンジンがアクセスする複数のライブラリを、同一のディレクトリに保存することも可能です(ただしお勧めはしません)。詳細については、“UNIX 環境における 1 つのライブラリへの複数エンジンの使用” (54 ページ)を参照してください。

ファイル拡張子区切り文字の処理方法

SAS ファイルを物理名で直接参照する場合、VALIDMEMNAME の設定にかかわらず、最後の埋め込みピリオドが拡張子区切り文字になります。結果として、EXTEND オプションを VALIDMEMNAME=と一緒に使用する場合で、メンバ名自身にピリオドが含まれているときは、参照の一部として全拡張子を指定する必要があります。詳細については、“VALIDMEMNAME= System Option” (*SAS System Options: Reference*) を参照してください。

次の例でこの概念を示します。例にあるコメントは SAS ログからのものです。

```
options validmemname=extend;
libname mylib './saslib';
/* NOTE: Libref MYLIB was successfully assigned as follows: */
/* Engine: V9 */
/* Physical Name: SAS-library */

data mylib."my.member"n;
x=1;
run;
/* NOTE: The data set MYLIB.'MY.MEMBER'n has 1 observations */
/* and 1 variables. */

data _null_;
set './saslib/my.member.sas7bdat';
run;
/* NOTE: There were 1 observations read from the data set */
```

```
/* ./saslib/my.member.sas7bdat. */

data _null_;
set './saslib/my.member';
run;
/* ERROR: Extension for physical file name "./saslib/my.member" */
/* does not correspond to a valid member type. */
/* NOTE: The SAS System stopped processing this step because of */
/* errors. */
```

ダイレクト I/O の使用

ダイレクト I/O について

ダイレクト I/O は入力ファイルおよび出力ファイルを処理する方法であり、ファイルの操作に用いられます。ダイレクト I/O は、先に UNIX 動作環境の読み取りおよび書き込みキャッシュを介さなくても、SAS で直接ストレージデバイスに対してファイルの読み書きができるようにします。ダイレクト I/O は SAS ファイルに対して使用できます。ダイレクト I/O を使用すると、実行するジョブの数や種類によりますが、システムのパフォーマンスが改善します。

SAS では、ダイレクト I/O 関連する 3 つのオプションを使用します。

- ENABLEDIRECTIO ステートメントオプション
- USEDIRECTIO=ステートメントオプション
- USEDIRECTIO=データセットオプション

LIBNAME ステートメントの ENABLEDIRECTIO オプションは、ダイレクト I/O 処理を DATA ステートメント内にリストされているデータセットに使用できるようにします。データセットを指定するライブラリ参照名は、ENABLEDIRECTIO オプションを使用する LIBNAME ステートメント内で定義する必要があります。ENABLEDIRECTIO を使用するだけでダイレクト I/O が有効化されません。

ENABLEDIRECTIO オプションを使用してディレクトリに割り当てられたライブラリ参照名は、ENABLEDIRECTIO オプションを使用しないその同一ディレクトリに割り当てられた別のライブラリ参照名とは一致しません。2 つのライブラリ参照名は同一のディレクトリを指定しますが、ENABLEDIRECTIO が指定されたライブラリ参照名を使用して開かれたファイルの読み込みと書き込みには、ダイレクト I/O が使用できます。別のライブラリ参照名を使用して開かれたファイルの読み込みと書き込みには、通常のディスク I/O 呼び出しが使用されます。

DATA ステートメントの USEDIRECTIO=データセットオプションまたは LIBNAME ステートメントの USEDIRECTIO=ステートメントオプションは、ENABLEDIRECTIO ステートメントオプションが適用されているデータセットに対してダイレクト I/O を有効化にします。最初に ENABLEDIRECTIO オプションを適用せずに USEDIRECTIO=を使用しても、データセット内のダイレクト I/O は影響を受けません。

ダイレクト I/O の有効化

ダイレクト I/O は 2 つの方法で有効化できます。

- LIBNAME ステートメントで、ENABLEDIRECTIO と SEDIRECTIO=の両方のオプションを使用します。

この方法により、LIBNAME ステートメント内のライブラリ参照名によって参照されるすべてのファイルが、ダイレクト I/O 用に開きます。

- ダイレクト I/O をレンダリングして使用可能にするには、LIBNAME ステートメントの ENABLEDIRECTIO オプションを使用し、I/O 機能を有効化するには DATA ステートメントの USEDIRECTIO=データセットオプションを使用します。

  この方法により、オプションが使用されているデータセットのみがダイレクト I/O 用に開きます。データセットは、LIBNAME ステートメントのライブラリ参照名によって参照される必要があります。

これらのオプションとその使い方について詳しくは、次の各項目を参照してください。

- ENABLEDIRECTIO。参照先: “エンジン/ホストオプション” (348 ページ)
- USEDIRECTIO=。参照先: “エンジン/ホストオプション” (348 ページ)
- “USEDIRECTIO= データセットオプション: UNIX” (258 ページ)

メモリ内でのファイルの保持: SASFILE ステートメント

SASFILE ステートメントを使用して SAS データセットを開くことができます。SAS は、データセット全体をメモリに保持できるだけのバッファを割り当てようとします。メモリが十分に使用できる場合は、データセットが閉じるまでの間、データセット全体がメモリに保持されます。メモリが十分に使用できない場合は、可能な限り多くバッファが割り当てられます。ファイルが非常に大きい場合、または SAS がすでに大量のメモリを使用している場合は、SASFILE ステートメントの使用による効果は期待できません。

SASFILE ステートメントの初回実行時に、SAS はファイルを開きます。そのファイルはメモリに残っているため、後続の DATA ステップおよび PROC ステップは、そのファイルを使用するときに再び開く必要がありません。そのファイルは、2 つ目の SASFILE ステートメントがファイルを閉じるまで、あるいはプログラムまたはセッションが終了するまで、開いたままになります。詳細については、“SASFILE Statement” (*SAS Statements: Reference*)を参照してください。

UNIX 環境での SAS ファイルの共有

SAS ファイルの共有

複数の SAS プロセスが同時に SAS ファイル(データセットカタログ、ライブラリなど)への書き込みアクセスを持っている場合は、そのファイルが更新されたときに予期しない結果が生じます。SAS では、複数のユーザーがファイルへの書き込みアクセスを持てないようにファイルがロックされます。1 つの SAS プロセスが書き込みアクセスによってファイルを開くとき、その他のプロセスは、最初のプロセスがファイルを閉じるまで、書き込みアクセスを持てなくなります。SAS には、このファイル保護を無効にするためのステートメントとシステムオプションを備えています。ただし、ほとんどの場合、ユーザーはファイル保護を有効にしておく必要があります。

ファイルロックに使用するオプション: SAS ファイル

SAS ファイルのファイルロックは、次の方法で無効化できます。

- LIBNAME ステートメントで FILELOCKS オプションを使用します。

- FILELOCKS システムオプションを使用します。

SAS ファイルのファイルロック: FILELOCKS ステートメントオプション

デフォルトでは、SAS は、Write アクセスを 1 ユーザーに制限します。LIBNAME ステートメントの FILELOCKS オプションは、このデフォルト設定を無効にし、複数のユーザーにファイルの書き込みアクセスを許可します。LIBNAME ステートメントのライブラリ参照名の下で開いている SAS ファイルは、ロックされたファイルです。複数のユーザーがファイルの読み込みアクセスを持っています。

FILELOCKS オプションは、LIBNAME ステートメントのライブラリ参照名の下で開いているほとんどの(一部は除く)SAS の I/O ファイル(データセット、カタログなど)に適用されます。

詳細については、“LIBNAME ステートメント: UNIX” (344 ページ)を参照してください。

SAS ファイルのファイルロック: FILELOCKS システムオプション

デフォルトでは、SAS は、Write アクセスを 1 ユーザーに制限します。FILELOCKS システムオプションは SAS ファイルと外部ファイルの両方についてこのデフォルト設定を無効にし、複数のユーザーにファイルの書き込みアクセスを許可します。ユーザーは FILELOCKS システムオプションを使用することで、開いている個々のファイルに対してグローバルに動作を適用できます。

FILELOCKS システムオプションは、OPTIONS ステートメントでまたはコマンド行で、起動時に使用できます。ユーザーは、FILELOCKS システムオプションの複数のインスタンスを指定できます。各インスタンスは、パスおよび設定の内部テーブルに追加されます。FILELOCKS システムオプションは、LIBNAME ステートメントのライブラリ参照名で開かれるほとんど(一部除く)の SAS I/O ファイル(データセット、カタログなど)に適用されます。詳細については、“FILELOCKS システムオプション: UNIX” (377 ページ) および “LIBNAME ステートメント: UNIX” (344 ページ)を参照してください。

ロックされたファイルが使用できるまでの待機

別のプロセスによってロックされている SAS ファイルを使用する必要がある場合は、LIBNAME ステートメントの FILELOCKWAIT オプションを使って、ロックされたファイルが別のプロセスで使用可能になるのを SAS が待機する時間を指定できます。FILELOCKWAIT ステートメントオプションは、LIBNAME ステートメントのライブラリ参照名の下で開いているファイルにのみ影響します。詳細については、“LIBNAME ステートメント: UNIX” (344 ページ)を参照してください。

FILELOCKS=NONE の場合に確認する条件

ファイルロックが無効化された場合(つまり FILELOCKS システムオプションが NONE に設定されると)、SAS は、そのファイルにすでにロックがあるかどうかをチェックせずにファイルを開こうとします。このようなファイルは、共有更新アクセスから保護されません。

注意:

FILELOCKS=NONE オプションは使用しないことをお勧めします。 複数のユーザーが書き込みアクセスのために同一のファイルを開くと、そのファイルは破損する可能性があります。FILELOCKS=NONE オプションは、主に、ファイルがロックされていることによりジョブが失敗したかどうかを判断するために使用します。

FILELOCKS システムオプションを NONE に設定した場合は、次のいずれかのタスクを実行する必要があります。

- 各 SAS セッションで **sasuser** ディレクトリが他と重複しないようにします。通常はシステム管理者が、システム構成ファイル内でこのディレクトリを割り当てます。システム構成ファイルまたはユーザーの個人構成ファイルの指定は、SAS セッションを一度に 1 件だけ実行する限りそのディレクトリが重複しないようにするのに役立ちます。

  2 件以上の SAS セッションを同時に実行する場合は、各セッションに異なる複数の **sasuser** ディレクトリを指定することにより、確実にユーザーファイルが一意となるようにできます。最初のセッションでは、次のようにできます。

  ```
  -sasuser ~/sasuser
  ```

  n 番目のセッションでは、次のように使用できます。

  ```
  -sasuser ~/sasusern
  ```

  詳細については、“SAS 構成ファイルの処理の優先順位” (24 ページ) および “RSASUSER システムオプション: UNIX” (410 ページ)を参照してください。RSASUSER オプションは、複数のユーザーで共有する場合、Sasuser ライブラリの変更内容を制御するのに使用できます(“RSASUSER システムオプション: UNIX” (410 ページ)参照。)
- X ステートメントを使用するか、別の 2 つのウィンドウから SAS を呼び出すことにより、2 つ以上の SAS セッションを同時に実行し、同一の Sasuser.Profile カタログを使用する場合は、両方のセッションが同一のカタログを同時に使用してしまう可能性があるため、その SAS セッション内で Sasuser.Profile カタログを変更するアクション(WSAVE コマンドの使用、キーの割り当ての変更など)を実行しないでください。
- 同一のデータセットを複数のユーザーが同時に読み込むことができます。ただし、そのデータが上書きされたり破損したりしないように、データセットへの書き込みやその更新は、1 回につき 1 人のユーザーのみ可能とする必要があります。

FILELOCKS=CONTINUE の場合

デフォルトでは、SAS は、Write アクセスを 1 ユーザーに制限します。FILELOCKS=CONTINUE オプションを使用する場合、ファイルが別のユーザーにロックされているとファイルを開けなくなり、エラーメッセージがログに書き込まれます。ただし、SAS が他のなんらかのエラーを識別するメッセージを返した場合、SAS はファイルのロックを無視してファイルを開き、ジョブの実行を続行します。

ネットワークのファイルの共有

複数ワークステーションでのファイル共有について

SAS は、類似のコンピュータのネットワーク内にある 1 つ以上のワークステーション上で実行するライセンスを受けることができます。このライセンスでは、SAS が実行可能なワークステーションが細かくリストされています。ネットワーク内にあるライセンスのないワークステーションが SAS 実行可能ファイルへのアクセス権を持っている場合がありますが、それらのワークステーションでは SAS を実行できない場合があります。

ライセンス契約のあるワークステーションがファイルシステムを共有するために NFS マウントを介して接続される場合、それらすべてのワークステーションは、SAS 実行可能ファイルのコピーを 1 つ共有できます(ただしコピーの共有は必須ではありません)。それらのワークステーションで、SAS ファイルを共有することも可能です。ただし、SAS セッションが更新のためにデータセットまたはカタログを開く場合、そのセッションは

SAS ファイル上で排他的なファイルロックを取得し、他の SAS セッションがそのファイルにアクセスできないようにする必要があります。そのファイルが開いている間は、他の SAS セッションはアクセスできなくなります。

SAS が、NFS を介して接続されている複数タイプのワークステーションにインストールされている場合、それぞれのワークステーションは独自の SAS 実行可能ファイルのコピーを持っている必要があります。

異なるネットワークのファイルへのアクセス

2 台のコンピュータが同一のファイルシステムに接続されている場合は、異なるタイプのワークステーション上のファイルにアクセスできます。異なる動作環境で作成された外部ファイルにアクセスできます。

データセットまたはカタログを作成し、それをディレクトリに保存して、後で異なるネットワーク上の別のコンピュータからそのファイルにアクセスする場合、そのファイルの操作にはいくつかの方法があります。

- そのファイルをほとんど使用しない場合は、コンピュータにリモートでログオンして、そこで作業することができます。
- そのファイルが頻繁に変更される場合は、コンピュータにリモートでログオンしてそこで作業することができます。この方法では、ファイルの最新版にアクセスできます。PROC CPORT を使用してファイルをコンピュータにコピーする場合、元のファイルが、コピーされてから読み取られるまでの間に変更される可能性があります。
- そのファイルを一度だけ使用する場合は、(ファイルを自分のコンピュータに転送するのではなく)コンピュータにリモートでログオンすることができます。PROC CPORT を使用するのは効率的ではない場合があります。また、PROC CPORT をローカルで実行できるだけのディスクの空き容量がない場合もあります。
- FTP (ファイル転送プロトコル)または RCP (経路制御プロセッサ)を使用して、ファイルをリモートコンピュータから自分のコンピュータへ転送できます。
- 自分のコンピュータで作業の一部を行い、その他はリモートコンピュータで行うことができます。たとえば、ローカルコンピュータの小規模なテストケースでステートメントのセットを実行し、次いで実際の作業を送信してリモートコンピュータで完了する、という方法があります。同様に、別のコンピュータで大規模なデータセットをサブセット化してから、そのサブセットについてローカル分析を行うこともできます。SAS/CONNECT ソフトウェアを使用してこのタスクを完了できます。リモートライブラリサービスの詳細については、*SAS/CONNECT User's Guide* を参照してください。

トラブルシューティング: NFS マウントを使用したデータへのアクセス

FILELOCKS オプションが FAIL または CONTINUE に設定されている場合、NFS マウントを使用してデータにアクセスすると SAS がハングすることがあります。この問題を軽減するには、両方のコンピュータですべての NFS ファイルロックデーモンが実行していることを確認してください(通常は statd デーモンと lockd デーモン)。担当の UNIX システム管理者が、statd デーモンと lockd デーモンの起動を支援してくれます。

注: ファイルロックに問題があるかどうかをテストするために、FILELOCKS システムオプションを一時的に NONE に設定します。FILELOCKS を NONE に設定して問題が解決した場合、その原因は statd デーモンと lockd デーモンだと推測できます。データの破損と予期しない結果を招くおそれがあるため、FILELOCKS は絶対に NONE には設定しないこととをお勧めします。

UNIX 環境での 32 ビット版 SAS ファイルの 64 ビット版への移行

ファイルの移行について

ファイルの移行により、ライブラリが SAS の新しいリリースへ移行します。多くの場合、以前のリリースまたは他のホストの SAS ファイルは SAS 9.4 と互換性があります。ファイルに互換性がない場合は、Cross-Environment Data Access (CEDA)を使用して、ファイルおよびライブラリを移行できます。ユーティリティプロシジャ PROC MIGRATE を使用して、ファイルとライブラリを移行先に移動するプロセスを合理化できます。ファイルとライブラリを移行するときの考慮点として、自分のデータが現在置かれている SAS のリリース、ライブラリに存在するメンバタイプ、32 ビットライブラリから 64 ビットライブラリへメンバを移動する必要があるかどうか、といったことがあります。

CEDA 使用の詳細については、32 章: “Processing Data Using Cross-Environment Data Access (CEDA)” (*SAS Language Reference: Concepts*)を参照してください。MIGRATE プロシジャ使用の詳細については、35 章: “MIGRATE プロシジャ” (*Base SAS Procedures Guide*)を参照してください。MIGRATE プロシジャおよび Compatibility Calculator の使用に関する詳細については、次のサイトで Migration を参照してください。**SAS テクニカルサポート Web サイト**

SAS ファイルの移行の利点

SAS ファイルの移行により、次のことが可能になります。

- サポート対象外のデータファイルへの更新アクセスを得ること。
- インデックス、一貫性制約およびその他の機能へのアクセスを得ること。
- 出力形式および入力形式に長い名前を使用すること。
- 32,767 を超える変数を使用すること。
- 指定された変数の抑制トランスコードを使用すること。
- 64 ビット SAS セッションでの 32 ビットファイルに対する読み込みまたは書き込みのオーバーヘッドを回避すること。

Linux 環境での SAS ライブラリの移行方法

SAS ライブラリを移行するには、MIGRATE プロシジャを使用します。32 ビット Linux 環境から 64 ビット Linux 環境へ移行する場合で、カタログがライブラリ内に存在する場合は、32 ビットのリリース 9 SAS/CONNECT または SAS/SHARE Server にアクセスする必要があります。

MIGRATE プロシジャ使用の詳細については、35 章: “MIGRATE プロシジャ” (*Base SAS Procedures Guide*)を参照してください。MIGRATE プロシジャおよび Compatibility Calculator の使用に関する詳細については、次のサイトで Migration を参照してください。**SAS テクニカルサポート Web サイト**

以前のリリースで使用できる SAS ファイルの作成

V9 エンジンは、以前の SAS エンジンとは若干異なります。V9 エンジンでは、出力形式および入力形式の名前を、以前の SAS エンジンよりも長くすることができます。異なるリリース間の互換性の確認方法については、*SAS 言語リファレンス: 解説編*を参照してください。また、リリース間の互換性については、**テクニカルサポート Web サイト**で Migration を参照してください。

以前のリリースまたは他のホストからの SAS ファイルの読み込み

バージョン 6 のファイルの読み込み

SAS は、V6 読み込み専用エンジンを使用して、互換性のあるコンピュータタイプで作成されたリリース 6 のデータセットを読み込むことができます。ほとんどの場合、V6 エンジンが自動的に呼び出されるため、ユーザーがそれを指定する必要はありません。次の例で、V6 エンジンの使用方法を示します。

- Linux で SAS 9.4 を実行している場合、V6 エンジンを使用して、SAS の任意の Intel ABI リリース(SCO UNIX など)で作成されたリリース 6 のデータセットを読み込むことができます。
- HP-UX で SAS 9.4 を実行している場合、V6 エンジンを使用して、HP-UX、Solaris、AIX または IRIX 上で作成されたリリース 6 のデータセットを読み込むことができます。
- リリース 6 ファイルとの互換性に関する詳細については、“Introduction to Version Compatibility” (*SAS Language Reference: Concepts* 33 章)を参照してください。

互換性のあるコンピュータタイプからのバージョン 8 以降のファイルの読み込み

64 ビットの SAS で作成されたファイルは、SAS 9.4 と互換性があります。ファイルを読み込むために CEDA を使用する必要はありません。

互換性のないコンピュータタイプからのバージョン 8 以降のファイルの読み込み

既存の SAS ファイルと SAS 9.4 との互換性

SAS 9 の場合、AIX、HP-UX、および Solaris 環境に対応する SAS は 64 ビットのみです。SAS の 32 ビットリリースで作成された一部の SAS ファイルは、V9 エンジンで読み込むことはできません。

データセットを読み込むために、自動的に CEDA の使用が試みられます。CEDA を使用してデータセットを読み込む場合、コードに `msglevel=i` を含めると、SAS がログに NOTE メッセージを書き込みます。

次の表に、CEDA を使用する場合に各 SAS ファイルでサポートされる処理を示します。

表 2.2 SAS 9 でのリリース 8 の 32 ビットファイルでサポートされる処理

ファイルタイプ	サポート
SAS ファイル	入力処理および出力処理。(SAS 9 では、新規のデータファイルを 32 ビットファイルから作成する場合、新しいファイルは通常 64 ビットになります。CEDA の詳細については、32 章: “Processing Data Using Cross-Environment Data Access (CEDA)” (*SAS Language Reference: Concepts*)を参照してください。移行情報については、次のサイトで Migration を参照してください。**テクニカルサポートの Web サイト**)
MDDB ファイル	入力処理。
PROC SQL ビュー	入力処理。
Oracle または Sybase 用の SAS/ACCESS ビュー	入力処理。
Oracle または Sybase 用以外の SAS/ACCESS ビュー	サポートなし。
SAS カタログ	サポートなし。
保存されコンパイルされた DATA ステッププログラム	サポートなし。
DATA ステップビュー	サポートなし。
アイテムストア	サポートなし。

注: SAS 9 では、新規のデータファイルを 32 ビットファイルから作成する場合、新しいファイルは通常 64 ビットになります。CEDA の詳細については、32 章: “Processing Data Using Cross-Environment Data Access (CEDA)” (*SAS Language Reference: Concepts*)を参照してください。移行情報については、次のサイトで Migration を参照してください。**テクニカルサポートの Web サイト**。

CEDA を使用したバージョン 8 以降のファイルへのアクセス

CEDA を使用すると、任意のディレクトリベースの動作環境(UNIX、Windows など)にある、バージョン 8 以降で作成された SAS データセットを、別のディレクトリベースの環境で実行している SAS セッションで読み込めるようになります。SAS 9.4 では、以前のリリースで作成されたデータセットにユーザーがアクセスしようとすると、SAS は自動的に CEDA を使用してファイルを処理します。たとえば、Linux で SAS 9.4 を実行している場合、64 ビット Solaris ホスト上のリリース 8 で作成されたデータセットを処理するために、SAS は CEDA を使用します。CEDA を使用する場合、ユーザーはファイルの読み込みアクセスおよび書き込みアクセスを持っています。しかしながら、ファイルは更新できません。互換性については、**テクニカルサポート Web サイト**で Migration を参照してください。

最適なシステムパフォーマンスを得るために、ネイティブ形式のデータセットを使用することをお勧めします。そうしないと、CEDA に必要な CPU リソースが増加し、システムパフォーマンスが低下する可能性があります。

32 ビット SAS データセット、Oracle または Sybase の SAS/ACCESS ビュー、SQL ビューまたは 64 ビット SAS セッションの MDDB ファイルにアクセスする必要がある場合は、CEDA を使用してこれらのファイルにアクセスできます。CEDA はファイルの読み込みアクセスおよび書き込みアクセスを提供します。ただし、CEDA 更新処理をサポートしていません。ユーザーがこれらのファイルに対して読み込みや書き込みを行うたびに、CEDA が消費するリソースは増加します。CEDA の詳細については、32 章: "Processing Data Using Cross-Environment Data Access (CEDA)" (*SAS Language Reference: Concepts*)を参照してください。

カタログおよび他の SAS ファイル(SAS データセットは含まない)には、作成元のアプリケーションのみが認識できるデータ構造が含まれています。これらのカタログとファイルは、文字または数値オブジェクト以外のデータオブジェクトが含まれている場合があるため、64 ビット SAS と以前の 32 ビットの SAS リリースとの間で共有することはできません。

UNIX 環境でのライブラリ参照名の使用による SAS ファイルの参照

SAS ファイルの参照法

永久 SAS ファイルに対して読み込みまたは書き込みを行う場合は、次の 2 つの方法のうちいずれかによって SAS を参照することができます。

- 適切なステートメント(DATA、SET、MERGE、UPDATE、OUTPUT、PROC など)のパス名を使用して、データファイルを直接参照します。
- ライブラリ参照名をデータファイルが含まれている SAS ライブラリ(ディレクトリ)に割り当てて、そのライブラリ参照名を、2 レベルのファイル名の第 1 レベルとして使用します。

ライブラリ参照名について

ライブラリ参照名は、SAS のセッションまたはジョブの続行中にライブラリを参照するのに使用できるエイリアスです。次のいずれかが該当する場合は、ライブラリ参照名を使用してください。

- データファイルのパス名が長く、プログラム内で何回か指定する必要がある。
- パス名が変更される可能性がある。パス名が変更される場合、ファイルへの参照すべてではなく、ライブラリ参照名を割り当てているステートメントのみを変更する必要があります。
- ご使用のアプリケーションを別のプラットフォームで使用する。ライブラリ参照名を使用すると、アプリケーションを別のオペレーティング環境に移すことが容易になります。
- ライブラリを連結する必要があります。詳細については、"複数のディレクトリへの 1 つのライブラリ参照名の割り当て(ディレクトリの連結)" (52 ページ)を参照してください。

ライブラリ参照名は SAS レジストリに保存できます。詳細については、"SAS レジストリファイルのカスタマイズ" (18 ページ)を参照してください。

ライブラリ参照名の割り当て

ライブラリ参照名の割り当て方式

次のいずれかを使用して SAS ライブラリ参照名を割り当てることができます。

- LIBNAME ステートメント
- LIBNAME 関数
- DMLIBASSIGN コマンド
- **LIBNAME** ウィンドウ
- **SAS Explorer** ウィンドウ

ライブラリ参照名の割り当ては、ユーザーがライブラリ参照名をクリアしたり、別の LIBNAME ステートメントまたは LIBNAME 関数で同一のライブラリ参照名を使用したりしない限り、SAS のジョブ、セッションまたはプロセスが続行する間は有効です。

SAS プロセスからライブラリ参照名を割り当てる場合、そのライブラリ参照名はその SAS プロセス内でのみ有効です。SAS プロセス内からライブラリ参照名をクリアする場合、そのライブラリ参照名は他の SAS プロセスからはクリアできません。

LIBNAME ステートメントの使用

LIBNAME ステートメントは、SAS に対する SAS ライブラリを識別し、そのライブラリにエンジンを関連付け、そのライブラリのオプションをユーザーが指定できるようにし、そのライブラリにライブラリ参照名を割り当てます。LIBNAME ステートメント構文については、“LIBNAME ステートメント: UNIX” (344 ページ)を参照してください。

LIBNAME 関数の使用

LIBNAME 関数では、LIBNAME ステートメントと同一の引数とオプションを使用します。LIBNAME 関数の詳細については、“LIBNAME Function” (*SAS Functions and CALL Routines: Reference*)を参照してください。

DMLIBASSIGN コマンドの使用

DMLIBASSIGN コマンドを使用してライブラリ参照名を割り当てます。次の操作を行ってください。

1. コマンドウィンドウで DMLIBASSIGN コマンドを発行します。

   New Library ダイアログボックスが表示されます。

2. **Name** フィールドでライブラリ参照名を指定します。
3. **Engine** フィールドでメニューからデフォルトのエンジンまたは別のエンジンを選択し、ライブラリ参照名用のエンジンを指定します。どのエンジンを選択するかによって、**Library Information** 領域に表示されるフィールドが変化します。
4. SAS を呼び出すとき、**Enable at startup** をクリックしてこのライブラリ参照名を割り当てます。
5. SAS ライブラリに必要な情報を、**Library Information** 領域で指定します。選択したエンジンによっては、入力に使用する **Path** フィールドが表示されない場合があります。
6. **Options** フィールドで LIBNAME オプションを指定します。これらのオプションは、別のソフトウェアベンダのリレーショナルデータベースシステムにアクセスする SAS エンジン固有のオプションを含め、使用するホストまたはエンジンに固有のものにすることができます。

7. **OK** をクリックします。

LIBNAME ウィンドウの使用

次の手順を実行して、**LIBNAME** ウィンドウからライブラリ参照名を割り当てます。

1. コマンドウィンドウに LIBNAME コマンドを発行します。

   LIBNAME ウィンドウが表示されます。

2. **File** メニューで **New** を選択します。

   New Library ダイアログボックスが表示されます。

3. **New Library** ダイアログボックスのフィールドを入力します(“DMLIBASSIGN コマンドの使用” (49 ページ)を参照)。

4. **OK** をクリックします。

SAS Explorer ウィンドウの使用

次の手順を実行して、**SAS Explorer** ウィンドウからライブラリ参照名を割り当てます。

1. ツリー構造内の **Libraries** ノードがアクティブになった後、**File** メニューで **New** を選択します。

   New ダイアログボックスが表示されます。

2. **Library** を選択して **OK** をクリックします。

   New Library ダイアログボックスが表示されます。

3. **New Library** ダイアログボックスのフィールドを入力します(“DMLIBASSIGN コマンドの使用” (49 ページ)を参照)。

4. **OK** をクリックします。

ライブラリ参照名の永久的な割り当て

ライブラリ参照名を、それが SAS セッション間で有効になるように保存することもできます。次のいずれかの方法で、ライブラリ参照名を常に割り当てることができます。

- autoexec ファイル内で LIBNAME ステートメントまたは LIBNAME 関数を指定します。詳細については、“LIBNAME Function” (*SAS Functions and CALL Routines: Reference*) または “LIBNAME ステートメント: UNIX” (344 ページ)を参照してください。
- DMLIBASSIGN コマンド、**LIBNAME** ウィンドウまたは **SAS Explorer** ウィンドウを使用してライブラリ参照名を割り当てるときは、**Enable at startup** を選択します。このオプションを選択すると、SAS レジストリにライブラリ参照名が保存されます。これらの方法の詳細については、“ライブラリ参照名の割り当て” (49 ページ)を参照してください。
- 環境変数をライブラリ参照名として使用します。環境変数を起動ファイルに含めて、SAS が呼び出されたときにその環境変数が設定されるようにします。

ライブラリ参照名を使用した永久 SAS ライブラリへのアクセス

ライブラリ参照名を定義した後、ライブラリ参照名を次の 2 つの方法のいずれかによって使用し、永久 SAS ライブラリにアクセスします。

- 2 レベルの SAS ファイル名の第 1 レベルとして使用。

libref.member-name
ここで *libref* はファイルが保存されるディレクトリを示す第 1 レベルの名前、*member-name* は読み込むか作成するファイルの名前です。

- USER=オプションの値として使用。(詳細については、“1 レベル名を使用した永久ファイル(ユーザーライブラリ)へのアクセス” (59 ページ)を参照してください。)

たとえば、これらの SAS ステートメントは、**/users/myid/mydir** というディレクトリに保存されている Sales ライブラリの、データファイル Final.sas7bdat にアクセスします。

```
libname sales '/users/myid/mydir';
data sales.final;
```

UNIX 環境でのパス名の指定

ディレクトリ名およびパス名の指定ルール

UNIX ディレクトリとファイルのパス名の指定においては、さまざまな SAS ステートメント内でデータファイル名を直接指定するか、LIBNAME ステートメント内でライブラリ名を指定してからライブラリ参照名を参照するかにかかわらず、同一のルールが適用されます。

ディレクトリおよびファイルのパス名を、引用符で囲んで指定します。指定のレベルは、ユーザーのカレントディレクトリによって異なります。

例 1: カレントディレクトリにないファイルへのアクセス

/u/2011/budgets がカレントディレクトリではない場合、May という名前のデータファイルにアクセスするには、パス名全体を指定する必要があります。

```
data '/u/2011/budgets/may';
```

ライブラリ参照名を使用する場合は、次のように指定します。

```
libname budgets '/u/2011/budgets';
data budgets.may;
```

例 2: カレントディレクトリにあるファイルへのアクセス

/u/2011/budgets がカレントディレクトリである場合は、ファイル名のみの指定が可能です。

```
data 'quarter1';
merge 'jan' 'feb' 'mar';
run;
```

注: 引用符を省略すると、それらのデータセットは Work ディレクトリに保存されるものと判断されます。

ライブラリ参照名を使用する場合は、次のように指定します。

```
libname budgets '.';
data budgets.quarter1;
merge budgets.jan budgets.feb budgets.mar;
run;
```

パス名の有効な文字の置換

次の表に示す文字の置換を使用してパス名を指定できます。

表 2.3 パス名での文字の置換

文字	意味
~/	$HOME/ パス名の先頭でのみ使用できます。
~name/	name のホームディレクトリです(**/etc/passwd** ファイルから取得)。パス名の先頭でのみ使用できます。
!sasroot	**sasroot** ディレクトリの名前です(“!SASROOT ディレクトリ” (445 ページ)を参照)。パス名の先頭でのみ指定できます。
.	現在の作業ディレクトリです。
..	現在の作業ディレクトリの親です。
$VARIABLE	環境変数 VARIABLE です。

複数のディレクトリへの 1 つのライブラリ参照名の割り当て(ディレクトリの連結)

ディレクトリの連結について

LIBNAME ステートメントを使用して、ライブラリ参照名とエンジンを 1 つ以上のディレクトリ(Work ディレクトリを含む)に割り当てることができます。

SAS データセットが複数のディレクトリに配置されている場合は、次の例のように、単独のライブラリ参照名を指定して、複数のディレクトリの場所を連結することにより、それらのディレクトリを単独の SAS ライブラリとして扱うことができます。

```
libname income ('/u/2011/revenue', '/u/2011/costs');
```

このステートメントは、2 つのディレクトリ **/u/2011/revenue** および **/u/2011/costs** が、単独の SAS ライブラリとして扱われることを示しています。

複数のライブラリ参照名がすでに SAS ライブラリに割り当てられている場合は、そのライブラリ参照名を次の例のように使用して、ライブラリの連結を示すことができます。

```
libname income ('/u/2011/corpsale', '/u/2011/retail');
libname costs ('/u/2011/salaries', '/u/2011/expenses');
libname profits (income, costs, '/u/2011/capgain');
```

このステートメントは、**/u/2011/corpsale**、**/u/2011/retail**、**/u/2011/salaries**、**/u/2011/expenses**、**/u/2011/capgain** という 5 つのディレクトリが、単独の SAS ライブラリとして扱われることを示しています。

連結されたライブラリへの SAS のアクセス法

SAS ライブラリを連結するとき、SAS はライブラリへのアクセス用のプロトコルを使用します。このプロトコルは、読み込み、書き込みまたは更新のいずれの目的でライブラリにアクセスしているかによって異なります。(プロトコルとはルールのセットのことです。)

次のセクションでは、SAS がプロトコルを使用して、アクセス先となるディレクトリを判断しています。(これらの例で示されるプロトコルは、SAS ファイル(DATA ステップの DATA、UPDATE、MODIFY の各ステートメント、SQL プロシジャ、APPEND プロシジャなど)にアクセスする、すべての SAS ステートメントとプロシジャに適用されます。)

入力時と更新時のファイルへのアクセス

入力または更新の目的で SAS データセットがアクセスされるときは、名前で検索される最初の SAS データセットがそのアクセス対象となります。たとえば、次のステートメントをサブミットする場合、データセット **old.species** が両方のディレクトリに存在するときは、**mysasdir** ディレクトリにあるほうが印刷されます。

```
libname old ('mysasdir','saslib');
proc print data=old.species;
run;
```

FSEDIT プロシジャを使用して更新の目的で **old.species** を開いた場合も、同じ結果になります。

出力時にファイルにアクセスする

データセットが出力の目的でアクセスされた場合、データセットは常に最初のディレクトリに書き込まれます(当該ディレクトリが存在する場合)。当該ディレクトリが存在しない場合はエラーメッセージが表示されます。たとえば、ユーザーが次のステートメントを送信した場合、**old.species** データセットは最初のディレクトリ(**mysasdir**)に書き込まれ、既存のデータセットはすべて同じ名前に置き換わります。

```
libname old ('mysasdir','saslib');
data old.species;
x=1;
y=2;
run;
```

old.species データセットのコピーが 2 番目のディレクトリに存在する場合は、置き換わりません。

名前が同じデータセットへのアクセス

DATA ステートメントおよび SET ステートメントを使用して同一名のデータセットにアクセスする場合、DATA ステートメントは出力ルールを使用し、SET ステートメントは入力ルールを使用します。次のステートメントを実行するとき、**test.species** はもともとは 2 番目のディレクトリ **mysasdir** にのみ存在すると仮定します。次のステートメントを実行します。

```
libname test ('sas','mysasdir');
data test.species;
set test.species;
if value1='y' then
value2=3;
run;
```

DATA ステートメントは、出力ルールに従って、**test.species** を出力用に開きます。つまり、連結されたライブラリの最初にあるデータセットが開きます(**sas**)。SET ステートメントは、入力ルールに従って、2 番目のディレクトリ(**mysasdir**)にある既存の **test.species** データセットを開きます。したがって、元の **test.species** データセットは更新されません。DATA ステップの実行後は、2 つの **test.species** データセットがそれぞれのディレクトリに一つずつ存在します。

UNIX 環境における 1 つのライブラリへの複数エンジンの使用

ライブラリ参照名を単独のディレクトリに割り当て、各ライブラリ参照名に別のエンジンを割り当てることができます。たとえば、次のステートメントを実行すると、**one** で参照されるデータセットは、デフォルトのエンジンを用いて作成され、アクセスされます 一方、**two** で参照されるデータセットは、順次エンジンを用いて作成され、アクセスされます。

```
libname one v9 '/users/myid/educ';
libname two v8 '/users/myid/educ';
```

注: ユーザーは各ライブラリにアクセスするための最適なエンジンを覚えておく必要があるため、タイプの異なる複数のライブラリを 1 つのディレクトリに保持することはお勧めしません。タイプの異なる複数のライブラリが含まれているディレクトリ内では、ライブラリにアクセスするためのエンジンが正しく判断されません。詳細については、“LIBNAME ステートメントからのエンジン名の省略” (347 ページ)を参照してください。

UNIX 環境におけるライブラリ参照名としての環境変数の使用

環境変数はライブラリ参照名として使用できます。変数名はすべて大文字にする必要があります。また、変数の値はディレクトリの完全なパス名とする必要があります。つまり、ディレクトリの名前はスラッシュで始まっている必要があります。

注: UNIX 版 SAS では、USER 環境変数を使用した User ライブラリ参照名の割り当てはサポートされていません。

/users/mydir/educ でライブラリを使用し、EDUC 環境変数でそれを参照するとします。その変数を次のタイミングで定義します。

- SAS を起動する前。“UNIX 環境で環境変数を定義する” (439 ページ)を参照してください。たとえば、Korn シェルでは次のように使用します。

  ```
  export EDUC=/users/mydir/educ
  ```

- SAS を起動した後。X ステートメント(“SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)を参照)および SAS **setenv** コマンドを使用できます。

  ```
  x setenv EDUC /users/mydir/educ;
  ```

ライブラリ参照名を環境変数として定義するときはエンジンを指定できないため、使用するエンジンは自動的に決定されます(“LIBNAME ステートメントからのエンジン名の省略” (347 ページ)を参照)。

ライブラリ参照名が定義された後、ユーザーはそれを使ってライブラリに保存されているデータセットにアクセスできます。

```
proc print data=educ.class;
run;
```

注: 変数とライブラリ参照名が同じ名前ではあるが別のライブラリを参照している場合は、ライブラリ参照名が使用されます。

UNIX 環境で SAS によって割り当てられるライブラリ参照名

3 つのライブラリ参照名が自動的に定義されます。

Sashelp
: SAS セッションのさまざまな要素を制御するための情報が含まれたカタログのグループが含まれています。Sashelp ライブラリは**!SASROOT** ディレクトリにあります。詳細については、"!SASROOT ディレクトリ" (445 ページ)を参照してください。

Sasuser
: SAS の機能(ウィンドウサイズ、フォント設定、プリンタエントリなど)を必要に応じてカスタマイズできる、SAS カタログが含まれています。Sashelp ライブラリのデフォルト設定は、ユーザーのアプリケーションには適していません。デフォルトを修正したカスタム設定を、Sasuser ライブラリに保存することができます。

Work
: 各 SAS セッションまたはジョブの開始時に SAS によって自動的に定義される、一時的またはスクラッチライブラリです。Work ライブラリには、2 種類の一時ファイルが保存されます。ユーザーが自分で作成するファイルと、通常の処理の一環として SAS 内部で作成されるファイルです。

これらのライブラリ参照名とライブラリのライブラリ参照名は、予約されたライブラリ参照名です。SAS/GRAPH ソフトウェアをお持ちの場合は、MAPS ライブラリ参照名が自動的に定義されます。これらのライブラリすべての詳細については、"Permanent and Temporary Libraries" (*SAS Language Reference: Concepts* 24 章)を参照してください。Sasuser および Work には、オペレーティングシステム依存関係があります。

Sasuser ライブラリ

Sasuser ライブラリについて

Sasuser ライブラリには、ユーザーが SAS セッション用に指定したカスタマイズ項目(ウィンドウサイズ、位置決め、色、フォント、プリンタエントリなど)が含まれています。SAS の起動時に、**Sasuser** ディレクトリでこれらのカスタマイズ項目が検索されます。このディレクトリは存在しない場合は、SASUSER システムオプションによって自動的にディレクトリが作成されます。デフォルトのディレクトリはシステム構成ファイル(sasv9.cfg)で設定され、通常は次のようになります。

```
-sasuser ~/sasuser.v94
```

この指定により、ユーザーのホームディレクトリ内に Sasuser ライブラリ参照名用のディレクトリが作成されます。ご使用のシステムでのこのディレクトリの値を決定するには、PROC OPTIONS または **libname sasuser LIST** を使用します。

Sasuser ライブラリには、RSASUSER システムオプションを使用することにより、読み込み専用アクセスのみを許可できます。SASUSER システムオプションおよび RSASUSER システムオプションの詳細については、“SASUSER システムオプション: UNIX” (417 ページ) および “RSASUSER システムオプション: UNIX” (410 ページ)を参照してください。

Sasuser ライブラリが作成された後は、SAS セッションを開始するたびに、同じ Sasuser ライブラリ参照名がそのライブラリに割り当てられます。SAS セッションの続行中は、それをクリアしたり再割り当てしたりはできません。ライブラリを削除した場合は、次回セッションが開始するときにそれが再作成されます。ユーザーのためにライブラリ参照名が割り当てられるので、このライブラリを参照する前に LIBNAME ステートメントを使用する必要はありません。

Sasuser ライブラリのコンテンツ

ユーザーがカスタマイズした内容は、Sasuser ライブラリ内の、次のいずれかの場所に保存されます。

- “Sasuser.Profile カタログ” (56 ページ)
- “Sasuser.Registry カタログ” (57 ページ)
- “Sasuser.Prefs ファイル ” (58 ページ)

Sasuser.Profile カタログ

Sasuser.Profile カタログの概要

Sasuser.Profile カタログは、ユーザーの Sasuser ライブラリにある profile.sas7bcat ファイルです。このカタログを使用して、ユーザーによる SAS の操作方法をカスタマイズできます。SAS はこのカタログには、ファンクションキーの定義、グラフィックアプリケーション用フォント、ウィンドウの属性および対話型のウィンドウプロシジャからのその他の情報を保存します。ファンクションキーの定義、ウィンドウ属性(サイズ、色、位置など)、PMENU 設定などのユーザーが変更した内容は、Sasuser.Profile カタログに保存されます。Sasuser.Profile カタログの情報は、ユーザーが処理で必要とするときに自動的にアクセスされます。

SAS から Sasuser.Profile カタログへのアクセス法

Sasuser.Profile カタログは、最初にそれが検索されて存在しなかったときに、自動的に作成されます。 対話型のウィンドウ環境を使用している場合は、初回の SAS セッションでシステムを初期化しているときに、Sasuser.Profile カタログが作成されます。他のいずれかの実行モードを使用している場合は、Sasuser.Profile カタログを必要とする SAS プロシジャを最初に実行するときに、そのカタログが作成されます。

Sasuser.Profile カタログが存在しない場合

Sasuser.Profile カタログが存在しない場合は、起動時に自動的に Sashelp.Profile カタログが確認されます。(このカタログは、ユーザーが自分の Sasuser.Profile カタログを Sashelp ライブラリにコピーしておいた場合に限り、存在します。) Sashelp.Profile カタログが存在する場合は、それが Sasuser ライブラリにコピーされ、そのカタログがユーザーの新しい Sasuser.Profile カタログとなります。Sashelp.Profile カタログ存在しない場合は、SAS セッションのデフォルトの設定を用いて自動的に Sasuser.Profile が作成されます。ユーザーの SAS セッションのデフォルト設定は、Sashelp ライブラリ内の複数のカタログに保存されます。キー設定またはその他のオプションを変更した場合は、Sasuser.Profile カタログに新しい情報が保存されます。元のデフォルト設定を Sasuser.Profile カタログに復元するには、CATALOG プロシジャまたは **CATALOG**

ウィンドウを使用して、ユーザーの Sasuser.Profile カタログからエントリを削除します。デフォルトでは、次いで Sashelp ライブラリの対応するエントリが使用されます。

Sasuser.Profile カタログの破損のチェック

SAS を起動すると、破損していない既存の Sasuser.Profile カタログが自動的に確認されます。カタログが検出された場合、Sasuser.Profile カタログが Sasuser.Profbak にコピーされます。このバックアップカタログは、Sasuser.Profile が破損した場合に使用されます。

SAS を起動して、カスタマイズした内容が失われていた場合は、ユーザーの Sasuser.Profile カタログが破損しているか、同じユーザー ID で開始された別の SAS セッションによってそれがロックされています。いずれかの条件に該当する場合は、ロックされているか破損した Sasuser.Profile カタログを、Sashelp.Profile または Sasuser.Profbak を使用して SAS が置換します。

Sasuser.Profile カタログがロックまたは破損している場合

ユーザーの Sasuser.Profile カタログがロックされている場合は、Sashelp.Profile が自動的に確認されます。Sashelp.Profile が存在する場合は、Work.Profile がコピーされ、次いでカスタマイズした内容が Sasuser.Profile カタログのかわりに Work.Profile カタログに保存されます。この Work.Profile カタログは、SAS セッションが続行する間使用されます。Work ディレクトリの内容は一時的なものであるため、ユーザーが Work.Profile カタログに保存するカスタマイズ内容は、SAS セッションの終了時に失われます。

ユーザーの Sasuser.Profile カタログが破損している場合、SAS は破損したカタログを Sasuser.Badpro.SAS にコピーし、Sasuser.Profbak を確認します。Sasuser.Profbak が存在する場合は、SAS はそれを Sasuser.Profile にコピーします。以前のセッションでユーザーが Sasuser.Profile カタログに加えた変更内容は、すべて失われます。ユーザーの Sasuser.Profile カタログが複数の SAS セッションで使用されている場合、ユーザーは RSASUSER システムオプションを指定して、Sasuser ライブラリに読み込み専用アクセス権を与えることができます。この権限は読み込み専用のため、カスタマイズした内容を、その SAS セッションの続行中にユーザーの Sasuser.Profile カタログに保存することはできません。

Sasuser.Profile カタログおよびそれに関連するカタログの詳細と、ロックされているか破損しているプロファイルカタログの復元については、30 章: “SAS Catalogs” (*SAS Language Reference: Concepts*)を参照してください。

Sasuser.Registry カタログ

Sasuser.Registry カタログの概要

Sasuser.Registry カタログは、ユーザーの Sasuser ライブラリにある regstry.sas7bitm ファイルです。SAS セッションの続行中にユニバーサル印刷エントリまたはライブラリ参照名の割り当てを変更する場合は、Sasuser.Registry カタログの変更内容が保存されます。

SAS から Sasuser.Registry カタログへのアクセス法

SAS の起動時に、**Sasuser** ディレクトリで Sasuser.Registry カタログに書き込みが可能かどうかが確認されます。このカタログに書き込みができない場合は、SAS ログに次の警告が表示されます。

```
WARNING: Unable to open SASUSER.REGISTRY. WORK.REGISTRY will be used instead.
NOTE: All registry changes will be lost at the end of the session.
```

Sasuser.Registry カタログの読み込みが可能な場合は、Sasuser.Registry カタログがコピーされ、Work.Registry カタログが(Work ライブラリに)作成されます。この Work.Registry カタログは、その SAS セッションが続行する間使用されます。Work ライブラリの内容は一時的なものであるため、ユーザーが Work.Registry カタログに保存するカスタマイズ内容は、SAS セッションの終了時に失われます。ただし、Sasuser.Registry カタログに保存されたカスタマイズ内容は存続します。

Sasuser.Registry カタログの読み込みが不可能な場合は、SAS セッションのデフォルトの設定を用いて自動的に Work.Registry カタログが作成されます。この場合、SAS ログに対し追加の警告が発行されます。

```
WARNING: Unable to copy SASUSER.REGISTRY to WORK.REGISTRY.
```

Sasuser.Prefs ファイル

ユーザーが **Preferences** ダイアログボックスで指定する設定(ただし **General** タブのリソースは例外)は、Sasuser.Prefs ファイルに保存されます。これらのリソースの詳細については、“Preferences ダイアログボックスを使用し、X リソースを変更する” (172 ページ)を参照してください。

Work ライブラリ

Work ライブラリは、各 SAS セッションまたはジョブの開始時に自動的に定義される、一時ライブラリです。Work ライブラリには、ユーザーが作成する一時 SAS ファイルと、SAS の内部で作成されるファイルが保存されます。

Work ライブラリのファイルにアクセスするには、そのファイルに 1 レベル名を指定します。ライブラリ参照名 Work は、User ライブラリ参照名が割り当てられていない限り、これらのファイルに自動的に割り当てられます。

SAS 起動時に、Work ライブラリ参照名は、WORK システムオプション(“WORK システムオプション: UNIX” (434 ページ)参照)で指定されたディレクトリのサブディレクトリに割り当てられます。このサブディレクトリは通常、**SAS_workcode_nodename** という名前で、次の特性があります。

workcode
: 12 文字コードです。最初の 4 文字はランダムに生成された番号です。次の 8 文字は、SAS セッションの 16 進数のプロセス ID 番号です。

nodename
: SAS プロセスが実行している UNIX コンピュータの名前です。

SAS セッションの続行中は、このライブラリ参照名をクリアしたり再割り当てしたりはできません。

WORKINIT システムオプションおよび WORKTERM システムオプションが Work ライブラリの作成と削除を制御します。詳細については、“WORKINIT System Option” (*SAS System Options: Reference*)および“WORKTERM System Option” (*SAS System Options: Reference*)を参照してください。

注: SAS セッションが不適切に(たとえば、**kill -9** コマンドを使用して)終了された場合、SAS は **SAS_workcode_nodename** ディレクトリを削除しません。その場合、“cleanwork コマンド” (452 ページ)を使用してディレクトリを削除してください。

複数の作業ディレクトリ

SAS は、Work ライブラリを複数のディレクトリにわたって配布することで、Work ライブラリの配布を動的にすることができます。この機能は、単独のボリュームをすべての Work ディレクトリで埋めてしまう可能性を排除します。

WORK システムオプションは、SAS が Work ライブラリの割り当てに使用できる PATHNAME 引数を持ちます。この引数として、ディレクトリまたはディレクトリのリストを含むファイルを指定できます。ユーザー個別の Work ライブラリは 1 つのディレクトリ内に存在します。ユーザーは、構成ファイルまたはコマンド行で、WORK システムオプションを使用できます。

WORK への引数がファイル内のディレクトリのリストであるときは、WORK に使用するディレクトリの選択方法を指定できます。METHOD=RANDOM を指定する場合は、使用可能なディレクトリのリストから、ランダムにディレクトリが選択されます。METHOD=SPACE を選択する場合は、使用可能な空き領域が最大であるディレクトリが選択されます。

詳細については、“WORK システムオプション: UNIX” (434 ページ)を参照してください。

1 レベル名を使用した永久ファイル(ユーザーライブラリ)へのアクセス

1 レベル名について

SAS データセットは、1 レベル名または 2 レベル名を使用して参照されます。2 レベル名は *libref.member-name* という形を持っています。ここで、*libref* はデータセットが常駐している SAS ライブラリ、*member-name* はそのライブラリ内の特定の *member* を示します。1 レベル名は、*member-name* (*libref* はなし)という形をとります。この場合、ファイルは一時 Work ライブラリに保存されます。このアクションよりも優先して、ファイルを 1 レベル名で永久ライブラリに格納するためには、最初に User ライブラリ参照名を既存のディレクトリへ割り当てておく必要があります。User が割り当てられているときに一時 SAS ファイルを参照するには、2 レベル名をライブラリ参照名として WORK に使用します。

User ライブラリ参照名の割り当て法

User ライブラリ参照名を割り当てるには 3 つの方法があります。

- LIBNAME ステートメントを使用して、User ライブラリ参照名ディレクトリを割り当てます。

  ```
  libname user '/users/myid/mydir';
  ```

- SAS セッションを開始する前に、USER=システムオプションを指定します。たとえば、SAS を起動するときに、User ライブラリ参照名を割り当てることができます。

  ```
  sas -user /users/myid/mydir
  ```

- SAS セッションを開始した後に、USER=システムオプションを指定します。まず、ライブラリ参照名を永久ライブラリに割り当てます。次に、OPTIONS ステートメントで

USER=システムオプションを使用して、そのライブラリ参照名を User と同等にします。たとえば、これらのステートメントは、ライブラリ参照名 User をライブラリ参照名 *mine* を持つディレクトリに割り当てます。

```
libname mine '/users/myid/mydir';
options user=mine;
```

USER システムオプションの詳細については、“USER システムオプション: UNIX” (432 ページ)を参照してください。

注: UNIX 版 SAS では、USER 環境変数を使用した User ライブラリ参照名の割り当てはサポートされていません。

UNIX 環境におけるディスク形式のライブラリへのアクセス

ディスク上のライブラリは、他のタイプのライブラリよりも作成頻度とアクセス回数が多くなります。デフォルトのエンジンと互換エンジンは、ディスク上の SAS ファイルに対する読み込み、書き込みおよび更新アクセスを許可します。またこれらのエンジンは、インデックス付けとオブザベーションの圧縮をサポートしています。

次の例では、Stats1 データセットが含まれるディレクトリに In ライブラリ参照名が割り当てられています。

```
libname in '/users/myid/myappl';
proc print data=in.stats1;
run;
```

このディレクトリに対して読み込みまたは書き込みを行うには、まず *SAS-data-library* が存在してる必要があります。たとえば、ディレクトリ内に SAS データセット Orders を作成する場合は、X ステートメントを使用して **mkdir**UNIX コマンドを発行します。次に、LIBNAME ステートメントを使用して、ライブラリ参照名を次のディレクトリに関連付けます。

```
x mkdir /users/publish/books;
libname books '/users/publish/books';
data books.orders;
... more SAS statements ...
run;
```

デフォルトでは、LIBNAME ステートメントは V9 エンジンをディレクトリに関連付けます。

UNIX 環境における順次形式のライブラリへのアクセス

順次エンジンの利点と制限

順次エンジンにより、ディスク上のライブラリに順次形式でアクセスできます。順次エンジンは、インデックスおよびオブザベーションの圧縮をサポートしていません。

注: 順次エンジンを使用する前に、“Sequential Data Libraries” (*SAS Language Reference: Concepts* 24 章)で順次形式のライブラリに関する情報を参照してください。

名前付きパイプに順次データセットを書き込む

名前付きパイプを使用する理由

名前付きパイプを使用することで、動作環境に出力を送信したり、そこから入力を読み込んだりできます。たとえば、中間ファイルを作成せずに、データセットを圧縮したり、それを順次アクセス管理システムへ送信したりできます。

LIBNAME ステートメントの構文

ユーザーは、LIBNAME ステートメントでパイプ名を指定することにより、自分の SAS セッション内から名前つきパイプに対して読み込みと書き込みを実行できます。

LIBNAME *libref* '*pipename*';

パイプファイルの位置付けはできないため、順次エンジンを用いて順次アクセスが確保されます。エンジン名を指定する必要はありません。

例: 名前付きパイプを使用した SAS データセットの作成

非圧縮の中間ファイルを生成することなく SAS データセットを作成し、そのデータセットを圧縮するには、名前付きパイプ(**mypipe** など)を作成し、**compress** コマンドを入力します。

```
mkfifo mypipe p compress <mypipe >sasds.Z
```

SAS セッションで、ライブラリ参照名をパイプに割り当て、データセットへの書き込みを開始します。

```
libname x 'mypipe';
data x.a;
...more SAS statements...
output;
run;
```

データは **mypipe** へ送信され、圧縮されてから、データセットに書き込まれます。SAS がデータセットを閉じると、圧縮が完了して、圧縮された順次データセット **sasds.Z** が作成されます。

他方のタスク(このケースでは **compress** コマンド)が読み込みを開始する前に名前付きパイプへの書き込みを開始すると、そのタスクが読み込みを開始するまで SAS セッションは保留されます。

UNIX 環境で BMDP、OSIRIS、SPSS ファイルにアクセスする

BMDP、OSIRIS、SPSS ファイルについて

SAS は、SAS プログラムから外部データへの直接アクセスを可能にする 3 つのインターフェイスライブラリエンジン(BMDP、OSIRIS、SPSS)を搭載しています。これらのエンジンはすべて読み込み専用です。

いずれも順次エンジンであるため、SET ステートメントでは POINT=オプションを使用できません。また、FSBROWSE、FSEDIT または FSVIEW プロシジャとの併用もできません。PROC COPY、PROC DATASETS または DATA ステップを使用して、BMDP または OSIRIS システムファイルまたは SPSS エクスポートファイルを SAS データセッ

トにコピーし、次いで SAS データセット上でそれらの関数を実行できます。また、一部のプロシジャ(PROC、PRINT など)は、順次エンジンが使用されていることについて警告メッセージを発します。

順次エンジンの場合、ライブラリ参照名に関連付けられている物理的なファイル名は実際のファイル名であり、ディレクトリではありません。この関連付けは、ライブラリ参照名に関するルールにおける例外です。

CONVERT プロシジャを使用して、BMDP、OSIRIS および SPSS ファイルを SAS ファイルに変換できます。詳細については、“CONVERT プロシジャ: UNIX” (305 ページ)を参照してください。

BMDP Engine

BMDP Engine について

BMDP インターフェイスライブラリエンジンにより、BMDP 統計ソフトウェアアプリケーションの BMDP ファイルを、SAS プログラムから直接読み込むことができます。BMDP Engine は読み込み専用エンジンです。次のセクションでは、ユーザーが BMDP 保存ファイルの用語を使い慣れていることを前提としています。詳細については、Web サイト上の BMDP Statistical Solutions で提供されているドキュメントを参照してください。

注: このエンジンは AIX、HP-UX および Solaris で使用できます。

BMDP 保存ファイルへのアクセス構文

BMDP 保存ファイルを読み込むには、BMDP Engine を明示的に指定する LIBNAME ステートメントを発行します。この場合、LIBNAME ステートメントは次の形をとります。

LIBNAME *libref* BMDP '*filename*';

引数をここで説明します。

libref
SAS ライブラリ参照名を指定します。

filename
BMDP の物理ファイル名を指定します。

注: ライブラリ参照名がファイル参照名として先に表示される場合は、*filename* を省略します。SAS ではファイル参照名に関連付けられた物理ファイル名が使用されるためです。

このエンジンは UNIX 上で作成された保存ファイルのみを読み込むことができます。

単独の物理ファイルには複数の保存ファイルを含めることができるので、ユーザーは SAS 言語内のデータセットのメンバ名として CODE=値を参照します。たとえば、保存ファイルに CODE=ABC および CODE=DEF が存在し、ライブラリ参照名が MyLib である場合、ユーザーは MyLib.ABC および MyLib.DEF としてファイルを参照します。すべての CONTENT タイプは同じ扱いになります。DEF というメンバが CONTENT=CORR という値を持っている場合でも、その値が CONTENT=DATA だったかのように扱われます。

物理ファイル内の最初の save ファイルにアクセスしようと思っている場合、または save ファイルが 1 件しかない場合は、メンバ名を_FIRST_として参照します。この参照は、CODE=値が不明なときに便利です。

例: BMDP Engine

物理ファイル mybmdp.dat に保存ファイル ABC が含まれていると想定します。次の SAS コードはライブラリ参照名 mylib を BMDP 物理ファイルに関連付け、保存ファイル上で CONTENTS プロシジャおよび PRINT プロシジャを実行しています。

```
libname mylib bmdp 'mybmdp.dat';
proc contents data=mylib.abc;
run;

proc print data=mylib.abc;
run;
```

次の例では、LIBNAME ステートメントを使用して、ライブラリ参照名 mylib2 を BMDP 物理ファイルに関連付けています。物理ファイル内の最初の保存ファイルにデータが書き込まれます。

```
libname mylib2 bmdp 'mybmdp.dat';
proc print data=mylib2._first_;
run;
```

OSIRIS Engine

OSIRIS Engine について

ICPSR (Inter-University Consortium for Political and Social Research)は OSIRIS ファイル形式をデータファイルの配信用に使用しています。SAS は、ICPSR データの多くのユーザーをサポートし、PROC CONVERT との互換性を保つために、OSIRIS インターフェイスライブラリエンジンを搭載しています。

OSIRIS Engine を使用することで、ユーザーは OSIRIS データおよびディクショナリファイルを、SAS プログラムから直接読み込むことができます。次のセクションでは、ユーザーが OSIRIS ファイルの用語と構造を熟知していることを前提としています。そうでない場合は、ICPSR のドキュメントを参照してください。

OSIRIS データディクショナリファイルについての注

OSIRIS ソフトウェアは z/OS 環境以外では実行できないため、OSIRIS データディクショナリのレイアウトは、動作環境全体で一貫性があります。ただし、OSIRIS Engine は、SAS が実行されている他の動作環境にあるデータディクショナリは受け付けるようになっています。OSIRIS Engine は EBCDIC データを予期しているため、ディクショナリおよびデータファイルが EBCDIC から ASCII へ変換されないことが重要です。

ディクショナリファイルは、長さ 80 の固定長レコードで構成されている必要があります。データファイルは、ディクショナリに記述されているデータを保持できるだけの大きさのレコードを含んでいる必要があります。

OSIRIS ファイルへのアクセス構文

OSIRIS ファイルを読み込むには、OSIRIS Engine を明示的に指定する LIBNAME ステートメントを発行します。この場合、LIBNAME ステートメントの構文は次の形をとります。

LIBNAME *libref* OSIRIS '*data-filename*' DICT='*dictionary-filename*';

引数をここで説明します。

libref
SAS ライブラリ参照名を指定します。

'*data-filename*'
データファイルの物理ファイル名を指定します。

ライブラリ参照名がファイル参照名としても表示される場合は、*data-filename* を省略します。

DICT='*dictionary-filename*'
ディクショナリファイルの物理ファイル名を指定します。*dictionary-filename* が環境変数またはファイル参照名である場合は、それを引用符で囲まないでください。DICT=オプションは必須です。

OSIRIS データファイルにはメンバ名がありません。したがって、ユーザーが希望するメンバ名を使用してください。

異なる複数のデータファイルと同じディクショナリファイルを使用するには、データファイルごとに個別の LIBNAME ステートメントを使用します。

例: OSIRIS Engine

次の例では、データファイルは**/users/myid/osr/dat** であり、ディクショナリファイル**/users/myid/osr/dic** です。この例では、ライブラリ参照名 mylib を OSIRIS ファイルに関連付けて、CONTENTS プロシジャおよび PRINT プロシジャを実行しています。

```
libname mylib osiris '/users/myid/osr/dat'
dict='/users/myid/osr/dic';
proc contents data=mylib._first_;
run;
proc print data=mylib._first_;
run;
```

SPSS Engine

SPSS Engine について

SPSS Engine は読み込み専用エンジンです。SPSS インターフェイスライブラリエンジンによって、SPSS エクスポートファイルのみを読み込むことができます。このエンジンは、SPSS-X ネイティブファイルを読み込みません。

SPSS エクスポートファイルへのアクセス構文

SPSS エクスポートファイルを読み込むには、SPSS Engine を明示的に指定する LIBNAME ステートメントを発行します。この場合、LIBNAME ステートメントの構文は次の形をとります。

LIBNAME *libref* SPSS '*filename*';

引数をここで説明します。

libref
SAS ライブラリ参照名を指定します。

'*filename*'
物理ファイル名を指定します。

注: ライブラリ参照名がファイル参照名としても表示される場合は、*filename* を省略します。SAS ではファイル参照名に関連付けられた物理ファイル名が使用されるためです。

エクスポートファイルは SPSS EXPORT コマンドで作成する必要があります。また、いずれかの動作環境に由来するものである必要があります。エクスポートファイルは、ユーザーの動作環境との間で、ASCII 形式を使用して転送する必要があります。バイナリ形式で転送された場合は、その他の動作環境では読み込むことができません。

SPSS-X ファイルには内部名がないため、任意のメンバ名で参照してください。エクスポートファイルの共通の拡張子は.por ですがこの拡張子は不要です。

SPSS には、システム欠損の欠損データとユーザー定義の欠損データがあります。SPSS Engine または PROC CONVERT を使用するときは、(ユーザー定義またはシステム欠損の)欠損値がシステム欠損値に変換されます。ユーザー定義の欠損値は、有効な値として記録される必要があります。データセットを変換する場合は、PROC FORMAT を使用して(たとえば、-1 を.A に、-2 を.B に)翻訳できます。

SPSS ファイルの再フォーマット

SAS では、変数全体の幅よりも小数点以下桁数が大きい数値出力形式の変数が含まれた SPSS ファイルは、使用できません。たとえば、SPSS ファイルに幅 17 で小数点以下桁数 35 桁の変数が含まれる場合、そのファイルで DATA ステップを実行するかテーブルビューアでそれを表示しようとすると、SAS はエラーを返します。SAS で SPSS ファイルを使用するには、変数を再フォーマットする必要があります。

小数点以下桁数を変数の幅におさまる値まで減らすことにより、変数を再フォーマットできます。次の例では、**revision=cat(format,formatl,'.2');**というステートメントが小数点以下桁数を 2 に変換しています。この値は、変数の幅を超えないように小数点以下桁数を減らします。

```
libname abc spss 'FILENAME.POR';
proc contents data=abc._all_ out=new;
run;

filename sascode temp;
data _null_;
set new;
file sascode
if formatd > formatl then do;
revision=cat(format,formatl,'.2');
put 'format' +1 name +1 revision ';' ;
end;
run;

data temp;
set abc._all_;
%inc sascode/source2;
run;
```

注: OPTIONS NOFMTERR ステートメントは、DATA ステップまたはテーブルビューアでデータセットを使用できないようにします。DATA ステップまたはテーブルビューアを使用するには、変数の幅より大きな 10 進桁数を持つ数値変数を再フォーマットする必要があります。

例: SPSS Engine

次の例では、ライブラリ参照名 mylib を物理ファイル**/users/myid/mydir/myspssx.por** に関連付け、CONTENTS プロシジャおよび PRINT プロシジャをエクスポートファイルで実行しています。

```
libname mylib spss '/users/myid/mydir/myspssx.por';
proc contents data=mylib._first_;
proc print data=mylib._first_;
run;
```

次の例では、FILENAME ステートメントがファイル参照名 mylib2 を**/users/myid/mydir/aspssx.por**SPSS 物理ファイルに関連付けており、LIBNAME ステートメントがライブラリ参照名を SPSS Engine に関連付けています。PRINT プロシジャは、ポータブルファイルからのデータを出力します。

```
filename mylib2 '/users/myid/mydir/aspssx.por';
```

```
libname mylib2 spss;
proc print data=mylib2._first_;
run;
```

UNIX 環境でのリンクのサポート

UNIX 環境の SAS では、ハードリンクおよびシンボリックリンクが使用できます。ユーザーは、SAS データセットまたは SAS カタログを指定するリンクを作成できます。SAS プログラム内でリンクを参照すると、データセットまたはカタログを検索するためにそのリンクが追跡されます。

たとえば、UNIX プロンプトで次のコマンドを入力することで、**/tmp** ディレクトリに**/home/user/mydata.sas7bdat** データセットへのシンボリックリンクを作成できます。

```
ln -s /home/user/mydata.sas7bdat /tmp/mydata.sas7bdat
```

次の SAS コードは、**/tmp** ディレクトリでシンボリックリンクを使用して、**mydata.sas7bdat** データセットを検索します。このコードはシンボリックリンクを変更しませんが、データセット内のデータをソートします。

```
libname tmp '/tmp';

proc sort data=tmp.mydata;
by myvariable;
run;
```

SAS ウィンドウ環境で実行中の場合は、**SAS Explorer** ウィンドウを使用して、特定のディレクトリ内で保存されているシンボリックリンクを表示できます。存在しない SAS ファイルを指定するシンボリックリンクはいずれも、ファイルサイズが **0.0KB** で、変更日は **31DEC59:19:00:00** になります。

注: SAS では、バージョンデータセットまたはインデックスを持つデータセットのリンクはサポートされません。

3 章

外部ファイルとデバイスの使用

UNIX 環境の外部ファイルとデバイスについて

SAS セッション中に、データやテキストを含む外部ファイル、または、データやテキストを保存するファイルを使用することもあります。このようなファイルの作成および維持は、他のアプリケーションまたは SAS で行います。SAS 内では、外部ファイルの作成、読み込み、書き込み、削除が可能です。

SAS セッションで外部ファイルを使用すると、次の機能を実行できます。

- INPUT ステートメントで読み込む生データの保持
- SAS プロシジャで作成したレポートの印刷版の保存
- SAS ステートメントを含む処理用ファイルの送信
- PUT ステートメントで書き込まれたデータの保存

SAS では、外部ファイルおよびデバイスは、入力ソースおよび出力のレシーバとしての役割も持ちます。この入力は、DATA ステップ、または SAS で処理する SAS ステートメント内のどちらかで読み込む生データとなります。出力は次のいずれかになります。

- このプログラムで生成されたメモおよびメッセージを含む SAS ログ
- SAS プロシジャのフォーマット済み出力
- DATA ステップにて PUT ステートメントで書き込まれたデータ

プリンタ、プロッタまたは各自の端末など、周辺機器も使用できます。UNIX では、I/O デバイスをファイルのように扱います。各デバイスは、特殊ファイルと呼ばれるファイルに関連していますが、このファイルは通常のディスクファイルとして扱います。特殊ファイルに書き込む場合、これに対応するデバイスが自動的に起動します。特殊ファイルはすべて、**dev** ディレクトリ内またはそのサブディレクトリ内に存在します。デバイスの使用方法はデバイスタイプによりさまざまですが、基本概念はすべて同じです。

UNIX では、I/O デバイスであるかのようにパイプを使用してオペレーティングシステムコマンドとデータをやり取りできます。

移送データライブラリを含む外部ファイルにアクセスする必要がある場合は、*SAS ファイルの移動とアクセス*を参照してください。

UNIX 環境で外部ファイルまたはデバイスにアクセスする

ファイル参照名の指定

外部ファイルまたは外部デバイスにアクセスするには、該当する SAS ステートメント内のパス名またはファイル参照名を指定する必要があります。

FILE
PUT ステートメントに対する現在の出力ファイルを指定します。

%INCLUDE
プログラムエディタからプログラムを送信する場合に実行する SAS ソーステートメントを含むファイルを含みます。

ヒント %INCLUDE を使用する場合の行の最大長は 6000 バイトです。

INFILE
INPUT ステートメントで読み込む外部ファイルを特定します。

SAS ステートメントでは、ファイルやデバイスの参照は次の 2 つの方法のどちらかで行ってください。

- 外部ファイルのパス名を特定してください。詳細については、“UNIX 環境でのパス名の指定” (70 ページ)を参照してください。
- 1 つのファイル参照名を、1 つのデバイス、1 つ以上のファイル、または 1 つのディレクトリに割り当て、そのファイル、ディレクトリまたはデバイスを参照する場合にこのファイル参照名を使用してください。

  ほとんどの場合にファイル参照名を使用します。

ファイル参照名について

ファイル参照名は、ファイルまたはデバイスに割り当てるニックネームです。ファイル参照名を割り当て、必要に応じてこれを使用します。ファイル参照名は次の条件下では特に役立ちます。

- パス名が長く、プログラム内で何回か指定する必要がある。
- パス名が変更される可能性がある。パス名を変更する場合は、ファイルの各参照ではなく、ファイル参照名を割り当てるステートメントのみを変更する必要があります。

Explorer の File Shortcuts ウィンドウのファイル参照名を FILENAME ステートメントまたは FILENAME 関数に割り当てることができます。また、ファイル参照名を環境変数として定義することによりファイル参照名を割り当てることも可能です。

注: FILENAME ステートメントと FILENAME 関数の詳しい説明については、“FILENAME Statement” (*SAS Statements: Reference*) および “FILENAME Function” (*SAS Functions and CALL Routines: Reference*)を参照してください。

UNIX 環境でのパス名の指定

パス名の指定ルール

FILE ステートメント、INFILE ステートメントまたは%INCLUDE ステートメントのパス名を指定すると、外部ファイルを直接参照できます。外部ファイルを直接参照するには、FILENAME ステートメント内のファイル参照名とパス名を指定し、FILE ステートメント、INFILE ステートメント、または%INCLUDE ステートメント内にあるそのファイル参照名を使用します。

ファイルの参照が直接的でも間接的でも、適当なステートメント内のパス名を指定する必要があります。またほとんどの場合、名前を引用符で囲む必要があります。たとえば、次の INFILE ステートメントではファイル **/users/pat/cars** を参照します。

```
infile '/users/pat/cars';
```

次の FILE ステートメントでは、出力を指定した特別デバイスファイルに向けます。

```
file '/dev/ttyp1';
```

注: ファイル名の先頭に空白があれば、その空白は削除されます。

指定のレベルは、ユーザーのカレントディレクトリによって異なります。表 2.3 (52 ページ) の置換文字を使ってパス名を指定できます。また、“ワイルドカードのパス名への使用(入力のみ)” (71 ページ)で説明されているように、ワイルドカードを使用することもできます。

ファイル名の引用符を省略する

次のうちのいずれかを満たす場合は引用符を省略できます。

- ファイル名で定義されたファイル参照名がまだ存在しません。
- 対象ファイルには、ファイル参照に使用しているステートメントが予期するファイル名拡張子がついています。ファイル名を引用符で囲まない場合、FILE ステートメントと INFILE ステートメントが予測するファイル拡張子は.dat となり、%INCLUDE が予測するファイル拡張子は.sas になります。
- 対象ファイルはカレントディレクトリ内に格納されています。
- ファイル名が小文字で書かれています。

たとえば、カレントディレクトリが **/users/mkt/report** であり、そこにファイル **qtr.sas** が含まれている場合、次のステートメントのいずれかで **qtr.sas** を参照できます。

```
%include '/users/mkt/report/qtr.sas';
%include 'qtr.sas';
file 'qtr.sas';
```

qtr ファイル参照名が未定義の場合は、%INCLUDE ステートメントで引用符とファイル名拡張子を省略できます。

```
%include qtr;
```

大文字小文字混在または大文字のファイル名の操作

UNIX オペレーティングシステムでは、ファイル名の大文字と小文字は区別されます。つまり、ファイル名が **PROGRAM** であるファイルは、**program** というファイルとは同一ではありません。大文字と小文字の両方または大文字のみを使用しているファイル名を参照する場合に、そのファイル名が引用符で囲まれていなければ、そのファイル名は SAS により小文字に変換されます。そのファイル名にファイル拡張子が付いていない場合は、SAS により必要なファイル拡張子が追加されます。

たとえば、各自のプログラムで**%include code(PROGRAM);**を指定すると、ファイル名 PROGRAM は SAS により小文字に変換され、拡張子.sas が追加され、PROGRAM は **program.sas** となります。

SAS ログのメッセージの解釈

次のプログラムを実行する場合、SAS により **TEMP** は **temp** に変換され、拡張子.sas がファイル名に追加されます。

```
filename inc_code 'your-directory';
%include inc_code(TEMP);
```

SAS で次のメッセージが SAS ログに書き込まれます。

```
WARNING: Physical file does not exist, A.../your-directory/TEMP.sas.
ERROR: Cannot %INCLUDE member TEMP in the aggregate INC_CODE.
```

警告メッセージには、小文字に変換されたもの(temp.sas)ではなくもとのファイル名(TEMP.sas)のみが表示されます。TEMP.sas というファイルが存在しない場合、この状況に混乱をきたす場合があります。

このような混乱を避けるためには、ファイル名に拡張子が含まれている場合はファイル拡張子がついたファイル名を含めてください。ファイル名に拡張子がなければ、大文字小文字の両方または大文字のみのファイル名を引用符で囲んでください。次に例を示します。

```
%include code(TEMP.sas);
%include code("TEMP");
```

いずれの表記も、SAS では TEMP は小文字に変換されません。

ワイルドカードのパス名への使用(入力のみ)

有効なワイルドカードについて

*、?、[ ]というワイルドカードを用いて、FILENAME ステートメント(ファイル参照名を入力に使用する場合のみ)内、INFILE ステートメント内および%INCLUDE ステートメント内のパス名と、INCLUDE コマンドを指定できます。

*
: 1 つ以上の文字に付きますが、ファイル名の冒頭のピリオドには付きません。

?
: 1 文字に付きます。

[ ]
: 角かっこ内で定義された文字セットのうちの 1 文字に付きます。最初の文字と、ハイフンで分割された最後の文字を指定すると、文字範囲を指定できます。

ワイルドカード対応となるのは入力時のみです。FILE ステートメントではワイルドカードを使用できません。

例 1: ワイルドカードの文字列への挿入によるファイル選択

次の例では、カレントディレクトリ内にあり、文字列 **wild** で始まり.dat で終わる各ファイルから入力を読み込みます。

```
filename wild 'wild*.dat';
data;
infile wild;
input;
run;
```

例 2: カレントディレクトリ内の各ファイルの読み込み

次の例では、現在のワーキングディレクトリの各サブディレクトリにある各ファイルから入力を読み込みます。

```
filename subfiles '*/*';
data;
infile subfiles;
input;
run;
```

サブディレクトリのどれかに新規ファイルを追加する場合は、FILENAME ステートメントを変更することなく、Subfiles というファイル参照名でその新規ファイルにアクセスできます。

例 3: 集計構文使用時のファイル名のワイルドカード

集計構文を用いる場合、ファイル名にはワイルドカードを使用できますが、ディレクトリ名には使用できません。

```
filename curdir ".";
data;
infile curdir('wild*');
input;
run;
```

この例では、FILENAME ステートメントのピリオドはカレントディレクトリを示します。

UNIX で使用できる置換文字については、“パス名の有効な文字の置換” (52 ページ)を参照してください。

例 4: ファイル参照名の複数のファイルへの関連付け

次のステートメントにより、ファイル参照名 MyRef をアルファベット文字で始まる全ファイルに関連付けます。数字またはピリオドやチルダなどの文字で始まるファイルは対象外です。

```
filename myref '[a-zA-Z]*.dat';
```

次のステートメントにより、MyRef を、Sales で始まるファイル(大文字のみ、小文字のみ、大文字小文字混在のいずれかで表記)と年(2010 - 2019)に関連付けます。

```
filename myref '[Ss][Aa][Ll][Ee][Ss]201[0-9].dat';
```

FILENAME ステートメントを使用し、ファイル参照名を外部ファイルまたはデバイスに割り当てる

FILENAME ステートメントについて

外部ファイルまたは外部デバイスにファイル参照名を割り当てる最も一般的な方法は、FILENAME ステートメントを利用する方法です。FILENAME ステートメントにはいくつかの形式があり、アクセスするデバイスタイプにより使い分けます。詳細については、“FILENAME ステートメント: UNIX” (331 ページ)を参照してください。

DISK ファイルへのアクセス

FILENAME ステートメント最も一般的な用途は、DISK ファイルへのアクセスです。DISK ファイルに対する FILENAME 構文は次のようになります。

FILENAME *fileref* <DISK> *'pathname'* <*options*>;

次の FILENAME ステートメントは、ファイル参照名 **myfile** を外部ファイル **/users/mydir/myfile** に関連付けます。この外部ファイルはディスクデバイスに保存されます。

```
filename myfile disk '/users/mydir/myfile';
```

次の FILENAME ステートメントはファイル参照名 **prices** をファイル **/users/pat/cars** に割り当てます。その後の FILE ステートメントはこのファイル参照名を使用しているファイルを示します。

```
filename prices '/users/pat/cars';
data current.list;
file prices;
...PUT statements...
run;
```

DISK ファイルの使用については、“UNIX 環境でファイル名を連結する” (75 ページ)を参照してください。

注: ファイル名の先頭に空白があれば、その空白は削除されます。

DUMMY デバイスを使用したコードのデバッグ

DUMMY デバイスタイプを他のデバイスタイプと置換できます。このデバイスタイプは、実際に SAS コードを読み込むこともそのデバイスに書き込むこともなくデバッグするツールとして機能します。デバッグ完了後は、DUMMY デバイス名を適切なデバイスの種類に置き換えると、指定したデバイスタイプにプログラムがアクセスします。

次に示すのは、DUMMY ファイルに対する FILENAME 構文です。

FILENAME *fileref* DUMMY *'pathname'* <*options*>;

DUMMY デバイスへの出力は破棄されます。

出力を PRINTER デバイスを送る

PRINTER デバイスタイプにより、出力を直接プリンタへ送信できます。次に示すのは、ファイルを PRINTER に送る FILENAME 構文です。

FILENAME *fileref* PRINTER '<*printer*> <*printer-options*>' <*options*>;

たとえば、この SAS プログラムは、出力ファイルを BLDG3 プリンタに送ります。

```
filename myfile printer 'bldg3';

data test;
file myfile;
put 'This will appear in bldg3 .';
run;
```

詳細については、“ウィンドウのコンテンツの印刷” (97 ページ) および “UNIX 環境で PRINTTO プロシジャを使用する” (98 ページ)を参照してください。

一時ファイルの使用(TEMP デバイスタイプ)

TEMP デバイスタイプにより、Work ライブラリと同じディレクトリに保存されている一時ファイルにファイル参照名を関連付けます。(“Work ライブラリ” (58 ページ)を参照してください。) TEMP デバイスタイプを使用すると、SAS セッションの間だけ持続するファイルを作成できます。

次に示すのは、TEMP ファイルに対する FILENAME 構文です。

FILENAME *fileref* TEMP <*options*>;

たとえば、この FILENAME ステートメントでは、Tmp1 を一時ファイルに関連付けます。

```
filename tmp1 temp;
```

TERMINAL デバイスへのダイレクトアクセス

端末に直接アクセスするには、TERMINAL デバイスタイプを使用してください。次に示すのは、ファイルを端末に関連付ける FILENAME 構文です。

FILENAME *fileref* TERMINAL <*'terminal-pathname'*> <*options*>;

terminal-pathname は、端末に関連付けられる特殊ファイルのパス名である必要があります。詳細は UNIX システム管理者に確認してください。名前は引用符で囲んでください。端末のパス名を省略する場合は、端末にファイル参照名が割り当てられます。

たとえば、この FILENAME ステートメントは、ファイル参照名 **here** を次の端末に関連付けています。

```
filename here terminal;
```

次の FILENAME ステートメントはファイル参照名 **thatfile** を別の端末に関連付けています。

```
filename thatfile terminal '/dev/tty3';
```

他のシステムのファイルへのファイル参照名の割り当て(FTP、SFTP、SOCKET アクセスタイプ)

FTP アクセス方法、SFTP アクセス方法および SOCKET アクセス方法を利用すると、ネットワークの他のシステム上にあるファイルへのアクセスが可能です。次に示すのは、FILENAME ステートメントの形式です。

FILENAME *fileref* FTP *'external-file'* <*ftp-options*>;

FILENAME *fileref* SFTP *'external-file'* <*sftp-options*>;

FILENAME *fileref* SOCKET *'external-file'* <*tcpip-options*>;

FILENAME *fileref* SOCKET *':portno'* SERVER <*tcpip-options*>;

これらのアクセス方法は、*SAS ステートメント: リファレンス*に記載されています。UNIX では、FTP アクセス方法は追加オプションをサポートしています。

MACH='*machine*'
ユーザー名とパスワードを取得する場合に使用する `.netrc` ファイルのエントリを特定します。`.netrc` ファイルは、SAS プログラムが作動しているホスト上に存在します。`.netrc` ファイルの詳細については、UNIX man ページを参照してください。MACH オプションは、FILENAME ステートメントの HOST オプションと一緒には指定できません。

ファイルを z/OS 動作環境から UNIX に転送し、そのファイルのアクセスに S370V 形式または S370VB 形式のいずれかを使用する必要がある場合には、そのファイルを転送する前に、ファイルを RECFM=U および BLKSIZE=32760 にしておく必要があります。

注意:
FTP アクセス方法を使ってリモートファイルを作成する場合、そのファイルに対する UNIX の許可を-rw-rw-rw-に設定すると、そのファイルは、全ユーザーに対する読み込み許可および書き込み許可として設定されます。ファイル許可の変更については、`chmod` の UNIX man ページを参照してください。

UNIX 環境でファイル名を連結する

FILENAME ステートメント、%INCLUDE ステートメント、INFILE ステートメントのファイル名は連結できます。ファイル名を連結すると、連結したファイルを連続して読み込めます。

FILENAME *fileref* ("*pathname-1*" ... "*pathname-n*");

%INCLUDE '("*filename-1*" ... "*filename-n*")';

%INCLUDE "('*filename-1*' ... '*filename-n*')";

INFILE '("*filename-1*" ... "*filename-n*")';

INFILE "('*filename-1*' ... '*filename-n*')";

パス名は、単一引用符または二重引用符で囲んで、カンマまたは空白スペースで分割することができます。表 2.3 (52 ページ) に記載されている文字と “ワイルドカードのパス名への使用(入力のみ)” (71 ページ) で説明されているワイルドカードを使って、パス名を指定できます。

(集計構文を使用して)ディレクトリにファイル参照名を割り当てる

集計構文について

集計構文

集計構文により、ファイル参照名をディレクトリに割り当て、ファイル参照名の後の丸かっこ内のファイル名を指定して、ディレクトリ内のファイルを操作することができます。

FILENAME *fileref directory-name*;

集計構文は、1 つのディレクトリ内にある複数のファイルを参照する必要がある場合に、特に役立ちます。

例 1: 集計構文を利用したファイルの参照

ディレクトリ内のファイルを 1 つ参照するには、丸かっこ内の個別のファイル名に続くファイル参照名を指定してください。たとえば、この例で示されているように、ディレクトリ **/users/pat** のファイル cars.dat を参照できます。

```
filename prices '/users/pat';
data current.list;
file prices(cars);
...other SAS statements...
run;
```

例 2: 環境変数で定義されたファイル参照名がある集計構文の使用

環境変数を使って定義されているファイル参照名を含む集計構文も使用できます。("UNIX 環境で環境変数を使用してファイル参照名を割り当てる" (77 ページ)を参照してください。) たとえば、

```
x setenv PRICES /users/pat;
data current.list;
file prices(cars);
...other SAS statements...
run;
```

複数のディレクトリへのファイル参照名の割り当て

FILENAME ステートメントでは、デイレクトリ名を連結し、ファイル参照名を使ってこれらディレクトリ内のファイルを参照できます。

FILENAME *fileref* ("*directory-1*" ... "*directory-n*");

ディレクトリ名を連結すると、集計構文を使ってディレクトリのうちの 1 つにあるファイルを 1 つ参照できます。たとえば、Report.sas ファイルは MYPROGS 環境変数に関連付けられているディレクトリ内に存在すると予測します。SAS で次のコードを実行すると、FILENAME ステートメント内で指定されるパス名から Report.sas が検索され、プログラムが実行されます。

```
filename progs ("$MYPROGS" "/users/mkt/progs");
%inc progs(report);
```

SAS により、次の状況になるまで、FILENAME ステートメント内で指定された順にパス名を検索します。

- 指定した名前を含む最初のファイルが検出されるまで。ファイル名内のワイルドカード(“ワイルドカードのパス名への使用(入力のみ)” (71 ページ)を参照)を使用しても、SAS でマッチするファイルは 1 つだけです。
- FILENAME ステートメント内で指定したパス名の一覧の中のファイル名を引き当てるまで。

UNIX 環境で環境変数を使用してファイル参照名を割り当てる

変数名の要件

環境変数をファイル参照名として使用して、DISK ファイルを参照することも可能です。変数名はすべて大文字とし、変数値は必ず外部ファイルの完全なパス名とします。つまり、ファイル名はスラッシュで始まる必要があります。

注: 変数とファイル参照名に同じ名前が含まれているが、異なるファイルを参照する場合、SAS ではファイル参照名の方を使用します。たとえば、次の%INCLUDE ステートメントではファイル**/users/myid/this_one** を参照します。

```
filename ABC '/users/myid/this_one';
x setenv ABC /users/myid/that_one;
%include ABC;
```

データファイルの読み込み

データファイル**/users/myid/educ.dat** を読み込むが INED 環境変数でこれを参照する場合、変数を次のとおり 2 回定義できます。

- SAS を起動する前、“UNIX 環境で環境変数を定義する” (439 ページ)を参照してください。たとえば、Korn シェルではこれを使用します。

  ```
  export INED=/users/myid/educ.dat
  ```

- X ステートメント(“SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)を参照)と SAS の **setenv** コマンドを使用して SAS を起動した後。

  ```
  x setenv INED /users/myid/educ.dat;
  ```

INED がファイル**/users/myid/educ.dat** に関連付けられたら、**ined** をファイル参照名として使用し、INFILE ステートメント内の対象ファイルを参照できます。

```
infile ined;
```

外部ファイルへの書き込み

外部ファイルに書き込む場合は同一の方法を適用します。たとえば、SAS を起動する前に OUTFILE を定義できます。

```
OUTFILE=/users/myid/scores.dat
export OUTFILE
```

その後で、環境変数名をファイル参照名として使用し、対象ファイルを参照してください。

```
file OUTFILE;
```

UNIX 環境で SAS によって割り当てられるファイル参照名

標準入力、標準出力、標準エラーのファイル参照名

多くの場合、あるコマンドの引数またはオプションからそのコマンドに対し、入出力時に使用するものが伝えられますが、伝えられない場合は、入力用(標準入力)、出力用(標準出力)、エラーメッセージ用(標準エラー)の 3 つの標準ファイルをシェルから取得します。デフォルトでは、これらのファイルはすべて各自の端末に関連付けられており、具体的には、標準入力ファイルはキーボードに、標準出力ファイルと標準エラーファイルは端末のディスプレイに関連付けられています。SAS を起動すると、SAS により開かれた 3 つの標準ファイルそれぞれにファイル参照名が割り当てられます。SAS により、ファイル参照名である Stdin、Stdout、Stderr が、標準入力ファイル、標準出力ファイル、標準エラーファイルにそれぞれ割り当てられます。

ファイルディスクリプタ

ファイルディスクリプタについて

各ファイルには、そのファイルに割り当てられている内部ファイルディスクリプタがあります。デフォルトでは、0 は標準入力のファイルディスクリプタ、1 は標準出力のファイルディスクリプタ、2 は標準エラーのファイルディスクリプタです。他のファイルが開くと、別のファイルディスクリプタが割り当てられます。Bourne シェルと Korn シェルでは、“Bourne シェルと Korn シェルのファイルディスクリプタ” (78 ページ)で説明されているとおり、ファイルディスクリプタを使用して、ファイルから読み込む、またはファイルに書き込むデータを指定できます。

Bourne シェルと Korn シェルのファイルディスクリプタ

Bourne シェルまたは Korn シェルを使用する場合、SAS では、ファイルディスクリプタ(“UNIX 環境で SAS によって割り当てられるファイル参照名” (78 ページ)を参照)が 2 より大きいファイルに、次の形式のファイル参照名が割り当てられます。

```
FILDESnumber
```

number は、ファイルディスクリプタを 2 桁で表記したものです。SAS アプリケーションではこれらファイル参照名を使用できます。

たとえば、次のコマンドで SAS を起動すると、動作環境により sales_data ファイルが開き、ファイルディスクリプタの 4 がこのファイルに割り振られます。

```
sas salespgm 4< sales_data
```

SAS によりファイル参照名 FILDES04 がこのファイルに割り振られ、**salespgm** アプリケーションが実行されます。このアプリケーションは FILDES04 からの入力を読み込むときに、sales_data ファイルを読み込みます。ファイルディスクリプタをファイル参照名として使用すると、各ファイルを参照するアプリケーションは変更せずに同じアプリケーションを使って異なるファイルからのデータを処理できます。このアプリケーションを起動する場合に使用するコマンドでは、処理するファイルに適切なファイルディスクリプタが割り当てられます。

UNIX 環境の予約ファイル参照名

次のファイル参照名は予約されます。

INFILE ステートメント内の DATALINES ファイル参照名
DATALINES ステートメントの直後にインプットデータがくるように指示します。INFILE ステートメントのオプションを指定してインストリームデータを読み込む場合のみ、INFILE DATALINES を使用する必要があります。

FILE ステートメント内の LOG ファイル参照名
PUT ステートメントで生成する出力行が SAS ログに書き込まれるように指定します。LOG は出力行のデフォルトの宛先です。

FILE ステートメント内の PRINT ファイル参照名
PUT ステートメントで生成する出力行が、SAS プロシジャで生成する出力と同じ印刷ファイルに書き込まれるように指示します。

UNIX 環境で外部ファイルを共有する

外部ファイルの共有

1 名以上のユーザーが、外部ファイルに同時に Write アクセスできる場合、または、単独ユーザーが異なる SAS セッションからの同一ファイルに Write アクセスできる場合、そのファイルの共有を予測できなくなります。このような状況を改善するには、ステートメントまたはシステムオプションを利用して Write アクセスできるユーザーを 1 名に限定し、複数のユーザーには Read アクセスを許可します。詳細については、“SAS ファイルの共有” (41 ページ)を参照してください。

ファイルロックに使用するオプション: 外部ファイル

ファイルロックは、開いている全てのファイルに適用されます。外部ファイルに対するファイルロックを無効にするには、次の方法を実行してください。

- FILENAME ステートメントの LOCKINTERNAL オプションを使用します。
- FILELOCKS システムオプションを使用します。

外部ファイルのファイルロック: LOCKINTERNAL ステートメントオプション

FILENAME ステートメントの LOCKINTERNAL オプションを使用して、外部ファイルへのファイルロックを管理できます。AUTO オプション値により、Write アクセス専用のファイルまたは Read アクセス専用でないファイルをロックします。たとえば、更新目的または出力目的でファイルを開くと、内部プロセスからの他のすべてのアクセスがブロックされます。入力目的でファイルを開くと、他のユーザーもそのファイルを入力目的で開くことができます。この場合、そのファイルを更新目的や出力目的で開くとブロックされます。SHARED オプション値で、AUTO オプションの全ての動作が可能になります。ただし、そのファイルが 1 名のライタと複数のリーダーで共有できる場合はこれに該当しません。このファイル参照名に関連付けられている外部ファイルは、ロックされたファイルです。デフォルトでは、複数のユーザーが外部ファイルを同時に読み込めま

す。詳細については、“FILENAME ステートメント: UNIX” (331 ページ)を参照してください。

外部ファイルのファイルロック: FILELOCKS システムオプション

FILELOCKS システムオプションを使用して、(SAS ファイルのファイルロックだけでなく)外部ファイルのファイルロックを管理できます。このオプションにより、個別のファイルまたはディレクトリに対し動作を包括的に適用できます。FILELOCKS を使用すると、ライタアクセスが 1 名のユーザーに限定されます。ファイルロックが有効の場合、複数の SAS セッションで同時に同一ファイルを読み込めます。起動時に OPTIONS ステートメントまたはコマンド行で FILELOCKS を使用できます。FILELOCKS オプションの複数のインスタンスを指定できます。各インスタンスは、パスおよび設定の内部テーブルに追加されます。詳細については、“FILELOCKS システムオプション: UNIX” (377 ページ)を参照してください。

UNIX コマンド(PIPE)からの読み込みと書き込み

パイプについて

パイプにより、SAS アプリケーションで、標準出力に書き込む UNIX コマンドから入力を受け取ったり、標準入力から読み込む UNIX コマンドに出力したりできます。UNIX コマンドでは、パイプは縦棒(|)で表します。たとえば、各自のディレクトリ内のファイル数を調べるには、`wc`(単語数)コマンドへのパイプを通じて `ls` コマンドの出力をリダイレクトします。

```
ls | wc -w
```

ファイル参照名をパイプに割り当てる FILENAME ステートメントの構文

UNIX では、FILENAME ステートメントを使用して、外部ファイルや I/O デバイスだけでなくパイプにもファイル参照名を割り当てることができます。次に示すのは、FILENAME ステートメントの構文です。

FILENAME *fileref* PIPE '*UNIX-command*' <*options*>;

fileref
SAS からのパイプを参照する場合に使用する名前です。

PIPE
デバイスタイプを UNIX パイプとして特定します。

'*UNIX-command*'
UNIX コマンドの名前、実行可能なプログラムの名前、または出力を送るまたは入力を読み込むシェルスクリプトの名前です。このコマンドは、二重引用符または単一引用符で囲む必要があります。

options
外部ファイルの処理方法を管理します。これらオプションの説明については、“FILENAME ステートメント: UNIX” (331 ページ)を参照してください。

このコマンドを入力として使用しているのか出力としてなのかは、読み込み時または書き込み時に *fileref* を使用しているかどうかに依存します。たとえば、INFILE ステートメントでこのファイル参照名を使用すると、SAS では入力は UNIX コマンドから送信されると予測されます。FILE ステートメントでこのファイル参照名を使用すると、SAS では出力が UNIX コマンドへ送られると予測されます。

読み込みにファイル参照名を使用する

読み込み用ファイル参照名を指定する

ファイル参照名を読み込みのために使用する場合、指定された UNIX コマンドを実行し、ファイル参照名により、標準出力または標準エラーに送信された出力を読み込みます。この場合、このコマンドの標準入力は **/dev/null** に接続されます。

例 1: プロセスコマンドの出力の SAS の DATA ステップへの送信

次の SAS プログラムでは、PIPE デバイスタイプのキーワードを用いて、**ps**(プロセス)コマンドを SAS の DATA ステップに送信します。結果として生じる SAS データセットには、現在 SAS を実行している各プロセスに関するデータが含まれます。

```
filename ps_list pipe "ps -e|grep 'sas'";
data sasjobs;
infile ps_list;
length process $ 80;
input process $ char80.;
run;
proc print data=sasjobs;
run;
```

ps -e コマンドにより、システムのすべてのアクティブなプロセスの一覧が作成されます。この一覧にはタスクを開始したコマンドの名前などが含まれます。BSD ベースの UNIX システムでは、**ps -ax** コマンドを使用できます。

この動作環境では、パイプを使用して **ps** コマンドからの出力を grep コマンドに送信します。**grep** コマンドは文字列**'sas'**をすべて検索します。FILENAME ステートメントは、**grep** コマンドの出力をファイル参照名 **ps_list** に接続します。その後 DATA ステップでは、入力源を示す NFILE ステートメントから **sasjobs** というデータセットが作成されます。INPUT ステートメントにより、各入力行の最初の 80 文字を読み込みます。

例 2: ファイル参照名 Stdin を使用した入力読み込み

次の例では、ファイル参照名 Stdin を用いて、SAS コマンドへのパイプで入力を読み込み、SAS プログラムが実行されます。パイプ処理作業を SAS プログラムの外側に位置付けることで、このプログラムはさらに一般的になります。前述の例で示したプログラムは変更され、ファイル ps.sas に保存されました。

```
data sasjobs;
infile stdin;
length process $ 80;
input process $ char80.;
run;
proc print data=sasjobs;
run;
```

このプログラムを実行するには、パイプを使用して **ps** の出力を **grep** に送信し、**grep** から次の SAS コマンドに送信します。

```
ps -e|grep 'sas'|sas ps.sas &
```

出力は **ps.lst** に保存され、ログは **ps.log** に保存されます。詳細は、“UNIX 環境での、SAS ログと SAS プロシジャのデフォルトの出力先” (91 ページ)を参照してください。

書き込みへのファイル参照名の使用

書き込みへのファイル参照名の指定

ファイル参照名を書き込みに使用する場合、指定された UNIX コマンドにより SAS からの出力を読み込み、これを実行します。

例 1: パイプを使用したメール送信

この例では、ファイル参照名 **mail** に送信されたデータはすべて **mail** コマンドに流し込まれ、ユーザー PAT に送信されます。

```
filename mail pipe 'mail pat';
```

例 2: リモートシェルと印刷出力の開始

この FILENAME ステートメントを考えてみましょう。

```
filename letterq pipe 'remsh alpha lp -dbldga3';
```

letterq ファイル参照名に送信されたデータはすべて、UNIX コマンドに引き継がれ、Alpha というコンピュータ上のリモートシェルが開始されます。リモートシェルを開始するコマンドの形式は、UNIX オペレーティングシステムにより異なるので、ご注意ください。その後、シェルにより、印刷先 BLDGA3 で特定されるプリンタで **letterq** 出力を印刷します。**lp** コマンドが生成するメッセージはすべて SAS ログに送信されます。

FILENAME ステートメント(EMAIL)を使用し、電子メールを送信する

SAS から電子メールを送信する利点

SAS により、DATA ステップまたは SCL で SAS 機能を使用して電子メールを送信します。SAS から電子メールを送信する場合にできることは次のとおりです。

- DATA ステップのロジックまたは SCL を使って、電子メールアドレスの大きなデータセットに基づいた、電子メール送信先のサブセットを作成します。
- バッチ処理のために送信した SAS プログラムの完了時に、電子メールを自動送信します。
- 処理の結果に基づき、電子メールを通じて出力を示します。
- ユーザーインターフェイスをカスタマイズされるように SAS/AF フレームアプリケーションからの電子メールを送信します。

電子メールの初期化

デフォルトでは、SAS は SMTP(Simple Mail Transfer Protocol)を使用して電子メールを送信します。SMTP は、外部スクリプトとは違い、添付ファイルをサポートします。このデフォルトは、EMAILSYS システムオプションにより指定されます。電子メールプロトコルの変更方法については、“EMAILSYS システムオプション: UNIX” (375 ページ)を参照してください。

ユーザーが SAS からの電子メールを送信する前に、システム管理者は SMTP サーバーを示すように EMAILHOST システムオプションを設定する必要があります。詳細については、“EMAILHOST= System Option” (*SAS System Options: Reference*)を参照してください。

電子メールの送信に使用される DATA ステップまたは SCL コードのコンポーネント

一般に、電子メールを送信する DATA ステップまたは SCL コードには次のコンポーネントがあります。

- EMAIL デバイスタイプキーボードに対応する FILENAME ステートメント
- 電子メールの受信者、件名、添付ファイルを示す FILENAME ステートメントまたは FILE ステートメントで指定されたオプション
- メッセージの本文を含む PUT ステートメント
- 電子メール属性(TO、CC、BCC、SUBJECT、ATTACH)をオーバーライドできる特別な電子メールディレクティブ(形式は!EM_*directive*!)を含む、またはアクション(SEND、ABORT、NEWMSG の開始など)を実行する PUT ステートメント

電子メールの FILENAME ステートメントの構文

DATA ステップまたは SCL からの電子メールを送信するには、次の形式の FILENAME ステートメントを発行してください。

FILENAME *fileref* EMAIL '*address*' <*email-options*>;

FILENAME ステートメントは次のオプションを受け入れます。

fileref
有効なファイル参照名です。

'*address*'
電子メールの送信先となるユーザーの電子メールアドレスです。ここでアドレスを指定する必要がありますが、TO 電子メールオプションでその値をオーバーライドできます。

email-options
次のいずれかになります。

TO=*to-address*
電子メールの最初の受信者を指定します。アドレスが 2 つ以上の単語を含む場合は、アドレスを引用符で囲んでください。2 つ以上のアドレスを指定するには、アドレスのグループを丸かっこで囲み、各アドレスを引用符で囲んで、アドレスとアドレスの間にスペースを 1 つ入れて分離してください。たとえば、**to='joe@someplace.org'**および **to=("joe@smplc.org" "jane@diffplc.org")** は有効な TO 値です。

注: CC=オプションまたは BCC=オプションのどちらかで受信者を指定している限り、TO=オプションで受信者を指定しないで電子メールを送信できます。

CC=*cc-address*
電子メールのコピーを受け取る受信者を指定します。アドレスが 2 つ以上の単語を含む場合は、アドレスを引用符で囲んでください。2 つ以上のアドレスを指定するには、アドレスのグループを丸かっこで囲み、各アドレスを引用符で囲んで、アドレスとアドレスの間にスペースを 1 つ入れて分離してください。たとえば、**cc='joe@someplace.org'**およ

び **cc=("joe@smplc.org" "jane@diffplc.org")** は有効な CC 値です。

BCC=*bcc-address*
電子メールのブラインドコピーを受け取る受信者を指定します。**bcc** フィールドに記載される個人は、この電子メールのコピーを受け取ります。BCC フィールドは電子メールのヘッダーには表示されないので、他の受信者がこれらの電子メールアドレスを見ることはできません。

BCC アドレスが 2 つ以上の単語を含む場合は、アドレスを引用符で囲んでください。2 つ以上のアドレスを指定するには、アドレスのグループを丸かっこで囲み、各アドレスを引用符で囲んで、アドレスとアドレスの間にスペースを 1 つ入れて分離してください。たとえば、**bcc='joe@someplace.org'** および **bcc=("joe@smplc.org" "jane@diffplc.org")** は有効な BCC 値です。

SUBJECT='*subject*'
メッセージの件名を入力します。件名の長さが 2 単語以上(1 つ以上の空白スペースを含む)場合は、件名を引用符で囲む必要があります。また、件名に特殊文字が含まれている場合も引用符で囲みます。たとえば、**subject=Sales** および **subject='June Report'** は有効な件名です。引用符で囲まれていない件名はすべて大文字に変換されます。

ATTACH='*filename.ext*' | ATTACH = ('*filename.ext*' <*attachment-options*>)
メッセージに添付するファイルの物理的な名前と、添付の仕様を変更するオプションを指定します。*filename.ext* を引用符で囲んでください。2 つ以上のファイルを添付するには、ファイル名のグループを丸かっこで囲んでください。たとえば、**attach='/u/userid/opinion.txt'** および **attach=("june11.txt" "july11.txt")** は有効な添付ファイルです。

デフォルトでは、SMTP 電子メール添付ファイルの値は、LRECL のデフォルト値と同じ 32K です。さらに長い添付ファイルを送信するには、*attachment-options* として、FILENAME ステートメントから LRECL=オプションと RECFM=オプションを指定します。LRECL=オプションと RECFM=オプションについては、“FILENAME ステートメント: UNIX” (331 ページ)を参照してください。

SMTP の使用時に有効なオプションの詳細については、“FILENAME Statement, EMAIL (SMTP) Access Method” (*SAS Statements: Reference*)を参照してください。

FILE ステートメントの電子メールオプションの指定

DATA ステップ内の FILE ステートメントの *email-options* も指定できます。FILE ステートメントで指定するオプションは、FILENAME ステートメントで指定した対応するオプションをオーバーライドします。

メッセージ本文の定義

DATA ステップでは、FILE ステートメントを使用して電子メールのファイル参照名を出力先として定義してから、PUT ステートメントを使用してメッセージ本文を定義します。

PUT ステートメントの電子メールディレクティブの指定

PUT ステートメントを使用して、電子メールの属性を変更する、または電子メールに伴うアクションを実行する電子メールディレクティブも指定できます。1 つの PUT ステートメントで指定するディレクティブは 1 つのみで、各 PUT ステートメントに含まれるのは、指定されたディレクティブに関連付けられているテキストのみです。

次に、メッセージの属性を変更するディレクティブを示します。

!EM_TO! *addresses*
最初の受信者のアドレスを *addresses* に置き換えてください。PUT ステートメントでは、単一引用符がない *addresses* を指定してください。

!EM_CC! *addresses*
現在のコピー受信者のアドレスを *addresses* に置き換えてください。PUT ステートメントでは、単一引用符がない *addresses* を指定してください。

!EM_BCC! *addresses*
現在のブラインドコピー受信者のアドレスを *addresses* に置き換えてください。PUT ステートメントでは、単一引用符がない *addresses* を指定してください。

!EM_SUBJECT! *subject*
現在のメッセージ件名を *subject* に置き換えてください。

!EM_ATTACH! *pathname*
添付ファイルの名前を *pathname* に置き換えてください。

次に示すのは、アクションを実行するディレクティブです。

!EM_SEND!
現在の属性でメッセージを送信します。デフォルトでは、ファイル参照名のクローズ時に SAS がメッセージを送信します。次の FILE ステートメントの実行時または DATA ステップの終了時にファイル参照名はクローズします。このディレクティブを使用する場合、SAS はこのディレクティブを実行するときにメッセージを送信し、DATA ステップの最後にもう一度メッセージを送信します。

!EM_ABORT!
現在のメッセージを中止します。このディレクティブを使用して DATA ステップ終了時に SAS のメッセージの自動送信を中止できます。

!EM_NEWMSG!
TO、CC、SUBJECT、ATTACH、メッセージ本文などといった現在のメッセージの属性をすべてクリアします。

例: DATA ステップからの電子メールの送信

config.sas ファイルのコピーを同僚の Jim(ユーザー ID は JBrown)と共有すると仮定してください。電子メールプログラムでエイリアス名と添付ファイルを扱う場合は、次の DATA ステップを送信することにより電子メールを送信できます。

```
filename mymail email 'JBrown'
subject='My CONFIG.SAS file'
attach='config.sas';

data _null_;
file mymail;
put 'Jim,';
put 'This is my CONFIG.SAS file.';
put 'I think you might like the
new options I added.';
run;
```

次の例では、1 つのメッセージと 2 つの添付ファイルを複数の受信者に送信します。FILENAME ステートメントではなく FILE ステートメントで電子メールオプションを指定します。

```
filename outbox email 'ron@acme.com';

data _null_;
```

```
file outbox

/* Overrides value in filename statement */
to=('ron@acme.com' 'lisa@acme.com')
cc=('margaret@yourcomp.com'
'lenny@laverne.abc.com')
subject='My SAS output'
attach=('results.out' 'code.sas')
;
put 'Folks,';
put 'Attached is my output from the
SAS program I ran last night.';
put 'It worked great!';
run;
```

DATA ステップの条件付きロジックを使用して、複数のメッセージを送信し、どの受信者がどのメッセージを受け取るのかを管理できます。たとえば、2 つの異なる部署のメンバにカスタマイズしたレポートを送信するとします。電子メールプログラムでエイリアス名と添付ファイルを扱う場合、DATA ステップは次のようになります。

```
filename reports email 'Jim';

data _null_;
file reports;
infile cards eof=lastobs;
length name dept $ 21;
input name dept;

/* Assign the TO attribute */
put '!EM_TO!' name;

/* Assign the SUBJECT attribute */
put '!EM_SUBJECT! Report for ' dept;

put name ',';
put 'Here is the latest report for ' dept '.';

/* ATTACH the appropriate report */
if dept='marketing' then
put '!EM_ATTACH! mktrept.txt';
else

put '!EM_ATTACH! devrept.txt';

/* Send the message */
put '!EM_SEND!';

/* Clear the message attributes */
put '!EM_NEWMSG!';

return;

/* Abort the message before the */
/* RUN statement causes it to */
/* be sent again. */
lastobs: put '!EM_ABORT!';
```

```
datalines;
Susan marketing
Jim marketing
Rita development
Herb development
;
run;
```

得られる電子メールのメッセージと添付ファイルは、受信者が所属する部署によって異なります。

注: !EM_NEWMSG! ディレクティブを使用して、受信者間のメッセージ属性をクリアする必要があります。!EM_ABORT! ディレクティブは、メッセージが DATA ステップの終了時に自動送信されないようにします。

例: SCL コードそ使用した電子メールの送信

次の例は、電子メールのフレームエントリデザインの後ろにある SCL コードです。フレームエントリには、ユーザーが情報を入力できる複数のテキストエントリフィールドが含まれています。

mailto
メール送信用のユーザー ID

copyto
メールコピー(CC)用のユーザー ID

attach
添付するファイルの名前

subject
メールの件名

line1
メッセージのテキスト

フレームエントリーには、この SCL コード(**send:**ラベルがついている)を実行する SEND というボタンも含まれています。

```
send:

/* set up a fileref */
rc = filename('mailit','userid','email');

/* if the fileref was successfully set up
open the file to write to */
if rc = 0 then do;
fid = fopen('mailit','o');
if fid > 0 then do;

/* fput statements are used to
implement writing the
mail and the components such as
subject, who to mail to, and so on. */
fputrc1 = fput(fid,line1);
rc = fwrite(fid);

fputrc2 = fput(fid,'!EM_TO! '||mailto);
rc = fwrite(fid);
```

```
fputrc3 = fput(fid,'!EM_CC! '||copyto);
rc = fwrite(fid);

fputrc4 = fput(fid,'!EM_ATTACH! '||attach);
rc = fwrite(fid);
fputrc5 = fput(fid,'!EM_SUBJECT! '||subject);
rc = fwrite(fid);

closerc = fclose(fid);
end;
end;
return;

cancel:
call execcmd('end');
return;
```

4 章

出力の印刷と出力先指定

UNIX 環境における出力印刷の概要

テキストまたはグラフィックを印刷する場合、SAS は、その出力先、書き込み方法、およびその出力の表示を知る必要があります。ユニバーサル印刷は、UNIX におけるデフォルトの印刷の仕組みです。ユニバーサル印刷は、すべての環境内の PostScript、PCL、GIF、PNG、SVG、EMF、および PDF ファイルをサポートしています。ユニバーサル印刷についての詳細は、“Universal Printing” (*SAS Language Reference: Concepts* 15 章)を参照してください。

フォーム印刷は、SAS で利用可能な古いテキスト印刷方法です。これには、**Form** ウィンドウで構成される FORM サブシステムを使用する必要があります。詳細は、“Forms Printing” (*SAS Language Reference: Concepts* 15 章)を参照してください。

グラフィックを印刷する場合、出力は、ネイティブの SAS/GRAPH ドライバにより制御されます。SAS/GRAPH のオンラインヘルプでネイティブ SAS/GRAPH ドライバについて詳細を確認してください。

UNIX 環境での出力のプレビュー

ユニバーサル印刷を使用して出力をプレビューする

ユニバーサル印刷では、プリンタ、プロッタ、または外部ファイルに送信する前に出力をプレビューできます。出力をプレビューするには、まず、ご使用のシステムのプレビューアを定義する必要があります。詳細については、“Universal Printing” (*SAS Language Reference: Concepts* 15 章)を参照してください。

SAS/AF アプリケーションからの出力のプレビュー

SAS/AF アプリケーション内からの出力をプレビューするには、DMPRTMODE および DMPRTPREVIEW コマンドでプレビューモードを有効にし、出力を印刷し、**Print Preview** ダイアログボックスを開いた後、プレビューモードを無効にします。たとえば、次のコードは、ホストドライバを使って GRAPH1 オブジェクトを印刷し、それを **Preview** ダイアログボックスで表示します。

```
/* Turn on preview mode. */
CALL EXECCMDI ("DMPRTMODE PREVIEW");

/* Print the graph */
GRAPH1._PRINT_();

/* Open the Preview dialog box */
CALL EXECCMDI ("DMPRTPREVIEW");

/* Turn off preview mode */
CALL EXECCMDI ("DMPRTMODE NORMAL");
```

UNIX 環境での、SAS ログと SAS プロシジャのデフォルトの出力先

各 SAS ジョブまたはセッションについて、SAS は自動的に 2 種類の出力を作成します。

SAS ログ
: SAS ステートメントの処理に関する情報を含んでいます。各プログラムステップが実行されると、該当エラーまたは警告メッセージと一緒に、SAS ログに注意が書き込まれます。

SAS 出力
: プロシジャ出力ファイルまたは印刷ファイルとも呼ばれます。SAS プログラムが印刷出力を生成する PROC ステップまたは DATA ステップを実行するたびに、SAS は、SAS 出力ファイルに出力を送信します。SAS 出力のデフォルトの出力先は、HTML です。

次のテーブルは、SAS ログおよび出力ファイルのデフォルトの出力先を示しています。

表 4.1 *SAS ログおよび出力ファイルのデフォルトの出力先*

処理モード	SAS ログファイル	SAS 出力ファイル
バッチ	*filename*.log	*filename*.lst
ウィンドウ環境	**Log** ウィンドウ	HTML
対話型ライン	ターミナル	ターミナル

デフォルトでは、ログファイルと出力ファイルの両方が、使用中のカレントディレクトリに書き込まれます。システム管理者がこれらのデフォルトの出力先を変更している場合があります。

UNIX 環境でデフォルトの出力先を変更する

出力先の指定方法

出力先を指定する主な方法として、次の 5 つあります。

- デフォルトの HTML 回付先の使用
- **Print** ダイアログボックスの使用 **Print** ダイアログボックスは、SAS ウィンドウ環境の使用時に利用できます。
- ウィンドウ環境コマンドの発行 PRTFILE、PRINT、および FILE コマンドは、コマンド行から発行でき、外部ファイルへ、または FILENAME ステートメントで定義された他のデバイスへ出力を送信するのに使用できます。
- PRINTTO プロシジャの使用 PRINTTO プロシジャはどのモードでも使用できます。PRINTTO プロシジャとともに FILENAME ステートメントを使用することが出力先の指定の最もフレキシブルな方法です。

- PRINT、LOG、ALTPRINT、または ALTLOG などの SAS システムオプションを使用した代替割り当て先の指定

出力先の変更に使用する手法の決定

次のテーブルを使用すると、出力先の変更にどの方法を選択するかを容易に決定できます。

表 4.2 デシジョンテーブル: デフォルトの割り当て先の変更

SAS ログまたはプロシジャ出力の出力先	処理モード	方法	参照
プリンタ	すべてのモード	FILENAME ステートメント (UPRINTER または PRINTER デバイスタイプ)および PRINTTO プロシジャ	“UNIX 環境で PRINTTO プロシジャを使用する” (98 ページ)
	ウィンドウ環境	DMPRINT コマンド	“UNIX 環境で Print ダイアログボックスを使用する” (94 ページ)
		Print ダイアログボックス	“UNIX 環境で Print ダイアログボックスを使用する” (94 ページ)
		FILENAME ステートメントおよび PRTFILE、PRINT、および FILE コマンド	“ウィンドウのコンテンツの印刷” (97 ページ)
外部ファイル	すべてのモード	PRINTTO プロシジャおよび FILENAME ステートメント	“UNIX 環境で PRINTTO プロシジャを使用する” (98 ページ)
	ウィンドウ環境	**Print** ダイアログボックス	“UNIX 環境で Print ダイアログボックスを使用する” (94 ページ)
		FILENAME ステートメントおよび PRTFILE、PRINT、および FILE コマンド	“ウィンドウのコンテンツの印刷” (97 ページ)
	バッチ	LOG および PRINT システムオプション	“SAS システムオプションを使用し、出力先を指定する” (100 ページ)
UNIX コマンド(パイプ)	すべてのモード	FILENAME ステートメントおよび PRINTTO プロシジャ	“UNIX 環境で PRINTTO プロシジャを使用する” (98 ページ)
	ウィンドウ環境	FILENAME ステートメントおよび PRTFILE および PRINT コマンド	“ウィンドウのコンテンツの印刷” (97 ページ)
その通常の場所および外部ファイルへ	すべてのモード	ALTLOG および ALTPRINT システムオプション	“SAS システムオプションを使用し、出力先を指定する” (100 ページ)

SAS ログまたはプロシジャ出力の出力先	処理モード	方法	参照
	ウィンドウ環境	FILE コマンド	“FILE コマンドの使用 ” (98 ページ)
		Print ダイアログボックス	“UNIX 環境で Print ダイアログボックスを使用する” (94 ページ)
ターミナル	バッチ	FILENAME ステートメントおよび PRINTTO プロシジャ	“ターミナルに出力する ” (100 ページ)

SYSLOGD に SAS ログ機能メッセージを出力する

SAS ログ機能を使用すると、ログイベントメッセージの分類と収集が可能になり、またさまざまな出力デバイスにそれらを書き込むことができます。このログ機能は、問題の診断、解決、パフォーマンスと容量管理、および監査・規制コンプライアンスをサポートしています。次の機能が提供されます。

- ログイベントは、大まかなレベルで、またはきめ細かいレベルでログを設定できる階層的命名方式によって分類されます。
- ログイベントは、ファイル、オペレーティングシステムの機能、データベース、およびクライアントアプリケーションなど、複数の出力先に送信できます。出力先毎に、次のように指定できます。
  - 報告するログイベントのカテゴリとレベル
  - 含めるデータのタイプ、データの順序、およびデータの形式などのメッセージレイアウト
  - 診断レベルやメッセージ内容などの基準に基づいたフィルター
- 診断レベルのログは、各プロセスを始動や停止なしに動的に調整できます。
- パフォーマンス関連のログイベントを作成し、Application Response Measurement (ARM) 4.0 サーバーで処理できます。

このログ機能は、ほとんどの SAS Server プロセスで利用されます。SAS プログラム内のログ機能も使用できます。

UNIX 動作環境では、ログ機能メッセージは、SYSLOGD に書き込めます。

UNIX 動作環境でのログ機能の使用についての詳細は、*SAS ログ機能: 構成とプログラミングリファレンス*を参照してください。

UNIX 環境で Print ダイアログボックスを使用する

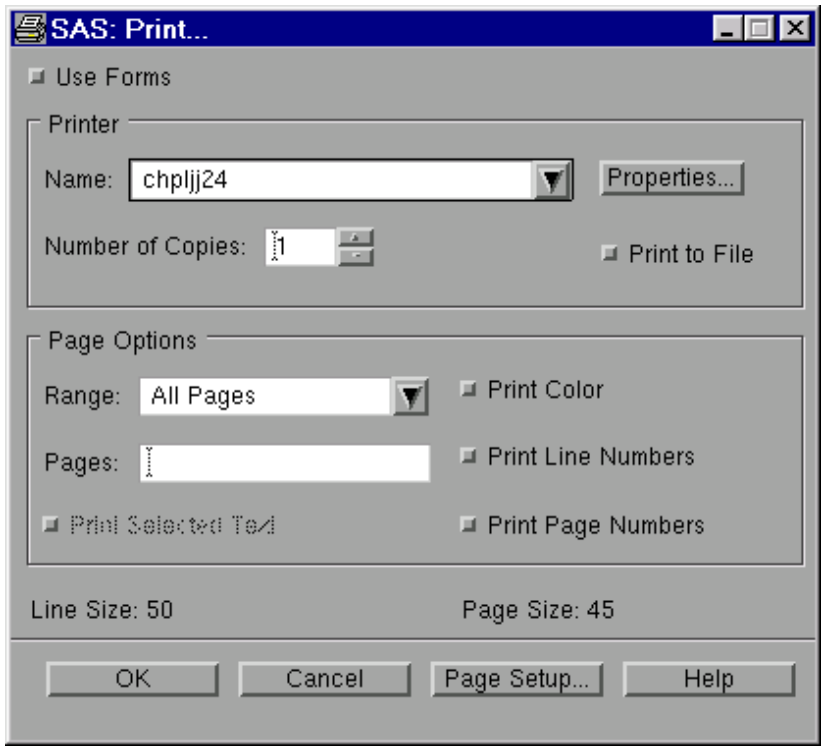
テキストウィンドウからの印刷

テキストウィンドウから Print ダイアログボックスを開く

ウィンドウのコンテンツの一部またはすべてを印刷するには、次の手順を実行してください。

1. アクティブウィンドウにするには、ウィンドウをクリックします。選択されたテキスト行だけをマークおよび印刷したい場合は、そのテキストをマークしてから、**Print** ダイアログボックスを開いてください。
2. DMPRINT コマンドを発行するか、また **File** ⇨ **Print** を選択して、**Print** ダイアログボックスを開きます。

画面 4.1　*Print ダイアログボックス*



デフォルトの印刷モード

UNIX 環境では、デフォルトの印刷モードはユニバーサル印刷です。ユニバーサル印刷の使用法についての詳細は、**Print** ダイアログボックス上の **Help** をクリックします。

印刷フォームの指定

印刷にフォームを使用するには、**Use forms** を選択します。SAS は、スプールコマンドおよびシステムプリンタ名を入力するように要求します。**OK** をクリックすると、指定したコマンドとプリンタ名、およびデフォルトフォームからの追加情報を使用して、アクティブウィンドウのコンテンツを SAS が印刷します。フォーム印刷についての詳細は、“Forms Printing” (*SAS Language Reference: Concepts* 15 章)を参照してください。

プリントサーバーエラーのトラブルシューティング

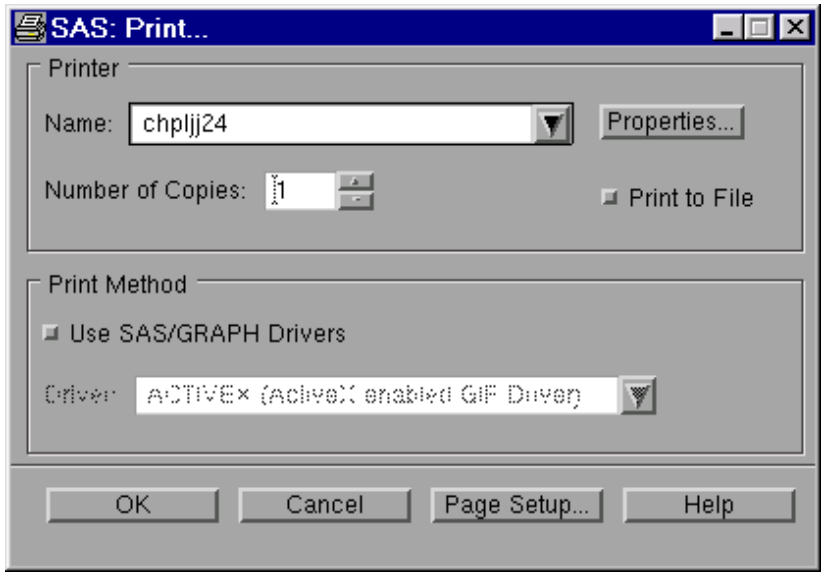
OK をクリックした後、SAS が時計アイコンを長時間表示し、出力がネットワークプリンタに送出されている場合、プリンタサーバーがダウンしている可能性があります。その場合、その SAS セッションを起動したシェルに、サーバーがダウンしていることを示すメッセージが最終的に表示されます。

GRAPH ウィンドウから印刷する

GRAPH ウィンドウから Print ダイアログボックスを開く

ユニバーサル印刷では、**Print** ダイアログボックスを使用し **SAS/GRAPH** ウィンドウのコンテンツを印刷できます。ウィンドウをクリックしてそれをアクティブにしてから DMPRINT コマンドを発行するか、**File** ⇨ **Print** を選択して **Print** ダイアログボックスを開きます。

画面 4.2 *グラフ用の Print ダイアログボックス*



注: ほとんどの場合、**Print** ダイアログボックスで設定されたフォントは、GRAPH ウィンドウから印刷する場合は、影響しません。しかし、SAS/GRAPH ドライバによってはユニバーサル印刷を使用するものもあり、このダイアログボックスに設定されたフォントにより影響を受ける場合があります。GOPTIONS ステートメントに正しいオプションが指定されていることを確認します。

SAS/GRAPH ドライバの指定

SAS/GRAPH ドライバを使用して出力を印刷するには、**Use SAS/GRAPH Drivers** を選択してください。利用可能なドライバを表示するには、**Driver** フィールドの横の下矢印を選択してください。ご使用のプリンタ割り当て先が GDEVICE プロシジャまたは GOPTIONS ステートメントを使用してデバイス内に設定されているか確認してください。GRAPH ウィンドウからの印刷についての完全な詳細は、*SAS/GRAPH: Reference* および SAS/GRAPH のオンラインヘルプを参照してください。

プリントサーバーエラーのトラブルシューティング

OK をクリックした後、SAS が時計アイコンを長時間表示し、出力がネットワークプリンタに送出されている場合、プリンタサーバーがダウンしている可能性があります。その

場合、その SAS セッションを起動したシェルに、サーバーがダウンしていることを示すメッセージが最終的に表示されます。

UNIX 環境で印刷コマンドを使用する

PRTFILE、PRINT、FILE コマンドの相違点

SAS ウィンドウ環境では、アクティブウィンドウのコンテンツを出力デバイスに送信するのに、PRTFILE、PRINT、および FILE コマンドを使用できます。

次のテーブルには、これらの各コマンドの結果が表示されています。

表 4.3 *出力先の指定コマンド*

コマンド	実行される操作
PRTFILE	出力のファイル名またはファイル参照名を指定します。
FILE	アクティブウィンドウのコンテンツを指定したファイル名またはファイル参照名に送信します。
PRINT	アクティブウィンドウのコンテンツを次のいずれかに送信します。 • ウィンドウのコマンド行から発行されたときは、ご使用のデフォルトプリンタへ。 • PRTFILE コマンドで指定された場所へ。

出力を UNIX コマンドに送信する

出力を UNIX コマンドへ送る場合は、FILENAME ステートメントを使用できます。The FILENAME ステートメントを使用すると、プリンタ、プロッタ、または UNIX コマンドに送る外部ファイルまたはファイル参照名を参照するファイル参照名を作成できます。詳細については、“FILENAME ステートメント: UNIX” (331 ページ)を参照してください。

印刷ファイルの指定

PRINT コマンドを発行する場合、SAS は、印刷ファイルを指定しない限り、出力をデフォルトのプリンタに送信します。PRTFILE コマンドを入力することで印刷ファイルを指定できます(たとえば、**PRTFILE** ***file-spec*** **CLEAR** | **APPEND** | **REPLACE**)。*file-spec* 引数には、ファイル参照名またはファイル名を使用できます。

フォーム印刷を使用しているときに **File** ⇨ **Print** を選択すると、ファイルオプションの選択を可能にするウィンドウが表示されます。**Print to File** を選択すると、**Save As** ウィンドウが表示されます。ファイルを保存する場所を入力します。このオプションは、ユニバーサル印刷が無効になっている場合のみ使用できます。

ウィンドウのコンテンツの印刷

ファイル参照名付きの PRTFILE と PRINT の使用

ウィンドウのコンテンツを印刷するには、PRTFILE コマンドに続けて PRINT コマンドを使用してください。PRTFILE により印刷先が設定され、PRINT により、その印刷先へウィンドウのコンテンツが送信されます。PRTFILE コマンドで宛先を指定しない場合、PRINT がウィンドウのコンテンツをデフォルトプリンタに自動的に送信します。

出力を直接プリンタに送るステップ

出力を直接プリンタへ送る場合は、最初に FILENAME ステートメントを送信して、PRINTER または PIPE デバイスにファイル参照名を割り当てます。たとえば、**Output** ウィンドウのコンテンツを印刷するには、次の手順を実行してください。

表 4.4 出力ウィンドウのコンテンツの印刷

ステップ	アクション	例

1	FILENAME ステートメントまたは FILENAME 関数を送信して、ファイル参照名をシステムプリンタ(PRINTER デバイスタイプ)または UNIX コマンド(PIPE デバイスタイプ)に関連付けます。プリンタ名または UNIX コマンドを一重引用符または二重引用符のいずれかで囲んでください。	`filename myrpt printer 'bldga2';`
		または
		`filename ascout pipe 'lp -dmyljet';`
		詳細については、“PRINTER と PIPE を使用した FILENAME ステートメントの例 ” (98 ページ)を参照してください。

2	PRTFILE コマンドを “印刷ファイルの指定” (96 ページ) に示すとおりに発行します。FILENAME ステートメントまたは FILENAME 関数からファイル参照名を指定します。	`prtfile myrpt`
3	コンテンツを印刷したいウィンドウのコマンド行から PRINT コマンドを発行します。システムプリンタに出力する場合またはフォームベースの印刷を使用する場合には、複数のウィンドウのコンテンツを印刷できます。	
4	出力先ファイルがすでに存在することを警告するダイアログボックスに **A** を入力します。**A** の値は、SAS に対して、ウィンドウのコンテンツを出力先ファイルへ追加するよう指示します。	
5	このファイル参照名をクリア(割り当て解除)するには、FILENAME ステートメントまたは FILENAME 関数を送信します。	`filename myrpt clear;`

印刷ファイルの設定をクリアするには、PRTFILE CLEAR コマンドを発行します。

PRINTER と PIPE を使用した FILENAME ステートメントの例

次のステートメントは、MyRpt を、BldgA2 という名前のシステムプリンタに関連付け、各 2 部のプリントアウトを指定します。

```
filename myrpt printer 'bldga2 -n2';
```

(指定可能な他のオプションについての詳細は、印刷コマンドのドキュメントを参照してください。)

次のステートメントを使用すると、**lp** コマンドを myljet という名前のプリンタ上で使用して出力を印刷できます。

```
filename ascout pipe 'lp -dmyljet';
```

次のステートメントでは、出力を **lp** コマンドへ送り、このコマンドにより生成されるエラーメッセージをご使用のホームディレクトリ内の LpError ファイルにリダイレクトします。

```
filename myrpt pipe 'lp 2>$HOME/lperror';
```

注: 標準エラーのリダイレクトは、Bourne シェルと Korn シェル内でのみ許可されます。

同じ印刷コマンドおよび出力先を頻繁に使用する場合は、適切な FILENAME ステートメントを autoexec ファイルに追加できます。詳細については、“システムオプションを使用し、SAS セッションをカスタマイズする” (18 ページ)を参照してください。

FILE コマンドの使用

多くの種類のウィンドウのコンテンツを外部ファイルにコピーするには、FILE コマンドを使用してください。コンテンツをコピーするウィンドウのコマンド行から FILE コマンドを発行します. たとえば、**Log** ウィンドウのコンテンツを**/u/myid/log/app1** にコピーするには、**Log** ウィンドウのコマンド行上で次のコマンドを発行してください。

```
file '/u/myid/log/app1'
```

ファイルが存在しない場合は、SAS により作成されます。ファイルがすでに存在している場合、ダイアログボックスは、それを置き換えるか、または既存のデータにデータを追加するかをたずねます。

ファイル参照名を外部ファイルにすでに関連付けている場合は、ファイル名ではなく、そのファイル参照名を使用できます。

```
file myref
```

FILE コマンドを使用して出力を保存する場合は、キャリッジコントロール情報が保存されません(つまり、出力から改ページが削除されます)。代わりに、FILE オプションを指定した PRINT コマンドを使用してください。

PRINT FILE=*fileref* | '*pathname*'

UNIX 環境で PRINTTO プロシジャを使用する

PRINTTO プロシジャに関する重要事項

PROC PRINTTO を使用して出力するときは、出力デバイスを閉じないと、PROC PRINTTO から、その出力やログが解放されず、指定した出力先に送信されません。出力デバイスを閉じるために、パラメータなしで PROC PRINTTO を発行します。

```
proc printto;
run;
```

パラメータを付けずに PROC PRINTTO を発行すると出力デバイスが閉じられ、出力が生成され、そしてログおよびプロシジャ出力がそのデフォルトの出力先にまた送られるようになります。デフォルト出力先のリストについては、表 4.1 (91 ページ)を参照してください。

詳細については、“PRINTTO プロシジャ: UNIX” (315 ページ) および 43 章: “PRINTTO プロシジャ” (*Base SAS Procedures Guide*)を参照してください。

LOG=オプションと PRINT=オプションの使用

PRINTTO プロシジャに LOG=および PRINT=オプションを使用すると、SAS ログや SAS プロシジャの出力を外部ファイルやファイル参照名に任意のモードから送れます。PROC PRINTTO ステートメント内で外部ファイルまたはファイル参照名を指定してください。次の例では、プロシジャ出力が **/u/myid/output/prog1** に送られます。

```
proc printto print='/u/myid/output/prog1' new;
run;
```

NEW オプションを使用すると、ファイル内の既存情報をクリアできます。PROC PRINTTO ステートメントから NEW オプションを省略すると、SAS ログまたはプロシジャが既存ファイルに追加されます。

SAS プログラム内で同じ出力先を複数回指定する場合は、FILENAME ステートメントを使ってファイル参照名をそのファイルに割り当てられます。(詳細および例については、“FILENAME ステートメントを使用し、ファイル参照名を外部ファイルまたはデバイスに割り当てる” (73 ページ)を参照してください。)

ユニバーサルプリンタに出力する

UPRINTER デバイスタイプを使用して、ユニバーサルプリンタへ直接出力できます。

```
filename myoutput uprinter;
proc printto print=myoutput;
run;
```

この場合、出力は、デフォルトユニバーサルプリンタに送られます。この出力は、PostScript または PCL 形式となります。

プリンタに出力する

PRINTER デバイスタイプを使用して、システムプリンタへ直接出力できます。

```
filename myoutput printer;
proc printto print=myoutput;
run;
```

この場合、出力は、デフォルトシステムプリンタに送られるか、SYSPRINT システムオプションが指定されている場合は、そのオプションで指定したプリンタに送られます。この方法では、ASCII 形式で出力が生成されます。

出力を UNIX コマンドにパイプする

また、PIPE デバイスタイプを使用して、出力を UNIX コマンドへ送信できます。印刷コマンドを指定する場合は、その印刷コマンドで生成されるエラーメッセージの送信先も指定してください。UNIX コマンドは、一重引用符または二重引用符で囲んでください。次の例では、ファイル参照名 MyOutput を印刷コマンド **lp** に関連付けています。これにより、myljet という名前のプリンタに出力が送られます。

```
filename myoutput pipe 'lp -dmyljet';
proc printto print=myoutput;
run;
```

LOG=オプションを使用することで、SAS ログを同一のプリンタへ送ることができます。

```
filename mylog pipe 'lp -dmyljet';
proc printto log=mylog;
run;
```

ログとプロシジャ出力は、他の PROC PRINTTO ステートメントが出力先を指定しなおすまで、指定した外部ファイルに送られ続けます。

ターミナルに出力する

バッチモードでは、ファイル参照名をターミナルに関連付けてから PROC PRINTTO により出力をそのファイル参照名に送信することにより、出力をターミナルに振り向けることができます。FILENAME ステートメントの中で、TERMINAL デバイスタイプ、およびそのターミナルに関連付けられた特殊ファイルを指定してください。たとえば、次のステートメントでは、SAS ログを、**/dev/tty3** 特殊ファイルに関連付けられたターミナルに送信します。

```
filename term terminal '/dev/tty3';
proc printto log=term;
run;
```

SAS システムオプションを使用し、出力先を指定する

LOG、PRINT、ALTLOG、ALTPRINT システムオプションを使用して出力先を変更する

SAS ログとプロシジャの出力先を変更するには、SAS システムオプションを使用してください。使用するオプションは、実行するタスクによって異なります。

- SAS ログまたはプロシジャの出力を、デフォルトの出力先でなく、外部ファイルに送るには、LOG および PRINT システムオプションを使用します。
- ログまたは出力をそのデフォルトの出力先に加えて外部ファイルに送るには、ALTLOG および ALTPRINT システムオプションを使用します。この方法は、SAS を実行するすべてのモードで使用できます。

LOG および PRINT は、通常、バッチモードおよび対話型ラインモードで 使用されます。これらのシステムオプションは、ウィンドウ環境では使用できません。ウィンドウ環境内で実行する場合は、ALTLOG および ALTPRINT システムオプションを使用してください。

これらのオプションを指定する場所は、次のとおりです。

- SAS コマンド
- 構成ファイル
- SASV9_OPTIONS 環境変数

たとえば、これらの変数は、SAS コマンド内では次のように指定できます。

```
sas -log '/u/myid/log' -print '/u/myid/prt'
sas -altlog '/u/myid/log' -altprint '/u/myid/prt'
```

詳細については、“SAS システムオプションの指定法” (18 ページ)を参照してください。

UNIX 環境で PIPE デバイスタイプを使用し、大容量ファイルを印刷する

lp コマンドでファイルを印刷する場合は、そのファイルから**/usr/spool** ディレクトリへのシンボリックリンクが作成されます。出力を **lp** コマンドにパイプする場合、出力は**/usr/spool** ディレクトリ下にコピーされます。

PIPE デバイスタイプを使用して大きいファイルを印刷するのに問題が発生する場合は、次のいずれかの方法で回避できます。

- その印刷ファイルをディスクファイルに保存してから、それを **lp** コマンドにより印刷します。**Output** または **Log** ウィンドウから PRINT コマンドを発行します。

  ```
  print file='bigfile'
  ```

  SAS セッションを終了してそのファイルを印刷するか、SAS X コマンドを使用して SAS セッション内からそのファイルを印刷します。

  ```
  x 'lp -dmylsrjt bigfile'
  ```

- 大きいファイルを扱うことのできる PIPE デバイスタイプを使用してファイル参照名を作成します。たとえば、次のファイル参照名は印刷ファイルをディスクに保存し、保存されたファイルを印刷してから、そのファイルを削除します。

  ```
  filename myfile pipe 'cat >bigfile;lp -dmylsrlt bigfile;rm bigfile;';
  ```

UNIX 環境でデフォルトの出力の印刷先を変更する

ファイル印刷時には、SAS が次の場所をチェックして出力の送信先を判断します。これらの場所は、優先順に表示されます。

1. ユニバーサル印刷で指定された出力先、または現在使用中のフォームプリンタデバイス。詳細については、*SAS 言語リファレンス: 解説編*のユニバーサル印刷またはフォーム印刷を参照してください。
2. SYSPRINT システムオプションに指定された値。デフォルトの印刷先を設定するには、SYSPRINT オプションを使用してください。印刷コマンドとともに使用する出力先オプションを指定するには、SYSPRINT システムオプションを使用します。たとえば、印刷コマンドが **lp** の場合は、次の OPTIONS ステートメントを入力することにより、デフォルト印刷先を myljet という名前のプリンタに設定できます。

   ```
   options sysprint='-dmyljet';
   ```

3. $LPDEST 環境変数の値。詳細については、“UNIX 環境で環境変数を定義する” (439 ページ) を参照してください。

SAS は、最初に検索した出力先を使用します。3 つのすべての場所内に出力先を指定すると、SAS は、ユニバーサル印刷により指定された出力先を使用します。

UNIX 環境でデフォルトの印刷コマンドを変更する

UNIX では、`lp` がデフォルト印刷コマンドとして使用されます。異なる印刷コマンドを指定する場合は、PRINTCMD システムオプションを使用できます。たとえば、SAS の起動時に次のものを入力することで、デフォルト印刷コマンドを `lpr` に変更できます。

```
sas -printcmd "lpr"
```

また、SAS 構成ファイル内でデフォルトの印刷コマンドをカスタマイズすることもできます。この方法を使用すると、SAS を起動するたびにデフォルトの印刷コマンドを変更する必要がなくなります。詳細については、“PRINTCMD システムオプション: UNIX” (408 ページ)を参照してください。

UNIX 環境で出力のコンテンツと表示画面を制御する

出力の内容と表示の制御について

SAS ログおよびプロシジャ出力の属性の中には、送信先に依存するものもあります。たとえば、ログと出力がディスプレイに送信される場合、デフォルトのラインとページサイズは、ディスプレイから導かれます。これらのファイルの一方または両方がシステムプリンタに送られるかファイルに書き込まれる場合には、デフォルトのラインサイズおよびページサイズは、使用するプリンタおよびページ設定に依存します。現在の設定のラインサイズおよびページサイズは、**Print** ダイアログボックス内で確認できます。

SAS ログおよびプロシジャ出力の属性の中には、実行中のモードに依存するものもあります。たとえば、対話型ラインモードで実行中の場合、SAS ソースステートメントは、SAS ログにエコーされません。SAS ウィンドウ環境を使用している場合は、すべてのソースステートメントが、サブミット時にログに書き込まれます。バッチモードでは、ログおよびプロシジャ出力は、標準のシステムプリンタに適合するよう形式が設定されます。

システムオプションの指定についての詳細は、“システムオプションを使用し、SAS セッションをカスタマイズする” (18 ページ)を参照してください。

SAS ログオプション

ログのコンテンツを管理するには、次のオプションを使用してください。オプションの指定についての詳細は、“UNIX 版に固有の SAS システムオプション” (360 ページ)を参照してください。

FULLSTIMER
NOFULLSTIMER
: 各 PROC または DATA ステップに使用されるリソース(実行された I/O、ページフォルト回数、経過時間、および CPU 時間など)がログに書き込まれるかどうかを制御します。NOFULLSTIMER がデフォルトです。

LINESIZE=*width*
: 使用されるライン長を制御します。*Width* は、64 - 256 までの任意の値を使用できます。

NEWS
NONEWS
メッセージが SAS ログに書き込まれるかどうかを制御します。NEWS がデフォルトです。

NOTES
NONOTES
ログ上の NOTES の印刷を制御します。NOTES は、すべての実行モード用のデフォルト設定です。SAS プログラムのデバッグが完了していない限り、NOTES を指定します。

PAGESIZE=*n*
各ページ上に印刷されるライン数を制御します。*N* は、15 - 32767 までの任意の数値を使用できます。

SOURCE
NOSOURCE
SAS ソースステートメントがログに書き込まれるかどうかを制御します
NOSOURCE が、対話型ラインモードではデフォルトの設定値です。そうでない場合は、SOURCE がデフォルトです。

SOURCE2
NOSOURCE2
%INCLUDE ステートメントと一緒に含まれる SAS ステートメントがログに書き込まれるかどうかを制御します。NOSOURCE2 は、すべての実行モード用のデフォルト設定です。

STIMER
NOSTIMER
ユーザー CPU 時間および経過時間がログに書き込まれるかどうかを制御します
STIMER がデフォルトです。

プロシジャの出力オプション

LISTING 出力先へのプロシジャ出力のコンテンツを制御するには、これらのシステムオプションを使用します。

CENTER
NOCENTER
印刷される結果は、プロシジャ出力ページ上で中央揃えにするか、または左揃えにするかを制御します。CENTER がデフォルトです。

DATE
NODATE
各プロシジャ出力ページの上部に日付が書き込まれるかどうかを制御します。
DATE がデフォルトです。

LINESIZE=*width*
使用されるライン長を制御します。*Width* は、64 - 256 までの任意の値を使用できます。

NUMBER
NONUMBER
各プロシジャ出力ページに出力ページ番号が書き込まれるかどうかを制御します。NUMBER がデフォルトです。

PAGENO=*n*
印刷ファイル内の現在のページ番号をリセットします。SAS セッション開始時のデフォルトページ番号は 1 です。セッション中、PAGENO オプションが OPTIONS ス

テートメント内に指定されていない限り、各ページには、SAS セッション全体を通じて連番が付けられます。

PAGESIZE=*n*
各ページ上に印刷されるライン数を制御します。*N* は、15 - 32,767 までの任意の番号を使用できます。

5 章

実行可能な共有ライブラリへの SAS からのアクセス

SAS の共有ライブラリの概要

共有ライブラリについて

UNIX での共有ライブラリはさまざまなプログラム言語で書かれた実行プログラムを含むライブラリです。UNIX では、これらのプログラムの名前は.so または.sl 拡張子で終了します。しかし、この命名規則に制限されてはいません。

共有ライブラリは複数のアプリケーションにて必要とされる可能性がある有用なルーチンを保存するメカニズムです。外部共有ライブラリに存在するルーチンをアプリケーションが必要とする場合、その共有ライブラリをロードし、ルーチンを呼び出し、終了後に共有ライブラリをアンロードします。

SAS 内からの共有ライブラリの呼び出し

SAS は SAS 内から外部ルーチンを呼び出せることのできるルーチンや関数を提供します。DATA ステップ、IML プロシジャおよび SCL コードなどから共有ライブラリルーチンにアクセスすることが可能です。SAS CALL ルーチンと関数(MODULE、MODULEN および MODULEC を含む)の MODULE ファミリ、そして SAS/IML CALL ルーチンと関数(MODULEIC、MODULEIN および MODULEI を含む)も同様に使用して、共有ライブラリに存在するルーチンを呼び出せます。このドキュメントは一般に MODULE 関数としての CALL ルーチンと関数の MODULE ファミリを参照します。

詳細については、“CALL MODULE Routine” (*SAS Functions and CALL Routines: Reference*), “MODULEC Function” (*SAS Functions and CALL Routines: Reference*)、および“MODULEN Function” (*SAS Functions and CALL Routines: Reference*)を参照してください。MODULEIC、MODULEIN、および MODULEI に関する詳細については、*SAS/IML 12.3 User's Guide* を参照してください。

外部共有ライブラリへのアクセスのステップ

次のステップを使用して外部共有ライブラリルーチンにアクセスします。

1. アクセスしたい共有ライブラリルーチンを記述するテキストファイルを作成します。そこには予想する引数および返す値(もしあれば)を含みます。この属性ファイルは、“SASCBTBL 属性テーブル” (106 ページ)に説明されている特殊な形式である必要があります。
2. FILENAME ステートメントを使用して SASCBTBL のファイル参照名を作成した属性ファイルに割り当てます。
3. DATA ステップまたは SCL コード内で、CALL MODULE ルーチンを使用するか、または MODULEN 関数や MODULEC 関数を使用して共有ライブラリルーチンを呼び出します。使用する特定の CALL ルーチンまたは関数は、期待される戻り値の種類(なし、数値、または文字)に依存します。(PROC IML ステップ内にて MODULEI、MODULEIN または MODULEIC を使用することが可能です)。MODULE 関数は “CALL MODULE ルーチン: UNIX” (268 ページ)にて説明されています。

注意:

経験のあるプログラマのみが共有ライブラリの外部ルーチンにアクセスする必要があります。 共有ライブラリの関数にアクセスすることで、外部関数にプロセス設定を転送することになります。正しく実行されなかったり、外部関数の信頼性がない場合、データを損失したり信頼できない結果を得たりまたは深刻なエラーを受け取る可能性があります。

SASCBTBL 属性テーブル

SASCBTBL 属性テーブルについて

MODULE 関数は、SAS が全く認識しない外部ルーチンを呼び出すため、必要に応じて、MODULE 関数がそれを検証して変換できるように、ルーチンの引数に関する情報を提供する必要があります。たとえば、引数として整数を必要とするルーチンを呼び出すものとします。SAS はすべての数字引数に浮動小数点値を使用しているため、外部ルーチンを呼び出す前に、浮動小数点値を整数に変換する必要があります。

MODULE 関数は、SASCBTBL のファイル参照名にて参照される属性テーブルでの属性情報を探します。

SASCBTBL 属性テーブルについて

属性テーブルは MODULE 関数で呼び出すルーチンの記述を含む連続テキストファイルです。テーブルは、コールされたルーチンへ送るパラメータリストを作成するときに MODULE 関数が与えられた引数をどのように処理するかを定義します。

MODULE 関数は SASCBTBL のファイル参照名によって参照されたファイルを開くことで表を見つけます。このファイル参照名を定義しない場合、MODULE 関数は引数を変更せずに単に要求された共有ライブラリルーチンをコールします。

注意:

属性テーブルを定義せずに MODULE 関数を使用すると、SAS がクラッシュしたり、予想外の結果が生じたり、重大なエラーを引き起こす原因になることがあります。 呼び出すすべての外部関数に属性テーブルを使用する必要があります。

属性テーブルの構文

属性テーブル

属性テーブルは次を含む必要があります。

- コールしようとする各共有ライブラリのルーチンのための ROUTINE ステートメントの記述
- コールしようとするルーチンに関連する各ステートメントのための ARG ステートメントの記述

属性テーブルファイルの任意の場所で、行の最初の空白ではない文字として、またはステートメント(セミコロンに続く)末尾の後としてアスタリスク(*)を使用したコマンドを作成することができます。ステートメントをセミコロンで終える必要があります。

ROUTINE ステートメント

ここに ROUTINE ステートメントの構文を示します。

ROUTINE *name* MINARG=*minarg* MAXARG=*maxarg*

<CALLSEQ=BYVALUE|BYADDR>

<TRANSPOSE=YES|NO> <MODULE=*shared-library-name*>

<RETURNS=DBLPTR | CHAR<*n*> | DOUBLE | LONG | PTR | SHORT | [U]INT32 | [U]INT64 | ULONG | USHORT>

次に ROUTINE ステートメント属性の記述を示します。

ROUTINE *name*

ROUTINE ステートメントを開始します。コールしようとする各共有ライブラリ関数のための ROUTINE ステートメントを必要とします。*name* の値は MODULE 関数の *module* 引数の一部として指定したルーチン名または序数に一致する必要があります。ここで *module* は共有ライブラリ(MODULE 属性にて指定されていない場合)の名前およびルーチン名または序数です。たとえば、MODULE 関数コールにて **`libc,getcwd`** を指定するには、ROUTINE *name* は **`getcwd`** である必要があります。

name 引数は大文字と小文字を区別し、ROUTINE ステートメントに必要です。

MINARG=*minarg*
共有ライブラリルーチンにて予想される引数の最小の数を指定します。ほとんどの場合はこの値は MAXARG;と同じですが、引数の数を変えることができるルーチンもあります。この属性は必須です。

MAXARG=*maxarg*
共有ライブラリルーチンにて予想される引数の最大の数を指定します。この属性は必須です。

CALLSEQ=BYVALUE | BYADDR
共有ライブラリルーチンで使用されるコール順序方法を意味します。値でのコールである BYVALUE とアドレスでのコールである BYADDR を指定します。デフォルト値は BYADDR です。

Fortran と COBOL はアドレスでのコール言語です。C は通常は値でのコールですが、特定のルーチンはアドレスでのコールが使用されています。

MODULE 関数はすべての引数が同じコール方法を使うことを必要とはしません。ARG ステートメントにて BYVALUE および BYADDR オプションを使用して例外を指定することが可能です。

TRANSPOSE=YES | NO
共有ライブラリルーチンをコールする前に、SAS が 1 行以上および 1 列以上の両方を持つ行列を入れ替えるように指定します。この属性は PROC IML 内で、MODULEI、MODULEIC および MODULEIN にてコールするルーチンにのみ適用されます。

行列を保存するのに行優先順序を使用しない言語にて書かれたルーチンをコールするときに、TRANSPOSE=YES は必要となります。(たとえば、Fortran は行優先順序を使用しています)。

たとえば、3 行 2 列の行列を考慮します。

```
 columns
1 2 3
------------
rows 1 | 10 11 12
2 | 13 14 15
```

PROC IML はこの行列を 10、11、12、13、14、15 の順でメモリに格納します。ところが、Fortran ルーチンはこの行列を 10、13、11、14、12、15 と期待します。

デフォルト値は NO です。

MODULE=*shared-library-name*
ルーチンが存在する実行可能なモジュール(共有ライブラリ)に名前を付けます。共有ライブラリの名前がルーチンと同じ名前の場合は、この属性を指定する必要はありません。MODULE 属性をこの ROUTINE ステートメントで指定する場合、MODULE ルーチンの *module* 引数にモジュール名を含める必要はありません(コールする共有ライブラリルーチン名が属性テーブルで固有の場合)。MODULE ルーチンは “CALL MODULE ルーチン: UNIX” (268 ページ)にて説明されています。

同じ MODULE 名を使用する複数の ROUTINE ステートメントを持つことが可能です。異なる共有ライブラリに存在する重複するルーチン名を持つことが可能です。

MODULE 属性で提供される共有ライブラリ属性をロードしようとするとき、MODULE 関数はオペレーティングシステムごとのライブラリパス環境変数で定義されたディレクトリを検索します。次の表は SAS がサポートする各 UNIX オペレーティングシステムの環境変数を示します。

表 5.1 共有ライブラリ環境変数名

オペレーティング環境	環境変数名
Solaris	$LD_LIBRARY_PATH
AIX/R	$LIBPATH
HP-UX	$LD_LIBRARY_PATH または $SHLIB_PATH
Linux	$LD_LIBRARY_PATH

注: これらの環境変数の詳細については、操作環境の man ページを参照してください。

PATH システムオプションを使用して MODULE= オプションにて指定された共有ライブラリを含むディレクトリを指定することが可能です。PATH システムオプションを使用すると、共有ライブラリをロードするときにユーザーのシステム環境変数は上書きされます。詳細については、"PATH システムオプション: UNIX" (406 ページ)を参照してください。

RETURNS=DBLPTR | CHAR<*n*> | DOUBLE | LONG | PTR | SHORT | [U]INT32 | [U]INT64 | ULONG | USHORT
共有ライブラリルーチンが応答する値のタイプを指定します。この値は、ユーザーが MODULEC(文字を戻す)または MODULEN(数値を戻す)のいずれを使用するかによって、適切に変換されます。次は考えられる応答値のタイプです。

DBLPTR
倍精度浮上小数点数字へのポインタ(浮上小数点レジスタの代替) 倍精度浮上小数点値の処理方法を決定するには共有ライブラリルーチンのドキュメントを参照してください。

CHAR<*n*>
最大 *n* バイト長の文字へのポインタ 文字列はヌルで終了が期待されており、必要に応じて、空白が埋め込まれるか、または切り捨てされます。*n* を指定しない場合、MODULE 関数は最大の長さの SAS 文字変数を使用します。

DOUBLE
倍精度浮上小数点数字

LONG
長整数

PTR
返された文字

SHORT
短整数

[U]INT32
32 ビット符号なし整数

[U]INT64
64 ビット符号なし整数

ULONG
符号なし長整数

USHORT
符号なし短整数

RETURNS 属性を指定しない場合、MODULE および MODULEI CALL ルーチンのみにてルーチンを呼び出す必要があります。RETURNS 属性を省略して、MODULEN および MODULEIN 関数または MODULEC および MODULEIC 関数を使用してルーチンを呼び出した場合、予想しない値を得ることになります。

ARG ステートメント

ROUTINE ステートメントは MAXARG=オプションにて指定した数と同じだけの ARG ステートメントを末尾に付ける必要があります。ARG ステートメントは MODULE 関数にて指定する引数の順番通りに表示する必要があります。

ここに各 ARG ステートメントの構文を示します。

ARG *argnum* NUM|CHAR <INPUT|OUTPUT|UPDATE> <NOTREQD|REQUIRED> <BYADDR|BYVALUE> <FDSTART> <FORMAT=*format*>;

次に ARG ステートメント属性の記述を示します。

ARG *argnum*
引数番号を定義します。これは必須の属性です。最初のルーチン引数(ARG 1)から始めて引数を昇順で定義します。

NUM | CHAR
引数を数字または文字として定義します。この属性は必須です。

ここで NUM を指定し文字引数のルーチンを渡すと、引数は標準数字入力形式を使用して変換されます。ここで CHAR を指定し数字引数のルーチンを渡すと、引数は BEST12.入力形式を使用して変換されます。

INPUT | OUTPUT | UPDATE
引数がルーチンへの入力、出力引数またはその両方を示します。INPUT を指定した場合、引数は変換され共有ライブラリルーチンに渡されます。OUTPUT を指定した場合、引数は変換されませんが、共有ライブラリルーチンからの出力値にて更新されます。UPDATE を指定した場合、引数は変換され、共有ライブラリルーチンに渡され、ルーチンからの出力値にて更新されます。

変数引数のみで(つまり定数や式は使用できません)、OUTPUT および UPDATE を指定することができます。

NOTREQD | REQUIRED
引数が必要かどうかを示します。NOTREQD を指定した場合、MODULE 関数は引数を省略できます。他の引数が省略された引数に続く場合、プレースホルダーとして追加のカンマを含むことで省略された引数を指定してください。たとえば、ルーチン XYZ への 2 つ目の引数を省略するには、次のように指定します。

```
call module('XYZ',1,,3);
```

注意:

NOTREQD を使用する際は注意してください。MODULE へのコールに引数が提供されていない場合、共有ライブラリルーチンはその引数にアクセスしないでください。ルーチンがそれにアクセスを試みた場合、予想しない結果または深刻なエラーを受け取る可能性があります。

REQUIRED 属性は必要な属性を示し、省略することはできません。デフォルト値は REQUIRED です。

BYADDR | BYVALUE
引数が参照または値のどちらかで渡されるかどうかを示します。

CALLSEQ=BYVALUE が ROUTINE ステートメントで指定されない限り、BYADDR がデフォルト値となります。この場合、BYVALUE がデフォルトとなりま

す。アドレスで渡される引数を持つ値でのルーチンを使用する場合は BYADDR を指定します。

FDSTART
単一引数として渡されるポインタを持つ構造に組み込まれる複数の値で引数が始まることを示します。MODULE 関数が別の FDSTART 引数を検出するまで、すべての後続の引数は構造の一部として取り扱われることに注意します。

FORMAT=*format*
共有ライブラリルーチンへの引数を提示する出力形式を名づけます。SAS、PROC FORMAT スタイル出力形式、または SAS/TOOLKIT 出力形式にてサポートされるあらゆる出力形式は有効です。引数のための UPDATE または OUTPUT 属性を指定した場合、この出力形式は該当する有効な入力形式を持つ必要があることに留意します。

FORMAT=属性は必要ありませんが、出力形式指定が属性テーブルの ARG ステートメントの主な目的であるため、この属性の指定をお勧めします。

注意:
間違った出力形式の使用は無効な結果を生成したり、SAS をクラッシュしたり、深刻なエラーを発生させたりします。

属性テーブルの重要性

MODULE 関数は属性テーブルの情報の精度に大きく影響されます。この情報が正確でない場合、予期しない結果を生じます(システムクラッシュを含む)。

整数とポインタの 2 つの引数を予期するルーチン **xyz** の例を考えます。整数は何が実行されるかを示すコードです。たとえば、実行 1 は 2 つ目の引数であるポイントにより指定された場所に書かれる 20 バイト文字の文字列を意味します。

MODULE 関数を使用して **xyz** をコールしたとします。ただし属性テーブルにて受入れ文字引数は 10 文字長のみだと指定します。

```
routine xyz minarg=2 maxarg=2;
arg 1 input num byvalue format=ib4.;
arg 2 output char format=$char10.;
```

MODULE への 2 つ目の引数にて与えられる値に関わらず、MODULE はポインタを 10 バイトの場所へ渡して **xyz** ルーチンに送ります。**xyz** がその場所に 20 バイトを書き込む場合、MODULE にて与えられた文字列に続く 10 バイトのメモリは上書きされ、予想しない結果を生じます。

```
data _null_;
length x $20;
call module('xyz',1,x);
run;
```

どの 10 バイトが上書きされたかによって、コールは動作する可能性もあります。しかし、上書きはデータの損失やシステムのクラッシュを引き起こす可能性があります。

また、PEEKLONG および PEEKCLONG 関数は与えたポインタの有効性に影響されることに留意します。ポインタが有効でない場合、深刻なエラーが生じる可能性があります。たとえば、このコードはエラーを生じます。

```
data _null_;
length c $10;
/* trying to copy from address 0!!!*/
c = peekclong(0,10);
run;
```

共有ライブラリを使用する場合に特に注意すべき点

32 ビット版と 64 ビット版の注意点

共有ライブラリと SAS との互換性

SAS 9 以降、SAS は SAS64 ビット対応のすべてのサポートされた UNIX 環境で実行される 64 ビットアプリケーションです。共有ライブラリの外部ルーチンをコールしたとき、共有ライブラリは SAS と互換性がある必要があります。

たとえば、SAS 9 を Solaris で実行していて、共用ライブラリ libc.so のルーチンをコールする必要がある場合、ライブラリをロードするにはこの共用ライブラリの互換バージョンが 64 ビットの SAS 9 であることが必要です。64 ビットアプリケーションは 32 ビットライブラリをロードできません。

ベンダ提供のライブラリが 32 ビットまたは 64 ビットかを特定するには、FILE コマンドが使用できます。次の出力は、32 ビットライブラリおよび 64 ビットライブラリの Solaris で FILE コマンドを使用した結果を示しています。

```
$ file libc-2.12.so
libc-2.12.so: ELF 64-bit LSB shared object, x86-64, version 1 (GNU/Linux),
dynamically linked (uses shared libs), for GNU/Linux 2.6.18,not stripped

$ file ./libc.so
./libc.so: ELF 64-bit MSB dynamic lib SPARCV9 Version 1, dynamically linked,
not stripped
```

ベンダ提供のライブラリにリンクする SAS モジュールをロード困難な場合、LD_LIBRARY_PATH 環境変数が正しく設定されていることを確認します。環境変数は、32 ビットディレクトリではなく、64 ビットディレクトリを指す必要があります。

共用ライブラリによる割り当て

ARG ステートメントの FORMAT 属性での各ルーチン引数のための SAS 出力形式および入力形式を特定する場合、共有ライブラリがパラメータの受け取りと応答のために割り当てるメモリの量を考慮する必要があります。外部共有ライブラリのルーチンでの入力および応答パラメータのために用意される量を決定するには、sizeof( ) C 関数を使用します。

次の表には、64 ビットシステムの C データタイプに対する一般的なメモリ割り当てをリストしてあります。

表 5.2 C データタイプ向けのメモリ割り当て

タイプ	64 ビットシステムサイズ(バイト)	64 ビットシステムサイズ(ビット)
文字	1	8
短整数	2	16
int	4	32
長整数	8	64

タイプ	64 ビットシステムサイズ(バイト)	64 ビットシステムサイズ(ビット)
長長整数	8	64
浮動	4	32
倍精度	8	64
ポインタ	8	64

データタイプのために使用する SAS 出力形式の詳細については、“MODULE 引数の使用による出力および入力形式の指定” (118 ページ)を参照してください。

共有ライブラリを使用する場合の名前付けにて注意点

名前付けの制約

SAS は次の名前付け規則を満たす外部共有ライブラリをロードします。

- 名前は 8 文字以下です。
- 名前はピリオドを含みません。

外部共有ライブラリの名前が 8 文字より多い場合やピリオドを含む場合、共有ライブラリの宛先を指すシンボリックリンクを作成することが可能です。リンク作成後、SASCBTBL 属性テーブルの MODULE ステートメントへシンボリックリンク名を追加することができます。SAS プログラムを実行する準備ができている場合、PATH システムオプションを使用してシンボリックリンクを含むディレクトリを指定します。

シンボリックリンクの作成例

/usr/lib/hpux64 ディレクトリにインストールされている Hewlett-Packard 共有ライブラリ libc.sl は名前にピリオドを含みます。SAS がこの共有ライブラリをロードする前に、8 文字以下でピリオドなしの命名規則に適合するシンボリックリンクを作成する必要があります。次の例で示すシンボリックリンクは libc.sl の宛先場所を指します。

```
$ ln -s /usr/lib/hpux64/libc.so /tmp/libclnk
```

シンボリックリンクを作成した後は、次に示すようなコードで、SASCBTBL 属性テーブルで MODULE=オプションを更新できます。

```
routine name minarg=2 maxarg=2 returns=short module=libclnk;
arg 1 char output byaddr fdstart format=$cstr9.;
arg 2 char output format=$cstr9.;
```

SAS 起動中に共有ライブラリをロードするには、次のコマンドを入力します。

```
/usr/local/sasv94/sas -path /tmp module.sas
```

PEEKLONG 関数を使用して文字列引数にアクセスする

SAS 言語はデータタイプとしてのポインタを提供しないため、SAS PEEKLONG 関数を使用してこれらのアドレス値に保存されたデータにアクセスすることが可能です。

たとえば、次のプログラムでは、ポインタアドレスの提供方法、連続する整数 1、2、3 を含む静的テーブルのアドレスにそのポインタを設定する方法を示します。また、64 ビ

ットオペレーティングシステムの **useptr** 共有ライブラリにある **useptr** ルーチンもコールします。

```
static struct MYTABLE {
int value1;
int value2;
int value3;
} mytable = {1,2,3};

useptr(toset)
char **toset;
{
*toset = (char *)&mytable
}
```

次は SASCBTBL 属性テーブル入力です。

```
routine useptr minarg=1 maxarg=1;
arg 1 char update format=$char20.;
```

次は SAS コードです。

```
data _null_;
length ptrval $20 thedata $12;
call module('*i','useptr',ptrval);
thedata=peekclong(ptrval,12);

/* Converts hexadecimal data to character data */
put thedata=$hex24.;

/* Converts hexadecimal positive binary values to fixed or floating point value */
ptrval=hex40.;
run;
```

SAS は次の出力をログに書きます。

アウトプット 5.1 *PEEKCLONG 関数での文字文字列アクセスのログ出力*

```
thedata=000000010000000200000003 ptrval=800003FFFF0C
```

この例では、ポインタの数字変数そしてバイト長さ、という2つの引数が PEEKCLONG 関数に与えられます。PEEKCLONG はポインタの場所での文字を含む指定された長さの文字文字列を応答します。

PEEKLONG 関数の詳細については、“PEEKLONG 関数: UNIX” (286 ページ)を参照してください。

共有ライブラリに 効率的にアクセスする

MODULE 関数は SASCBTBL のファイル参照名にて参照される属性テーブルをステップごとに読みます(DATA ステップ、PROC IML ステップまたは SCL ステップ)。テーブルを解析し、今後のステップ中の使用のために属性情報を保存します。MODULE 関数を使用する場合、SAS は一致するルーチンとモジュール名のために保存された属性情報を検索します。ステップの中で共有ライブラリに最初にアクセスするとき、SAS は共有ライブラリをロードし、要求されたルーチンのアドレスを特定します。呼び出す各共有ライブラリはステップ中はロードされたままで、以降のコールにて再びロー

ドされることはありません。すべてのモジュールとルーチンはステップの最後にアンロードされます。

次の例では、属性テーブルは次の基本形式を持ちます。

```
* routines XYZ and BBB in FIRST.Shared Library;
routine XYZ minarg=1 maxarg=1 module=FIRST;
arg 1 num input;
routine BBB minarg=1 maxarg=1 module=FIRST;
arg 1 num input;
* routines ABC and DDD in SECOND.Shared Library;
routine ABC minarg=1 maxarg=1 module=SECOND;
arg 1 num input;
routine DDD minarg=1 maxarg=1 module=SECOND;
arg 1 num input;
```

DATA ステップコードは次のように見えます。

```
filename sascbtbl 'myattr.tbl';
data _null_;
do i=1 to 50;
/* FIRST.Shared Library is loaded only once */
value = modulen('XYZ',i);
/* SECOND.Shared Library is loaded only once */
value2 = modulen('ABC',value);
put i= value= value2=;
end;
run;
```

この例では、MODULEN が DATA ステップコンパイル中に属性テーブルを解析します。最初の反復ループ(i=1)にて、FIRST.Shared Library がロードされ、MODULEN がコールしたときに XYZ ルーチンがアクセスされます。次に、SECOND.Shared Library がロードされ、ABC ルーチンがアクセスされます。後続の反復ループでは (i=2 から始める)、FIRST.Shared Library および SECOND.Shared Library はロードされたままで、MODULEN 関数は単に XYZ および ABC ルーチンにアクセスするのみです。SAS は DATA ステップの終わりに両方の共有ライブラリをアンロードします。

属性テーブルは任意のステップにてアクセスできないルーチンの任意の数の記述を含むことができることに留意します。属性テーブルの存在はさらなる負担を生じません (属性の記述を保持するための数バイトの内部メモリを除く)。上の例では、BBB および DDD は属性テーブルにありますが、DATA ステップからはアクセスされません。

構造引数として SAS 変数のグループ化

構造に引数を渡す

外部ルーチンをコールする場合の共通の条件はポインタを構造に渡すことです。構造の一部がそのルーチンへの入力として使用される場合があります。別の部分がルーチンにより置き換えられたり書き込まれたりすることもあります。SAS がその言語での構造を持たない場合でも、特定の引数のセットを単一構造に組み込みたいと MODULE 関数に指定することが可能です。ARG ステートメントの FDSTART オプションの使用にてこの組込みを指定して、属性テーブルにて構造を開始する引数を明示します。SAS はその引数および単一の連続ブロックに追従するすべての引数を集め(別の FDSTART オプションを検出するまで)、共有ライブラリルーチンへの引数としてポインタをブロックに渡します。

例: システム情報の構造引数としてグループ化

この例は HP-UX オペレーティング環境での **/usr/lib/hpux64/libc.so** 共有ライブラリの一部である **uname** ルーチンを使用します。このルーチンはコンピュータシステムの次の情報を応答します。

- SAS を実行しているノード名。
- オペレーティングシステムのバージョン。
- オペレーティングシステムのベンダ。
- コンピュータ識別番号。
- コンピュータのモデルタイプ。
- ハードウェア分類の固有の識別番号。この値はシリアル番号でもよいです。

次はこのルーチンの C プロトタイプです。

```
int uname(struct utsname *name);
```

C では、次のメンバにより **utsname** の構造が定義されます。

```
#define UTSLEN 9
#define SNLEN 15

char sysname[UTSLEN];
char nodename[UTSLEN];
char release[UTSLEN];
char version[UTSLEN];
char machine[UTSLEN];
char idnumber[SNLEN];
```

上の各構造メンバはヌルで終了された文字列です。

MODULE 関数を使用してこのルーチンをコールするには、次の引数表入力を使用します。

```
* attribute table entry;
routine uname minarg=6 maxarg=6 returns=short module=libc;
arg 1 char output byaddr fdstart format=$cstr9.;
arg 2 char output format=$cstr9.;
arg 3 char output format=$cstr9.;
arg 4 char output format=$cstr9.;
arg 5 char output format=$cstr9.;
arg 6 char output format=$cstr15.;
```

次の例は SAS ソースコードが DATA ステップ内から **uname** ルーチンをコールしています。

```
x 'if [ ! -L ./libc ]; then ln -s /usr/lib/hpux64/libc.so ./libc ; fi' ;
x 'setenv LD_LIBRARY_PATH .:/usr/lib:/lib:/usr/lib/hpux64'

data _null_;
length sysname $9 nodename $9 release $9 version $9 machine $9 idnumber $15.
retain sysname nodename release version machine idnumber " ";
rc=modulen('uname', sysname, nodename, release, version, machine, idnumber)
put rc = ;
put sysname = ;
put nodename = ;
put release = ;
put version = ;
put machine = ;
```

```
put idnumber = ;
run;
```

SAS は次の出力をログに書きます。

ログ 5.1 *構造として SAS 変数のグループ化*

```
rc=0
sysname=HP-UX
nodename=garage
release=B.11.31
version=u
machine=ia64
idnumber=103901537
```

MODULE 関数の引数として定数と式を使用する

任意のタイプの式を MODULE 関数の引数として渡すことが可能です。属性テーブルは引数が入力、出力または更新のどれであるかを指定します。

入力引数を定数および数式として指定することができます。しかし、出力および更新の引数は変更および応答することが可能である必要があるため、それらの変数のみを渡すことが可能です。更新可能な値が予想される場所にて定数や式を定義した場合、SAS はエラーを示す警告メッセージを発行します。プロセスは継続しますが、MODULE 関数は更新を実行できません(更新に必要な引数の値が失われたことを意味します)。

これらの例を考えます。これは属性テーブルです。

```
* attribute table entry for ABC;
routine abc minarg=2 maxarg=2;
arg 1 input format=ib4.;
arg 2 output format=ib4.;
```

これは MODULE コールを持つ DATA ステップです。

```
data _null_;
x=5;
/* passing a variable as the */
/* second argument - OK */
call module('abc',1,x);

/* passing a constant as the */
/* second argument - INVALID */
call module('abc',1,2);

/* passing an expression as the */
/* second argument - INVALID */
call module('abc',1,x+1);
run;
```

上の例では、**abc** ルーチンが 2 つ目の引数のために戻す値によって **x** が更新されるため、MODULE への最初のコールは正しく行われます。2 回目の MODULE へのコールは定数が渡されるため間違っています。MODULE は、定数を渡したと示す警告を発行し、暫定場所をかわりに渡します。3 回目の MODULE へのコールは間違っています。なぜなら数式が渡され、そのため DATA ステップからの暫定場所が使用され、応答値が失われるからです。

MODULE 引数の使用による出力および入力形式の指定

ARG ステートメントの FORMAT 属性の使用

ARG ステートメントの FORMAT 属性を指定することで、各共有ライブラリルーチン引数のための SAS 出力形式および入力形式を指定します。出力形式は、共有ライブラリルーチンに数字と文字の値がどのように渡されるべきか、そしてルーチンの終了後にどのように読み戻されるべきかを指定します。

通常、使用する出力形式は任意のプログラム言語のさまざまなタイプに一致します。次のセクションはさまざまなプログラム言語での異なる変数タイプに一致する適切な出力形式を説明します。

C 言語形式

表 5.3 C 言語形式

C タイプ	64 ビットシステムでの SAS 出力形式または入力形式
倍精度	RB8.
浮動	FLOAT4.
符号付 int	IB4.
符号付短整数	IB2.
符号付長整数	IB8.
文字 *	IB8.
符号なし int	PIB4.
符号なし短整数	PIB2.
符号なし長整数	PIB8.
char[*w*]	$CHAR*w*. または$CSTR*w* (“$CSTRw. 出力形式” (120 ページ) を参照してください)。

注: 文字文字列へのポインタとしての用途以外の文字データの渡されに関しての詳細については、“$BYVALw. 出力形式” (121 ページ)を参照してください。

FORTRAN 言語形式

表 5.4 FORTRAN 言語形式

Fortran タイプ	SAS 出力形式または入力形式
整数*2	IB2.

Fortran タイプ	SAS 出力形式または入力形式
整数*4	IB4.
実数*4	RB4.
実数*8	RB8.
文字*_w_	$CHAR_w_.

記述子によって渡されることのない場合に限り、MODULE 関数は Fortran 文字引数をサポートします。

PL/I 言語形式

表 5.5 PL/I 言語形式

PL/I タイプ	SAS 出力形式または入力形式
FIXED BIN(15)	IB2.
FIXED BIN(31)	IB4.
FLOAT BIN(21)	RB4.
FLOAT BIN(31)	RB8.
CHARACTER(_w_)	$CHAR_w_.

完全にするために PL/I 記述がここに追加されています。これらの記述は PL/I ルーチンを呼び出すという保障はしません。

COBOL 言語形式

表 5.6 COBOL 言語形式

COBOL 形式	SAS 出力形式または入力形式	説明
PIC S_xxxx_ BINARY	IB_w_.	バイナリ整数
COMP-2	RB8.	倍精度浮上小数点
COMP-1	RB4.	単一精度小数点
PIC_xxxx_ または S_xxxx_	F_w_.	印刷可能な数字
PIC _yyyy_	$CHAR_w_.	文字

次の COBOL の指定は、SAS 提供の出力形式とは厳密に一致しない可能性があります。ゾーンおよびパック 10 進数は、Intel アーキテクチャに基くシステムに対応するように適切に定義されていないからです。

表 5.7 COBOL 指定と SAS 出力および入力形式

COBOL 形式	SAS 出力形式または入力形式	説明
PIC S*xxxx* DISPLAY	ZD*w*.	ゾーン 10 進数
PIC S*xxxx* PACKED-DECIMAL	PD*w*.	パック 10 進数

次の COBOL 指定は同等の自然実数を持っておらず、該当する 370F*xxx* 出力形式および入力形式のみに関連して利用できます。これにより IBM メインフレーム型の式が UNIX 環境にて読み書きされます。

表 5.8 出力および入力形式の S370Fxxx グループとともに使用される COBOL 指定

COBOL 形式	SAS 出力形式または入力形式	説明
PIC *xxxx* DISPLAY	S370FZDU*w*.	ゾーン 10 進数
PIC S*xxxx* DISPLAY SIGN LEADING	S370FZDL*w*.	ゾーン 10 進数頭符号
PIC S*xxxx* DISPLAY SIGN LEADING SEPARATE	S370FZDS*w*.	ゾーン 10 進数頭符号分割
PIC S*xxxx* DISPLAY SIGN TRAILING SEPARATE	S370FZDT*w*.	ゾーン 10 進数末尾符号分割
PIC *xxxx* BINARY	S370FIBU*w*.	符号なしバイナリ整数
PIC *xxxx* PACKED-DECIMAL	S370FPDU*w*.	符号なしパック 0 進数

$CSTRw. 出力形式

文字引数をヌルで終了された文字列として渡す場合、$CSTR*w*.出力形式を使用します。この出力形式は文字引数の最後の空白ではない文字を探し、末尾の空白でない文字の後にヌルターミネータを持つ文字列のコピーを渡します。たとえば、次の属性テーブルのエントリを考えます。

```
* attribute table entry;
routine abc minarg=1 maxarg=1;
arg 1 input char format=$cstr10.;
```

このエントリにて、次の DATA ステップを使用することが可能です。

```
data _null_;
rc = module('abc','my string');
run;
```

$CSTR 形式は、**abc** ルーチンに渡す前に、ヌルターミネータを文字列 **my string** に追加します。ヌルターミネータを文字列に追加し、**abc** ルーチンへ文字列を渡すことは、次の属性入力と同等です。

```
* attribute table entry;
routine abc minarg=1 maxarg=1;
arg 1 input char format=$char10.;
```

入力は次の DATA ステップを持つことも可能です。

```
data _null_;
rc = module('abc','my string'||'00'x);
run;
```

最初の例は理解するのは簡単で、変数や式引数を使用するときにより簡単に使用できます。

$CSTR 入力形式はヌルターミネータ文字列を特定の長さの空白で埋められた文字列に変換されます。共有ライブラリルーチンが文字引数を更新するものと仮定されている場合、引数属性にて$CSTR 入力形式を使用します。

$BYVALw. 出力形式

MODULE 関数を使用して単一文字を値にて渡す場合、引数は自動的に整数に変換されます。MODULE コールで文字表現を使用する場合、$BYVAL*w.*と呼ばれる特殊な出力形式および入力形式を使用する必要があります。$BYVAL*w.* 出力形式および入力形式では単一文字を予測して、数値を生成しますが、そのサイズは *w.*に依存します。$BYVAL2.は短整数を生成し、$BYVAL4.は長整数を生成し、そして $BYVAL8.は倍精度を生成します。C 言語を使用したこの例を考慮します。

```
long xyz(a,b)
long a; double b;
{
static char c = 'Y';
if (a == 'X')
return(1);
else if (b == c)
return(2);
else return(3);
}
```

この例では、**xyz** ルーチンが長整数と倍精度という 2 つの引数を予測します。長整数が **x** である場合、この長整数の実際の値は 10 進法の 88 になります。この結果は、ASCII **X** が 16 進数の 58 として保存され、この値が 0x00000058 として表現される長整数(つまり 10 進数では 88)に変換されるために発生します。**a** の値が **x**、つまり 88 である場合、1 が返されます。2 つ目の引数(倍精度)が **y** (89 として解釈される)である場合、2 が返されます。

文字を引数として **xyz** へ受け渡す場合、C 言語では、次のように文字を呼び出します。

```
x = xyz('X',(double)'Z');
y = xyz('Q',(double)'Y');
```

x および **Q** 値は自動的に整数(この例では長整数)に変換され、**Z** および **Y** に対応する値は倍精度に型変換されるため、文字はこのように呼び出されます。

MODULEN 関数を使用して **xyz** をコールするには、属性テーブルが文字を渡したいという事実に反映したものである必要があります。

```
routine xyz minarg=2 maxarg=2 returns=long;
arg 1 input char byvalue format=$byval4.;
```

```
arg 2 input char byvalue format=$byval8.;
```

ARG ステートメントにも BYVALUE オプションが表示されることは重要であると留意します。さもないと、MODULEN はポイントを値ではなくルーチンに渡したいものと仮定します。

これは MODULEN を呼び出し文字を渡す DATA ステップです。

```
data _null_;
x = modulen('xyz','X','Z');
put x= ' (should be 1)';
y = modulen('xyz','Q','Y');
put y= ' (should be 2)';
run;
```

MODULE ログメッセージについて

MODULE の設定文字列パラメータに **i** を指定した場合、SAS はログにさまざまな情報メッセージを印刷します。これらのメッセージを使って、間違った引数を送ったかどうかまた属性テーブルを間違ってコードしたかどうかを特定することができます。

IML プロシジャ内で MODULEIN を使用するこの例を考えます。これは MODULEIN 関数を使用して **changi** ルーチン(仮想の TRYMOD.so に保存されています)を呼び出します。この例では、MODULEIN は定数 6 および行列 x2 を渡します。ここで行列は 4x5 であり整数行列に変換されます。**changi** の属性テーブルは次のようになります。

```
routine changi module=trymod returns=long;
arg 1 input num format=ib4. byvalue;
arg 2 update num format=ib4.;
```

次の IML ステップは MODULEIN を呼び出します。

```
proc iml;
x1 = J(4,5,0);
do i=1 to 4;
do j=1 to 5;
x1[i,j] = i*10+j+3;
end;
end;
y1= x1;
x2 = x1;
y2 = y1;
rc = modulein('*i','changi',6,x2);
....
```

'*i'設定文字列は次に示す出力行をログに書き込みます。

ログ 5.2 *MODULEIN ログ*

```
---PARM LIST FOR MODULEIN ROUTINE--- CHR PARM 1 885E0AA8 2A69 (*i)
CHR PARM 2 885E0AD0 6368616E6769 (changi)
NUM PARM 3 885E0AE0 0000000000001840
NUM PARM 4 885E07F0
0000000000002C400000000000002E4000000000000030400000000000003140000000000000324000
0000000000038400000000000003940000000000003A400000000000003B400000000000003C400000
0000000004140000000000080414000000000
---ROUTINE changi LOADED AT ADDRESS 886119B8 (PARMLIST AT 886033A0)--- PARM 1
06000000 <CALL-BY-VALUE>
PARM 2 88604720
0E0000000F0000001000000011000000120000001800000019000000 1A0000001B0000001C000000
2200000023000000240000002500000026000000 2C0000002D0000002E0000002F00000030000000
---VALUES UPON RETURN FROM changi ROUTINE--- PARM 1 06000000 <CALL-BY-VALUE>
PARM 2 88604720
140000001F0000002A0000003500000040000000820000008D000000980000 00A3000000AE000000
F0000000FB000000060100001101000 01C0100005E0100006901000074010000 7F0100008A010000
---VALUES UPON RETURN FROM MODULEIN ROUTINE--- NUM PARM 3
885E0AE00000000000001840
NUM PARM 4 885E07F0
000000000000344000000000000003F4000000000000045400000000000804A400000000000005040
0000000000406040000000000A0614000000000000063400000000000606440000000000C06540
0000000000006E400000000000606F4000000000
```

出力は 4 つのセクションに分かれます。

- 最初のセクションは MODULEIN に渡された引数の記述です。

  CHR PARM *n* 部は文字パラメータ *n* が渡したことを示します。この例では、885E0AA8 が MODULEIN への最初の文字パラメータの実際のアドレスです。このアドレスでの値は 16 進数の 2A69 であり、この値('*i')の ASCII 表現は 16 進数値の後のかっこ内となります。2 つ目のパラメータは同様に印刷されます。これらの最初の 2 つの引数のみが ASCII 同等を印刷されます。なぜなら他の引数は読み込むことができないバイナリデータを含む可能性があるからです。

  残りのパラメータは値の 16 進数表現のみで表示されます(例では NUM PARM 3 および NUM PARM 4)。

  MODULEIN への 3 つ目のパラメータは数字でアドレス 885E0AE0 にあります。浮動小数点 6 の 16 進数表現を示します。4 つ目のパラメータはアドレス 885E07F0 にあり、4x5 行列のすべての値を含む場所を指定します。**`*i`** オプションは全体の引数を印刷します。ログが非常に大きくなる可能性があるため、大きな行列を使ったこのオプションの使用は注意してください。

- ログの 2 つめのセクションは、要求されたルーチンに渡されこの場合では変更される引数を示します。このセクションは引数が正しくルーチンに渡されているかどうかを決定するのに重要です。このセクションの最初の行はルーチン名とメモリでのアドレスを含みます。MODULEIN が作成したパラメータブロックのアドレスも含みます。

  ログは各引数が渡される際のステータスを含みます。たとえば、例の最初のパラメータは値でコールされるものです(ログに示すように)。2 つ目のパラメータは行列のアドレスです。ログはアドレスと、そのアドレスが示すデータを示します。

  属性テーブルが出力形式を IB4 と指定してあるため、最初のパラメータと行列のすべての値は長整数となります。

- 3 つ目のセクションでは、ログが **`changi`** からの応答後の引数の値を含みます。値にてコールされる引数は変化しませんが、他の引数(行列)は異なる値を含みます。

- ログ出力の最後のセクション MODULEIN CALL ルーチンへ戻される引数の値を含みます。

実行可能な共有ライブラリへのアクセスの例

例 1: 文字列引数の更新

この例では、**/usr/lib/64** ディレクトリにインストールされている、Solaris が提供する共用ライブラリ libc.so の **tmpnam** ルーチンを使用します。**tmpnam** ルーチンは暫定ファイル名として安全に使用できる固有のファイル名を作成します。一時ファイル名は一般的に**/var/tmp** ディレクトリに置かれます。

このルーチンの C プロトタイプを示します。

```
char * tmpnam(char *s);
```

このプロトタイプの属性テーブルは次のようになります。

```
routine tmpnam minarg=1 maxarg=1 returns=char255. module=libc;
arg 1 char output byaddr format=$cstr255;
```

次は SAS コードです。

```
x 'if [ ! -L ./libc ] ; then ln -s /usr/lib/64/libc.so.1 ./libc ; fi' ;
x 'setenv LD_LIBRARY_PATH .:/usr/lib/64/usr/lib:/lib';

data _null_;
length tempname $255 tname $255;
retain tempname tname " ";
tname = modulec ('tmpnam', tempname);
put tempname = ;
put tname = ;
run;
```

SAS ログには、次の情報が表示されます。

ログ 5.3 *文字列引数の更新*

```
tempname=/var/tmp/aaaKraydG
tname=/var/tmp/aaaKraydG
```

パス名の最大文字数のための**/usr/include/limits.h** で定義された POSIX 基準は 255 です。よってこの例では 254 を生成されたファイル名(**tempname**)として使用し、1 文字分をヌルターミネータとして使用します。$CSTR255.の入力形式は、コントロールが DATA ステップに戻ったときに、ヌルターミネータとすべての後続文字が末尾空白に入れ替えられることを確実にします。

例 2: 値による引数の受け渡し

この例では、ほとんどの UNIX ベンダにて提供される **access** ルーチンをコールします。**/usr/lib/hpux64** ディレクトリにインストールされている Hewlett-Packard 共有ライブラリ libc.sl に、この特定の **access** ルーチンがあります。

このルーチンの C プロトタイプを示します。

```
int access(char *path, int amode);
```

access ルーチンは、**amode** に含まれるビットパターンに従って、アクセスパスで参照されるファイルをチェックします。テストする権限の種類に相当する **amode** には、次の整数値を使用できます。

```
4 Read access
2 Write access
1 Execute (search) access
0 Check existence of file
```

戻り値 0 は、正常な完了および要求したアクセスが許可されることを示します。戻り値–1 は、失敗および要求したアクセスが許可されないことを示します。

amode 引数は"値で渡される”であるという理由で、この例では属性テーブルの *arg 2* に BYVALUE 指定が含まれています。両方の引数が"値で渡される”で、ROUTINE ステートメントに CALLSEQ=BYVALUE 属性を使用できた場合は、*arg 2* に BYVALUE オプションを指定する必要はなかったでしょう。

属性テーブルは次のようになります。

```
routine access minarg=2 maxarg=2 returns=short module=libc;
arg 1 char input byaddr format=$cstr200.;
arg 2 num input byvalue format=ib4.;
```

SAS ソースコードは次のようになります。

```
x 'if [ ! -L ./libc ] ; then ln -s /usr/lib/hpux64/libc.so ; fi' ;
x 'setenv LD_LIBRARY_PATH .:/usr/lib/hpux64:/usr/lib:/lib' ;

data _null_;
length path $200.;
path='/dev';

/* A non-root user is testing for write permission in the /dev directory */
rc = modulen("*ie",'access',path,2);
put rc = ;
run;
```

SAS ログ出力は次のようになります。

ログ 5.4 *要求アクセスが許可されたかどうかの結果*

```
rc=-1
```

ユーザーの$HOME ディレクトリでの書き込み許可をチェックするように SAS ソースコードを変更した場合、出力は異なります。

```
data _null_;
length homedir $200.;
homedir=sysget('HOME');

/* A user is testing for write permissions in their $HOME directory */
rc = modulen(“*ie”,'access',homedir,2);
put rc = ;
run;
```

この場合、SAS ログ出力は次のようになります。

ログ 5.5 *正常終了の結果(アクセス許可済み)*

```
rc=0
```

例 3: PEEKCLONG を使用した応答されたポインタへのアクセス

この例では、Red Hat Linux 共有ライブラリ libc-2.12.so の一部である **strcat** ルーチンを使用しています。このライブラリは通常 **/lib64** ディレクトリにインストールされています。このルーチンは 2 つの文字列を連結し、ポインタを新しく連結された文字列に戻します。

このルーチンの C プロトタイプを示します。

```
char *strcat(char, *dest, const char *src);
```

正しい SASCBTBL の属性テーブルは次のようになります。

```
routine strcat minarg=2 maxarg=2 returns=ulong module=libc;
arg 1 char input format=$cstr200.;
arg 2 char input format=$cstr200.;
```

次の例は SAS コードを示します。

```
filenamesascbtbl './sascbtbl.txt';

data _null_;
file sascbtbl;
put "routine strcat minarg=2 maxarg=2 returns=ulong module=libc;";
put "arg 1 char input format=$cstr200.;";
put "arg 2 char input format=$cstr200.;";
run;

data _null_;
length string1 string2 newstring $200;
length chptr $20;
string1='This is string one and';
string2=' this is string two.';
chptr=modulec('strcat', string1, string2);
newstring=peekclong(chptr,200);
put newstring=;
run;
```

SAS は次の出力をログに書きます。

ログ 5.6 *戻されたポインタへの PEEKCLONG を使用したアクセス結果*

```
newstring=This is string one and this is string two.
```

PEEKLONG 関数および PEEKCLONG 関数の詳細については、“PEEKLONG 関数: UNIX” (286 ページ) および “PEEKCLONG Function” (*SAS Functions and CALL Routines: Reference*)を参照してください。

例 4: 構造の使用

“構造引数として SAS 変数のグループ化” (115 ページ) は、共有ライブラリルーチンへの単一構造引数として、複数の引数を渡すためにどのように FDSTART 属性を使用するか説明しています。複数の引数を単一構造として渡すのは、外部共有ライブラリでの別のルーチンと共に使用する構造の 1 つの例です。

ほとんどの UNIX オペレーティングシステムで使用可能な **statvfs** ルーチンが、ファイルシステム情報を検索します。この例では、一般的には **/usr/lib/sparcv9** ディ

レクトリにインストールされている、Solaris の共用ライブラリ libc.so.1 にある **statvfs** ルーチンを使用します。

このルーチンの C プロトタイプを示します。

```
int statvfs(const char *path, struct statvfs *buf);
```

statvfs ルーチンはルーチンが正常に終了すると0を返し、失敗すると–1 を返します。

statvfs 構造は次のメンバで定義されます。

```
unsigned long f_bsize; /* preferred file system block size */
unsigned long f_frsize; /* fundamental file system block */
unsigned long f_blocks; /* total number of blocks on file system in units */
unsigned long f_bfree; /* total number of free blocks */
unsigned long f_bavail; /* number of free blocks available to non-superuser */
unsigned long f_files; /* total number of file nodes (inodes) */
unsigned long f_ffree; /* total number of free file nodes */
unsigned long f_favail; /* number of inodes available to non-superuser */
unsigned long f_fsid; /* file system id (dev for now) */
char f_basetype[16]; /* target fs type name, null-terminated */
unsigned long f_flag; /* bit mask of flags */
unsigned long g f_namemax; /* maximum filename length */
char f_fstr[32]; /* file system specific string */
```

SASCBTBL の属性テーブルは次のようになります。

```
routine statvfs
minarg=14
maxarg=14
returns=short
module=libc;
arg 1 char input byaddr format=$char256.;
arg 2 num output byaddr fdstart format=pib8.;
arg 3 num output format=pib8.;
arg 4 num output format=pib8.;
arg 5 num output format=pib8.;
arg 6 num output format=pib8.;
arg 7 num output format=pib8.;
arg 8 num output format=pib8.;
arg 9 num output format=pib8.;
arg 10 num output format=pib8.;
arg 11 char output format=$cstr16.;
arg 12 num output format=pib8.;
arg 13 num output format=pib8.;
arg 14 char output format=$cstr32.;
```

DATA ステップ内から **statvfs** ルーチンをコールする SAS ソースコードは次のようになります。

```
x 'if [ ! -L ./libc ]; then ln -s /usr/lib/sparcv9/libc.so.1 ./libc ; fi' ;
x 'setenv LD_LIBRARY_PATH .:/usr/lib/sparcv9:/usr/lib:/lib';

data _null_;
length f_basetype $16. f_fstr $32.;
retain f_bsize f_frsize f_blocks f_bfree f_bavail f_files f_ffree f_favail
f_fsid f_flag f_namemax 0;
retain f_basetype f_fstr ' ';
rc=modulen ('statvfs' , '/tmp', f_bsize, f_frsize, f_blocks, f_bfree, f_bavail,
```

```
f_files, f_ffree, f_favail, f_fsid, f_basetype, f_flag,
f_namemax, f_fstr);
put rc = ;
put f_bsize = ;
put f_frsize = ;
put f_blocks = ;
put f_bfree = ;
put f_bavail = ;
put f_files = ;
put f_ffree = ;
put f_favail = ;
put f_fsid = ;
put f_basetype = ;
put f_flag = ;
put f_namemax = ;
/* Determining the total bytes available in the file system and then dividing the
total number of bytes by the number of bytes in a gigabyte */
gigsfree = ((f_bavail * f_bsize)/1073741824);
put 'The total amount of space available in /tmp is 'gigsfree 4.2' Gigabytes.';
run;
```

次は SAS ログ出力です。

ログ 5.7 *構造を使用した場合のログ出力*

```
rc=0
f_bsize=8192
f_frsize=8192
f_blocks=196608
f_bfree=173020
f_bavail=173020
f_files=884732
f_ffree=877184
f_favail=877184
f_fsid=2
f_basetype=tmpfs
f_flag=4
f_namemax=255

The total amount of space available in /tmp is 1.32 Gigabytes.
```

例 5: 共有ライブラリルーチンの呼び出し

この例は、PROC IML 内の引数としてどのように行列を渡すかを示します。例は 4x5 行列を作成します。各セルは $10x+y+3$ に設定されます。ここで x は列番号、y は行番号です。たとえば、列 1 行 2 のセルは(10*1)+2+3 つまり 15 に設定されています。

例では仮想の TRYMOD 共有ライブラリからさまざまなルーチンを呼び出します。ここでは **changd** ルーチンを使用して $100x+10y$ を各要素に追加します。ここで x は C 列番号(0 から 3)で y は C 行番号(0 から 4)です。**changd** への最初の引数は余分量の合計を指定します。**changdx** ルーチンは **changd** と同様に動作します。例外は、転置行列を対象としています。**changi** ルーチンは **changd** と同様に動作します。例外は、整数行列を対象としています。**changix** ルーチンは **changdx** と同様に動作します。例外は、整数を対象としています。

注: PROC IML から共有ライブラリルーチンを呼び出す場合、最大 3 つの引数が送られます。

この例では、x1、x2、y1、および y2 のすべて 4 つの行列が、該当する MODULEIN がコールした後に、同じ値に設定されます。ここに属性テーブルエントリを示します。

```
routine changd module=trymod returns=long;
arg 1 input num format=rb8. byvalue;
arg 2 update num format=rb8.;
routine changdx module=trymod returns=long
transpose=yes;
arg 1 input num format=rb8. byvalue;
arg 2 update num format=rb8.;
routine changi module=trymod returns=long;
arg 1 input num format=ib4. byvalue;
arg 2 update num format=ib4.;
routine changix module=trymod returns=long
transpose=yes;
arg 1 input num format=ib4. byvalue;
arg 2 update num format=ib4.;
```

これは PROC IML ステップです。

```
proc iml;
x1 = J(4,5,0);
do i=1 to 4;
do j=1 to 5;
x1[i,j] = i*10+j+3;
end;
end;
y1= x1; x2 = x1; y2 = y1;
rc = modulein('changd',6,x1);
rc = modulein('changdx',6,x2);
rc = modulein('changi',6,y1);
rc = modulein('changix',6,y2);
print x1 x2 y1 y2;
run;
```

次は PRINT ステートメントの結果です。

アウトプット 5.2 *PROC IML から共有ライブラリルーチンの呼び出す*

```
X1
20 31 42 53 64
130 141 152 163 174
240 251 262 273 284
350 361 372 383 394
X2
20 31 42 53 64
130 141 152 163 174
240 251 262 273 284
350 361 372 383 394
Y1
20 31 42 53 64
130 141 152 163 174
240 251 262 273 284
350 361 372 383 394
Y2
20 31 42 53 64
130 141 152 163 174
240 251 262 273 284
350 361 372 383 394
```

6 章

SAS リモートブラウザでの出力とヘルプの表示

リモートブラウジングについて

リモートブラウジングでは、SAS ドキュメント、WBROWSE コマンドに指定された URL、および ODS 出力を、ローカルコンピュータの Web ブラウザで表示できます。過去には、すべての Web ドキュメントを SAS Server で実行している Netscape ブラウザで表示していました。このドキュメントをローカルに表示することで、ドキュメントへのアクセスを速くし、Netscape にて使用されていた SAS Server のリソースを自由にできます。

リモートブラウザサーバーと呼ばれるソフトウェアエージェントがローカルコンピュータで実行されます。SAS が HTML の内容を表示する必要があるとき、リモートブラウザに接続して内容を参照する URL を送ります。リモートブラウザサーバーは表示のために URL をブラウザに送ります。リモートブラウザサーバーがコンピュータにて実行されていない場合、SAS はリモートブラウザサーバーをダウンロードする必要があるという URL を含むダイアログボックスを表示します。

リモートブラウジングを設定するための 2 つのシステムオプションが提供されます。HELPHOST および HELPPORT です。これらのオプションは HTML の内容が表示される場所のコンピュータのホスト名とポート番号を指定します。ほとんどの場合、これらのオプションを設定する必要はありません。クライアントが X11 転送を有効にした SSH を使用して UNIX ホストに接続する場合、HELPHOST は X11 DISPLAY 環境変数で指定されたホスト名、または SSH_CLIENT 環境変数にて指定された IP アドレスをデフォルト値とします。HELPPORT はリモートブラウザサーバーの標準ポートをデフォルトとします。

ODS 出力のリモートブラウジング

SAS Output Delivery System (ODS) は SAS データのグラフィックによるレポートを作成するのに使用できます。出力が生成または **Results** ウィンドウからの要求に合わせて、SAS セッションからの出力をリモートブラウジングにて直接表示することが可能です。

リモートブラウジングは ODS 出力を多くの形式で表示することが可能です。お使いのブラウザが HTML ではない出力のための適切なプラグインを持たない場合、ブラウザは出力ではなくダイアログボックスを表示します。このダイアログボックスにより出力をコンピュータにダウンロードし、XSL ファイルを Excel などのローカルプルグラムで表示することが可能です。

ODS 出力(HTML、PDF および RTF のみ)の自動表示はデフォルトでオフとなっています。**Results** ウィンドウにて AUTONAVIGATE コマンドを発行するか、**Preferences** ダイアログボックスの **Results** タブから **View results as they are generated** を選択することで、ODS 出力の自動表示をオンにすることが可能です。

リモートブラウザサーバーのインストール

SAS セッションから直接リモートブラウザサーバーをインストールすることが可能です。SAS がリモートブラウジングに接続することができない場合、SAS はインストーラをダウンロードする必要があるという URL を含むダイアログボックスを表示します。この URL を使用して、リモートブラウザサーバーのダウンロードおよびインストールを行います。SAS を終了しないでください。リモートブラウザサーバーをインストールするために、次の手順に従ってください。

1. ダイアログボックスに表示される URL をブラウザに入力して Enter を押すか、ダイアログボックスの **Copy URL** ボタンを使用して URL をコピーし、それをブラウザに貼り付けます。
2. ダウンロードページが表示された後、コンピュータに適切なインストーラをダウンロードします。
3. インストーラの実行
   - Windows 環境では、リモートブラウザサーバーをスタートアップ項目に追加しておくと、ユーザーがログオンするときいつもそのサーバーが開始します。システムトレイに表示されたアイコンはリモートブラウザサーバーが実行中であることを示します。
   - Linux の環境では、**`rbrowser`** コマンドをウィンドウ環境のスタートアップスクリプトに手動で追加します。リモートブラウザサーバーは最初に画面最小にて開始します。

リモートブラウジングのシステムオプション

リモートブラウザサーバーがコンピュータにて実行し始めた後、HELPHOST および HELPPORT システムオプションを指定することでリモートブラウザシステムを実行することが可能です。

- HELPHOST システムオプションはリモートブラウジングシステムが表示されるホストコンピュータの名前を指定します。このオプションを指定しない場合、X 表示名で指定されたホスト名が使用されます。詳細については、“HELPHOST システムオプション: UNIX” (385 ページ)を参照してください。
- HELPPORT システムオプションはコンピュータにインストールされているリモートブラウザサーバーのためのポート番号を指定します。UNIX では、このオプションのデフォルト値を使用することができます。詳細については、“HELPPORT= System Option” (*SAS System Options: Reference*)を参照してください。

これらのオプションは構成ファイルにて SAS 起動時または SAS セッション中の OPTION ステートメントか **SAS System Options** ウィンドウにて設定することが可能です。

SAS リモートブラウザの設定

SAS 起動時に SAS リモートブラウザを設定する

次の構文は UNIX 操作環境に特有で、リモートブラウザサーバーがネットワークポート 12000 を使用している場合どのように SAS リモートブラウザを設定するかを示しています。

```
sas94 -helpport 12000
```

HELPHOST システムオプションを指定しなかったため、SAS は X ディスプレイ名にて指定されたホスト名を使用しています。

SAS セッション時に SAS リモートブラウザを設定する

この例での構文は UNIX 環境に適用します。

SAS セッション中に OPTIONS ステートメントまたは **SAS System Options** ウィンドウを使用してリモートブラウジングシステムを設定することが可能です。次の例は OPTIONS ステートメントを使用して HELPPORT システムオプションの値を変更しています。

```
options helpport=12000;
```

HELPHOST システムオプションを指定しなかったため、値は変更されていません。

リモートブラウジングとファイアウォール

一般ユーザー向け

デスクトップコンピュータと SAS をホストしているコンピュータの間でネットワークにファイアウォールがある場合、Web ブラウザは SAS セッションからの Web ページを表示できません。通常、この問題は Web ブラウザからのタイムアウトまたは接続エラーで示されます。タイムアウトや接続エラーを受け取った場合、システム管理者に連絡してください。

システム管理者

デスクトップコンピュータと SAS をホストしているコンピュータの間にファイアウォールが存在する場合に Web ページを表示可能にするには、Web ブラウザが SAS に接続できる許可をファイアウォールのルールに追加する必要があります。ファイアウォールのルールは、SAS リモートブラウジングの接続を許可するネットワークポートの範囲を指定します。リモートブラウジングのためのネットワークポートの範囲を選択して設定することができる適任のシステム管理者に連絡してください。範囲は同時使用の SAS ユーザーの数によります。同時使用の SAS ユーザーの数の約3倍をネットワークポートに十分な数として割り当てる必要があります。

ファイアウォールのルールを追加した後、ネットワークポート範囲にてネットワーク接続を監視するように SAS を設定する必要があります。通常、SAS は使用されていないあらゆるネットワークポートを使用しますが、HTTPSERVERPORTMIN および HTTPSERVERPORTMAX システムオプションは SAS が選択できるネットワークポートを制限します。これらのシステムオプションを SAS 構成ファイルに追加します。HTTPSERVERPORTMIN をネットワーク範囲の一番低いポートに設定します。HTTPSERVERPORTMAX をネットワーク範囲の一番高いポートに設定します。たとえば、システム管理者がネットワークポートの範囲を 8000 から 8200 と定義した場合、システムオプションは次のようになります。

```
httpserverportmin=8000
httpserverportmax=8200
```

これらのシステムオプションが設定されると、デスクトップコンピュータが Web ページを表示できます。十分な数のネットワークポートが存在しない場合や、システムオプションが正しく指定されていない場合、SAS ログにメッセージが表示されます。

これらのシステムオプションの詳細については、“HTTPSERVERPORTMIN= System Option” (*SAS System Options: Reference*) および “HTTPSERVERPORTMAX= System Option” (*SAS System Options: Reference*)を参照してください。

7 章

UNIX でのパフォーマンスに関する注意点

ファイルシステムの I/O スループットの測定

概要

SAS プロセスでは、ファイルシステムと I/O プロセスに対して、標準的なデータベースまたはクエリプロセスとは異なる要求が出されます。このため、SAS ソフトウェアの実行時にシステムの I/O スループット率を決定することが重要になります。SAS テクニカルサポートがスクリプト iotest.sh を作成しました。UNIX および Linux プラットフォームでこれを実行すると、I/O テストを実行できます。このスクリプトは、コマンド行のパラメータを受け入れて、同時書き込み/読み取りテストの出力を生成します。iotest.sh スクリプトは、SAS をシステムにインストールする必要がないスタンドアロンプログラムです。スクリプト実行時のインスタンスごとに、プログラムで、経過実時間と I/O 率を取り込む出力ファイルが書き込まれます。I/O 率は、テストの各反復ごとに 1 秒あたりのメガバイト数(MB/秒)で表されます。

iotest.sh スクリプトは、`dd` シェルコマンドを使用して、定義された負荷でのシステムの I/O 動作を測定します。このスクリプトを使用して、ファイルシステムをフラッドする個別 I/O テストや複数の同時 I/O テストを起動し、システムの生パフォーマンスを決定します。

iotest.sh スクリプトでは、テストファイルが作成され、テストされるファイルシステムに書き込まれます。スクリプトで、ファイルが読み直されて、ファイルシステムの書き込み/読み取りパフォーマンスが計算されます。スクリプトで、1 秒あたりに書き込まれるメガバイト数(書き込み MB/秒)と、1 秒あたりに読み取られるメガバイト数(読み取り MB/秒)が決定されます。これらの測定基準を使用して、現在のシステムスループットを決定し、さらに調整が必要かどうかを判断します。

スクリプトのダウンロード

UNIX または Linux プラットフォーム用の iotest.sh スクリプトを、SAS テクニカルサポート(**ftp://ftp.sas.com/techsup/download/unix/iotest.sh**)からダウンロードします。

スクリプトを実行して結果を保存する予定のファイルシステムにスクリプトを置きます。このファイルシステムは、テストをするファイルシステムとは別にする必要があります。

次のコマンドを実行して、スクリプトを実行可能にします(#はコマンド行プロンプトを示しており、コマンドの一部ではありません)。

```
# chmod 0555 iotest.sh
```

スクリプトの実行

iotest.sh スクリプトを実行する前に、対象ファイルシステムのブロック数とブロックサイズを知っておく必要があります。iotest.sh スクリプトの構文は次のとおりです。

```
# ./iotest.sh -i <iterations> -t <target> -b <number-of-blocks> -s <block-size>
```

iotest.sh スクリプトのパラメータの説明は次のとおりです。

- -i 書き込み/読み取りテスト反復数
- -t テストする対象ファイルシステム
- -b 対象システムでのブロック数
- -s 対象システムの KB 単位でのブロックサイズ

次のコマンド例について考えます。

```
# ./iotest.sh -i 5 -t /saswork -b 2179072 -s 64
```

このコマンドでは、テストファイルの 5 回の同時書き込み反復の後に続けて、5 回の同時読み取り反復が要求されます。対象ファイルシステムは**/saswork** ディレクトリに格納されています。ブロック数 2,179,072 にブロックサイズ 64 K を掛けます。この結果、テストファイルサイズは 133 GB になります。これは、この例で使用されるマシン上の 132GB の RAM より大きいです。5 回の反復があるため、133GB サイズのテストファイルが 5 つ生成されます。スクリプトの結果は、**/saswork** ディレクトリとは別のカレントディレクトリに書き込まれます。

出力

iotest.sh スクリプトでは、画面および iotest.sh.results.<*n*>という出力ファイルへの出力が行われます。*n* の値は、iotest.sh スクリプトで要求した反復の数です。

iotest.sh 実行時の画面への出力例は次のとおりです。

```
iotest.sh-readtest.out.1:real 1062.15
iotest.sh-readtest.out.2:real 1000.38
iotest.sh-readtest.out.3:real 1004.22
iotest.sh-readtest.out.4:real 1001.28
iotest.sh-readtest.out.5:real 1019.17
iotest.sh-writetest.out.1:real 6695.02
iotest.sh-writetest.out.2:real 6684.86
iotest.sh-writetest.out.3:real 6688.99
iotest.sh-writetest.out.4:real 6661.41
iotest.sh-writetest.out.5:real 6682.23
```

```
RESULTS
-------

INVOCATION : iotest.sh -i 5 -t /saswork -b 1547695 -s 64


TARGET DETAILS
directory : /saswork
df : /dev/mapper/vg_saswork-lv_saswork 1548168396 201812 1469324204 1% /saswork
mount point : /dev/mapper/vg_saswork-lv_saswork on /saswork type ext4 (rw)
filesize : 101429739520 bytes or 96730.93 megabytes


STATISTICS
average read time in seconds : 1017.44
average read throughput rate : 95.07 megabytes per second
aggregate read throughput rate : 475.35 megabytes per second
average write time in seconds : 6682.50
average write throughput rate : 14.47 megabytes per second
aggregate write throughput rate : 72.35 megabytes per second
```

この出力は、読み取り/書き込みテスト反復を示しています。出力には、テスト結果を生成するために発行されたコマンドと、対象ファイルシステムについての情報が表示されます。この情報には、マウントポイント、ファイルシステムの種類、およびマウントオプションが含まれます。iotest.sh スクリプトでは、バイトおよび MB 単位でテストファイルのサイズが計算されます。

次に、すべての反復にわたる平均と集計 が、読み取り-時間統計量と読み取り-出力統計量について表示されます。平均と集計は、書き込み-時間統計量と書き込み-出力統計量についても表示されます。例では、これら 5 つの同時テストの集計 読み取り出力はおよそ 475 MB/秒で、集計 書き込み出力はおよそ 72 MB/秒であることがわかります。

SAS を使用した I/O スループットテストのベストプラクティス

ホストシステムでスループットをテストするための準備をする際のベストプラクティスは次のとおりです。

- **テストファイルがホストシステムのファイルキャッシュより大きいことを確認します。** SAS では、可能であれば、ホストシステムのファイルキャッシュによって読み取り操作と書込み操作が実行されます。テストファイルがファイルキャッシュより小さい場合、順次テストでは、ディスクのファイルを読み書きするのではなく、キャッシュ内のテストファイルにアクセスします。この結果、パフォーマンス測定基準がまぎらわしくなります。

  ベストプラクティスとして、ホストシステムのファイルキャッシュより大きいファイルを作成します。

- **テストスクリプトの複数のインスタンスを実行します。** 1 回の読み取りまたは書き込みテストで、1 つの SAS ジョブが、ある 1 時点でどのように実行されるのかが示されます。システムがすでに複数ユーザーのワークロードで非常にビジーな状態で

ある場合は、1 回のテストで十分とすることもあります。ただし、ベストプラクティスとしては、複数のテスト実行の平均を取って、標準的なシステム状況下での SAS の実行内容をより正確に理解します。

- **パフォーマンステストの複数インスタンス呼び出しを実行します。**複数インスタンス呼び出しを実行すると、作業ユーザーや実行ジョブが少ないときは特に、ホストシステムでの同時 SAS ワークロード動作がよりよく示されます。
- **複数のビジーな日にわたって異なる時刻にテストを実行します。**これにより、さまざまなワークロードを経験するシステムでの I/O 特性の包括的なプロファイルが提供されます。
- **最低でも、以前に問題を経験したときと同じ時刻のテストは行います。**SAS パフォーマンスの問題を経験した場合、観測されたパフォーマンス低下のいずれかと同じ状況下でテストします。これにより、調整パラメータの変更後に改善が見られたかどうかを評価できます。

調整ガイドライン

スループット対象

良好な基本的 SAS I/O パフォーマンスのためには、SAS ログのジョブステップとともに、経過実時間の 10%以内の、結合された CPU 時間とユーザー CPU 時間が表示されている必要があります。これが当てはまらない場合は、対象ファイルシステムの I/O 設定の調整に利点がある可能性が高くなります。

標準的な小型サイズから平均サイズまでのシステムについては、良好なスループット目標は、1 プロセッサコアあたり 75 MB/秒以上の必要があります。より大型で使用頻度が高いシステムについては、良好なスループット目標は、1 プロセッサコアあたり 100 MB/秒以上です。

オペレーティングシステム調整ガイドライン

ベンチマーキングと実用的観測に基づいて、よりよいパフォーマンスを達成するには、UNIX システムについて次の推奨事項に従います。

- **Red Hat Linux システム。**大量の I/O を呼び出す場合、Linux 6.4 以上を実行する 64 ビット Linux X64 マシンで SAS 9 を実行します。その他の推奨事項については、“Linux 環境” (139 ページ)を参照してください。
- **Oracle Solaris システム。**Oracle Solaris 10 については、推奨ファイルシステムは ZFS です。

  ZFS インテントログ(ZIL)ファイルが、SAS に関連するファイルシステムとは別の物理ディスクを有するファイルシステムに存在することを確認します。それ以外の場合は、高負荷のジャーナリングアクティビティによってパフォーマンスの問題が引き起こされる可能性があります。
- **IBM AIX システム。**AIX 5L、AIX 6、および AIX 7 システムについては、優先ファイルシステムは JFS2 です。AIX 6.1 を実行する場合、ベストプラクティスは、SAS Work に関連付けられたファイルシステムを、LOG=_NULL_ パラメータを指定してマウントすることです。
- **Hewlett-Packard HP-UX システム。**ベストプラクティスとしては、Veritas VxFS ツールを使用して、SAS で使用されるファイルシステムを作成します。SAS Work に

関連付けられたファイルシステムに対して提案されるマウントオプションは次のとおりです。

```
/dev/vgWork/work /work vxfs noatime,tmplog,convosynch=delay,
mincache=tmpcache,datainlog 0 2
```

SAS Work 環境には、再起動やクラッシュ後には使用できない一時コンテンツが含まれるので、そのコンテンツの整合性を確認する必要はありません。

サイズが 1KB 未満のファイルを作成するのでなければ、ファイルシステムでデフォルトブロックサイズ値を 1 KB から 8 KB に増やすことをお勧めします。

Linux 環境

tuned ツールの使用

前のガイドラインにリストされたプロファイルおよび環境設定の多くを設定するには、**tuned** ツールを使用します。**tuned** ツールの使用については次の情報を知っておく必要があります。

- **インストール。tuned** ツールは、Red Hat Enterprise Linux 6 ではデフォルトでインストールされていません。**tuned** ツールをインストールするには、次のコマンドを実行します。

  ```
  # yum install tuned*
  ```

- **使用可能な調整プロファイル。**使用可能な調整プロファイルはいくつかあります。スループットプロファイルと enterprise-storage プロファイルが最もよく使用されます。使用可能プロファイルのリストを参照するには、次のコマンドを実行します。

  ```
  # tuned-adm list
  ```

- **アクティブな調整プロファイル。** 現在アクティブな調整プロファイルを参照するには、このコマンドを実行します。

  ```
  # tuned-adm active
  ```

- **プロファイルのアクティブ化。**プロファイルをアクティブ化するには、**tuned-adm profile** コマンドを実行し、アクティブ化するプロファイルの種類を指定します。たとえば、enterprise-storage プロファイルをアクティブ化するには、次のコマンドを実行します。

  ```
  # tuned-adm profile enterprise-storage
  ```

- **プロファイルの非アクティブ化。**調整プロファイルを指定したパフォーマンスが好ましくない場合は、デフォルトプロファイルに戻します。現在のプロファイルを非アクティブ化するには、デフォルトプロファイルに戻し、次のコマンドを実行します。

  ```
  # tuned-adm profile default
  ```

Red Hat Enterprise Linux ガイドライン

次のパラメータまたは環境設定のデフォルト値を変更することをお勧めします。

- **先行読み取りサポート。** SAS ファイルシステムで使用される論理ユニット番号 (LUN)と論理ボリュームは、先行読み取りサポートの増加に合わせて調整する必要があります。これらの値を設定するお勧めの方法は、**blockdev** コマンドを使用することです。

  ブロックデバイスに対する先行読み取りを表示するには、次のコマンドを実行します。

  ```
  # blockdev --getra <path-to-block-device>
  ```

ブロックデバイスに対する先行読み取りを設定するには、次のコマンドを実行します。

```
# blockdev --setra <N> <path-to-block-device>
```

N の推奨値は 8192 または 16384 です。*N* の値は 2 の累乗の値に設定する必要があります。

注: 先行読み取り設定は、起動と起動の間は永続しません。ベストプラクティスとしては、起動中に調整サービスが開始された後で、実行レベル **init.d** サービススクリプトを作成して、先行読み取りパラメータを無効化します。

- **I/O エレベータ。**デフォルト Red Hat Enterprise Linux I/O エレベータは、Completely Fair Queuing (CFQ)です。パフォーマンステストでは、最適値が期日エレベータであることが示されます。または、場合によっては、最適値が、SAS 順次ワークロードの noop エレベータになります。I/O エレベータを **deadline** に設定するには、**tuned** ツールを使用します。
- **I/O バリア。**エンタープライズ記憶域キャッシュコントローラ RAM がバッテリによってバックアップされた場合、I/O バリアは安全に無効化できます。enterprise-storage プロファイルに対する **tuned** ツールによって、I/O バリアを無効化したファイルシステムが再マウントされます。I/O バリアが無効化された ext4 または XFS ファイルシステムをマウントするには、各ファイルシステムに対して **nobarrier** マウントパラメータを追加します。
- **transparent huge pages。**transparent huge pages 機能によって、匿名メモリに対して 4-KB ページのかわりに 2-MB ページの割り当てが試行されます。transparent huge pages 機能を無効化した場合、大きなファイルの I/O に対してページキャッシュを使用するアプリケーションでは、パフォーマンスが向上することもあります。この機能を無効化するには、次のコマンドを実行します。

  ```
  # echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled
  ```

  注: **tuned** ツールでは、デフォルトですべてのプロファイルに対して transparent huge pages 機能を有効化できます。**init.d** サービススクリプトでの起動時やコマンド行での手動起動時など、調整プロファイルが有効な場合はいつでも、この機能が再有効化されます。
- **ページング領域。**次の提案を含めるようにページング領域を設定します。
  - 専用ディスクにページング領域を配置して、I/O 競合を除去します。複数のディスクに分散した複数のページング領域を使用します。プライマリページング領域 hd6 は、セカンダリページング領域より少し大きくします。
  - ページング領域が同時 SAS プロセスの数をサポートするのに十分であることを確認します。同時 SAS プロセスの数は、アプリケーションワークロードに応じて動的になることもあります。
- **ディスクレイアウト。**SAS 一時領域とデータ領域間のディスクの競合を最小限に抑えます。
  - SAS 一時領域ファイルシステムと SAS データファイルシステムを物理的に別のディスクに配置します。
  - 複数の記憶域サーバーコントローラを使用して、SAS 一時領域とデータ領域間の I/O トラフィックをさらに分離します。
  - SAS データファイルシステムに対して複数のマウントポイントを使用します。オペレーティングシステムおよび SAS、ユーザー、SAS 一時領域、ならびに SAS データファイルシステムを別々の物理ディスクに配置します。
  - SAS ユーザーのグループによる使用が可能な複数の SAS Work 領域の作成を考慮します。

  - 少数の大容量ディスクスピンドルではなく多数の物理ディスクスピンドルに I/O ワークロードを分散します。ディスク容量ではなくディスク数量に基づいてサイズ変更を決定します。同じディスクスピンドルセットの周囲に LUN を配置しないでください。
  - ディスクスピンドルを RDBMS と共有しないでください。
- **ファイルシステムプリファレンス。**SAS の操作時には、ファイルシステムに関する次の情報に注意してください。
  - XFS ファイルシステムは通常、SAS ファイルシステムで最高のパフォーマンスを発揮します。
  - ext4 ファイルシステムは、SAS で良好なパフォーマンスを発揮します。
  - ext3 ファイルシステムは、Red Hat Enterprise Linux 用のレガシファイルシステムです。このファイルシステムは、削除遅延のため、SAS での使用はお勧めしません。

  グリッド環境で SAS を使用する場合、共有ファイルシステムが必須となります。グリッド環境における SAS 実行の詳細については、オンサイトサポート担当者に問い合わせてください。
- **ホストバスアダプタ(HBA)。**記憶域からホストサーバーまで適切な数の HBA を使用すると、必須アプリケーション帯域幅が提供されます。
  - 低速メディアのかわりに、ファイバーチャネルテクノロジなどのハイパフォーマンス記憶域チャネルを検討してください。
  - 可能であれば、動的マルチパスを使用して、複数の HBA に I/O 負荷を分散します。
- **RAID (redundant array of independent disks)。**記憶域システム RAID ストライピングを複数の物理ディスクに実装します。
  - 冗長性のレベル、およびファイルシステムの各種類ごとに必要な合計容量に応じて、RAID 10 または RAID 5 を使用します。ファイルシステムの種類に対する使用可能容量のかわりに、冗長性のレベルおよび合計容量を使用します。
  - すべての LUN にわたって、デフォルト―連結―ではなく論理ボリュームマネージャ(LVM)ストライピングを使用します。
- **LVM ストライピング。**LVM ストライピングは、複数の記憶域配列を連動させる際にはきわめて重要です。ディスクストライプもしくはセグメントサイズを選択するか、または配列ストライプサイズを選択する場合、Linux ファイルシステムが 16-KB 境界に配置されることに注意してください。
  - 64 KB または 128 KB の LVM ストライプサイズ、および 256 KB または 512 KB のストライプサイズにすると、SAS 9.2 ワークロードでは I/O パフォーマンスに改善が見られます。8 KB のストライプサイズは、SAS ワークロードには小さすぎます。
  - SAS BUFSIZE システムオプションを、記憶域システムストライプサイズ、LVM ストライプサイズ(LVM ストライピングの使用時)、および先行読み取り増分と同期させます。
  - I/O サイズの同期によって、I/O 処理が効率化され、記憶域サブシステムへの I/O 要求数が削減されます。

SAS 中間層と SAS Web Application Server

SAS 中間層と SAS Web Application Server に対する調整値の詳細については、*SAS Web Applications: Tuning for Performance and Scalability* を参照してください。

2 部

SAS ウィンドウ環境

8 章

SAS ウィンドウ環境の操作

SAS ウィンドウ環境の定義

SAS ウィンドウ環境は、SAS 起動時に開くウィンドウを参照します。これらのウィンドウには、**Program Editor**、**Log**、**Output**、**Explorer** および **Results** が含まれます。ご使用の X ワークステーションから、または X エミュレータから SAS を起動すると、次のウィンドウが表示されます。これらのウィンドウの詳細については、オンライン SAS ヘルプとドキュメントを参照してください。

SAS ウィンドウ環境では、X ベースのグラフィカルユーザーインターフェイス(GUI)の使用がサポートされます。UNIX 環境において、SAS は、Motif スタイルに基づいた X Window System インターフェイスを提供します。

SAS ウィンドウ環境の多くの機能は、X リソースで制御されます。たとえば、ウィンドウサイズ、SAS ToolBox の外観およびキー定義などは、すべて X リソースにより制御されます。“SAS ウィンドウ環境のカスタマイズ” (170 ページ) は、リソースの指定方法などのリソースに関する一般情報を提供し、またインターフェイスのカスタマイズに使用できるすべてのリソースについて説明します。

X 環境の SAS について

X Window System の定義

X Window System は、ネットワーク化されたウィンドウシステムです。複数のコンピュータが 1 つのネットワーク上にある場合、X サーバーを実行できます。それによって、

ネットワーク内の他のコンピュータから(クライアントとして)交互に X アプリケーションを提供します。

X Window マネージャ

UNIX 環境での SAS は、Motif ベースの X Window System インターフェイスを利用できます。このインターフェイスは、システム上のウィンドウマネージャにより、ディスプレイに表示される各ウィンドウを管理します。Inter-Client Communication Conventions Manual (ICCCM)に準拠したウィンドウマネージャであれば、SAS の Motif インターフェイスに使用できます。ベンダは、X Window System 環境をもつウィンドウマネージャを 1 つ以上提供します。一般的なウィンドウマネージャは、GNOME です。 他のウィンドウマネージャとしては、KDE があります。使用するウィンドウマネージャについては、ベンダから提供されるドキュメントを参照してください。

すべてのウィンドウマネージャの実行する基本機能は同じですが、そのスタイルおよび拡張機能が異なります。SAS に接続するインタフェースの外観と機能は、ある程度、ご使用の X Window マネージャによって異なります。ほとんどのウィンドウマネージャは、ウィンドウの周りに一種のフレームを提供します。また、ウィンドウマネージャは、ウィンドウの配置、サイズ設定、重なり、外観、そしてキーボードとの連携を設定します。SAS を操作する基本は、メニューのオープン、ウィンドウの移動、ダイアログボックスへの応答、テキストのドラッグなど、すべてのウィンドウマネージャについて同じです。

SAS ウィンドウのセッション ID

X ワークステーション上で SAS を実行すると、SAS は、他の SAS セッションを含め、他の X アプリケーションとディスプレイを共有します。各種アプリケーションと SAS セッションを区別するため、SAS は、アプリケーション名(デフォルトでは **sas**)に番号を追加することにより、各セッションに SAS ウィンドウセッション ID を生成します。このセッション ID は、各 SAS ウィンドウのウィンドウタイトルバーおよびウィンドウアイコンタイトルに表示されます。これらの SAS セッションは順次割り当てられます 最初の SAS セッションには番号は割り当てられないため、セッション ID は **sas** となり、2 番目の SAS セッションにはセッション ID**sas2** が割り当てられ、その後も同様にセッション ID が割り当てられます。デフォルトのアプリケーション名は **sas** ですが、インスタンス名を変更するには、**-name** X オプションまたは**-title** X オプションを使用できます。インスタンス名の最大長は 64 文字であり、入力時に使用された大文字、小文字、またはそれらの組み合せで表示されます。

SAS セッションのワークスペースとグラビティ

X Window 上で SAS を使用する場合、ディスプレイが多くの並行アプリケーションにより共有されていることがあります。各種セッションの SAS ウィンドウと他のアプリケーションのウィンドウがディスプレイに表示されていると、ディスプレイが雑然とした状態になることがあります。この問題を軽減するため、SAS セッションの各ウィンドウは、最初に AWS(アプリケーションワークスペース)内に表示されます。AWS は、SAS ウィンドウが最初に作成される仮想ディスプレイを表す長方形の領域を定義します。SAS は、ディスプレイの左上隅に対して AWS を位置付けようとします。つまり、このワークスペースは、ディスプレイ上の一定の方向に引き寄せられます(セッショングラビティ)。いくつかのウィンドウマネージャ構成では、SAS がウィンドウに対して選択した配置が無効になる場合があります。

ウィンドウコマンドを発行するか、または新規の SAS ウィンドウを作成する SAS 手順を実行すると、最初の位置とサイズの同一規則がこれらのウィンドウに適用されることから、各ウィンドウは最初、SAS AWS 内に配置されます。現在のウィンドウ位置(またはジオメトリ)を保存するには、WSAVE コマンドを使用してください。詳細については、

“UNIX 環境でセッションワークスペース、セッショングラビティ、ウィンドウサイズをカスタマイズする” (211 ページ)を参照してください。

ウィンドウの種類

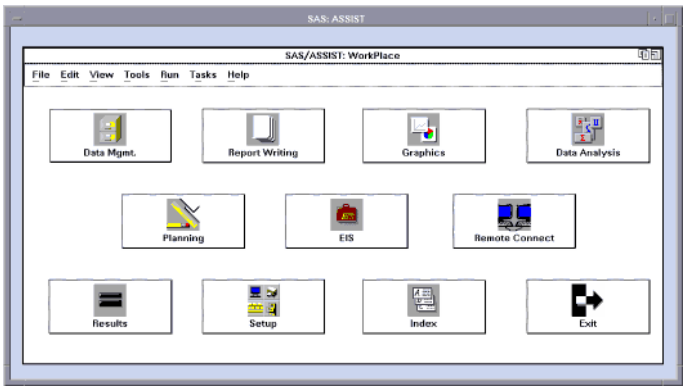
最上位ウィンドウ

SAS では、プライマリウィンドウおよびインテリアウィンドウが使用されます。SAS アプリケーションによっては、SAS により管理されるインテリアウィンドウに加えて、X Window マネージャにより管理される 1 つ以上のプライマリウィンドウによって構成されているものがあります。これらの SAS ウィンドウのプライマリウィンドウは、ほとんどの SAS アプリケーションウィンドウと同様、最初は、最上位ウィンドウとして表示されます。最上位ウィンドウは、X Window マネージャを直接操作します。これらには、他のウィンドウマネージャの装飾とともにフルタイトルバーがあります。これらは、ディスプレイ上に表示されたあとに、個別に操作できます。

内部ウィンドウ

インテリアウィンドウの動作は、プライマリウィンドウとは異なります。SAS/ASSIST ソフトウェアは、インテリアウィンドウをもつアプリケーションです。インテリアウィンドウはコンテナウィンドウの中に含まれていますが、これらはプライマリウィンドウでない場合があります。次のディスプレイでは、SAS/ASSIST ソフトウェア内のインテリアウィンドウが表示されています。

画面 8.1 インテリアウィンドウのサンプル



SAS では、インテリアウィンドウに対するある程度のウィンドウ管理が可能です。具体的には、インテリアウィンドウには、次のサイズ変更機能および移動機能があります。

- インテリアウィンドウを移動するには、そのインテリアウィンドウのタイトルバーをクリックして、ウィンドウを必要な位置までドラッグします。インテリアウィンドウの移動先がコンテナウィンドウの境界を超えると、そのコンテナウィンドウは、`SAS.awsResizePolicy` リソースの値に従って変化します。(コンテナウィンドウの中のスペースは、アプリケーションワークスペースです。これについては、“SAS セッションのワークスペースとグラビティ” (147 ページ)に説明があります。)詳細については、“X リソースの概要” (170 ページ)を参照してください。
- インテリアウィンドウを個別に最小化することはできません。コンテナウィンドウアイコンボタンをクリックすると、そのコンテナウィンドウおよびそのインテリアウィンドウが最小化されます。
- インテリアウィンドウでは、**push-to-back** ボタン(右上隅の小さい重なり合った四角)が利用できます。ただし、アクティブウィンドウを非アクティブウィンドウの背後に移動できません。

UNIX 環境の SAS Session Manager (motifxsassm)

SAS Session Manager について

X (**motifxsassm**)用の SAS Session Manager は、SAS ウィンドウ環境を使用するときに SAS により実行される X クライアントです。SAS Session Manager は、SAS 起動時に自動的に最小化されます。SAS Session Manager 用の **SAS: Session Management** ダイアログボックスは、次のディスプレイの示される通りに表示されます。

画面 8.2 SAS: Session Management ダイアログボックス

SAS: Session Management ダイアログボックスには、次の情報が列挙されます。

- 管理対象の SAS セッション
- SAS セッション起動元のホストコンピュータ
- SAS セッションの UNIX プロセス識別子

SAS Session Manager の機能

SAS: Session Management ダイアログボックスの中の各ボタンにより、次のタスクが実行できます。

最小化
SAS セッションのすべてのウィンドウのマッピングと最小化に使用されます。この機能は、標準ライブラリ呼び出しにより実行され、またほとんどの X Window マネージャが使用できます。

リストア
SAS セッションの中で開いている、SAS Session Manager により制御されるすべてのウィンドウをリストアします。この機能は、標準ライブラリ呼び出しにより実行され、またほとんどの X Window マネージャが使用できます。

割り込み
UNIX シグナルを SAS へ送信します。SAS がそのシグナルを受信すると、**Tasking Manager** ダイアログボックスを表示します。(“SAS セッションに割り込む” (150 ページ)を参照してください。)

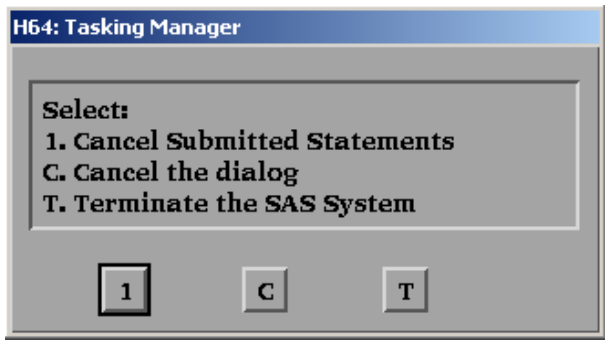
終了
SAS セッションを終了するかどうかの確認を要求するダイアログボックスを表示します。

ヘルプ
SAS: Session Management ダイアログボックスのヘルプを提供します。

SAS セッションに割り込む

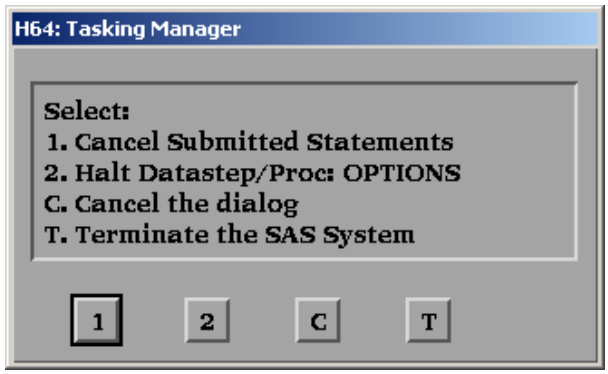
Interrupt を **SAS: Session Management** ダイアログボックスでクリックすると、PROC または DATA ステップが実行中でない場合、次の **Tasking Manager** ダイアログボックスが表示されます。

画面 8.3 *Tasking Manager ダイアログボックス*



PROC または DATA ステップが実行中の場合、次の **Tasking Manager** ダイアログボックスが表示されます。

画面 8.4 *Tasking Manager ダイアログボックス: DATA ステップまたは PROC ステップの実行中*



Tasking Manager ダイアログボックス内の次のいずれかのボタンをクリックします。

1
現在の PROC または DATA ステップステートメントが削除されます。

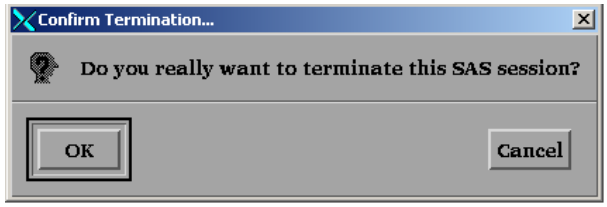
2
現在の PROC または DATA ステップが処理への割り込み要求を受け取ります。このアクションを確認するよう要求されます。

C
SAS 処理に影響を与えずにダイアログボックスを閉じます。

T
SAS に強制的に SAS セッションを終了させます。終了を確認するよう要求されます。

次の **Confirm Termination** ダイアログボックスが表示されます。

画面 8.5 Confirm Termination ダイアログボックス



OK をクリックすると、SAS Session Manager が、SAS セッションに UNIX シグナルを送信し、セッションを強制的に終了します。

注意:

SAS セッションを終了すると、データが損失または破損する場合があります。SAS セッションを終了する前に、"SAS の終了法" (26 ページ)に記載する方法で SAS を終了できないか確認してください。

SAS セッションからのホストエディタの使用

HOSTEDIT コマンドを発行すると、SAS はその要求を SAS Session Manager へ渡し、そこからホストエディタが呼び出されます。HOSTEDIT コマンドを適用するためには、SAS Session Manager が実行されている必要があります。HOSTEDIT コマンドを発行すると、アクティブな SAS ウィンドウのデータを含む一時ファイルを作成し、このファイルをホストエディタに渡します。(これらの一時ファイルは、SAS WORK オプションにより指定されたディレクトリに格納されます。) ホストエディタの中で保存されたファイルは、書き込み可能なウィンドウであれば、元の SAS ウィンドウにコピーされます。SAS セッションが終了すると、その一時ファイルは削除されます。詳細については、"UNIX 環境でホストエディタがサポートされるように SAS を構成する" (167 ページ)を参照してください。

SAS Session Manager を閉じる

SAS: Session Management ダイアログボックスを閉じると、SAS Session Manager を取得できません。SAS Session Manager を再度表示するには、**`!SASROOT/utilities/bin/motifxsassm`** を、*-pid* または*-sessionid* 引数で再起動してください。これらのコマンドは UNIX プロンプトで実行するか、または X ステートメントとともに使用してください。

```
!SASROOT/utilities/bin/motifxsassm -pid pid

!SASROOT/utilities/bin/motifxsassm -sessionid integer
```

SAS Session Manager の無効化

SAS Session Manager は、次の方法で無効化できます。

- **Tools ⇨ Options ⇨ Preferences** を選択します。

  General タブで、**Start Session manager** チェックボックスを選択解除します。

- 起動時に SAS コマンド行上で次の X リソースを小文字で指定します。

  ```
  sas -xrm 'SAS.startSessionManager: False'
  ```

  `sas.startSessionManager` X リソースを指定すると、**Preferences** ダイアログボックスの **Start Session manager** チェックボックスが選択解除されます。

注: SAS は、終了時、**Preferences** ダイアログボックスに設定値を保存します。セッション中に SAS Session Manager を無効にした場合、次回に SAS を起動すると、SAS Session Manager は実行されません。SAS Session Manager を起動するには、**Start Session manager** チェックボックス(**Preferences** ダイアログボックス内)を選択するか、または起動時に SAS コマンド行上に、次のコマンドをを小文字で指定します。

```
sas -xrm 'SAS.startSessionManager: True'
```

UNIX 環境でファンクションキーの定義を表示する

ファンクションキーの定義を割り当てる利点

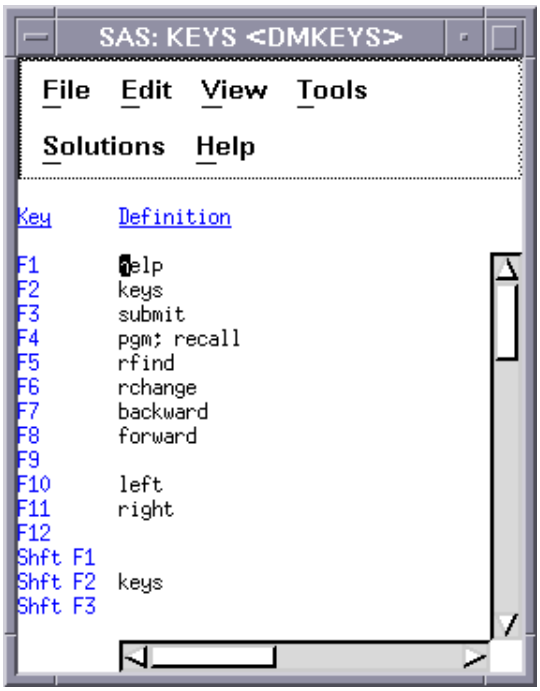
ファンクションキーを使用すると、コマンドに素早くアクセスできます。コマンドの発行、テキスト文字列の挿入、またプログラムの中へのコマンドの挿入が可能になります。ファンクションキーの定義は、端末によって異なる場合があります。これらの定義は、完全にカスタマイズできます。

ファンクションキーの定義の表示法

次のいずれかの KEYS (DMKEYS)ウィンドウを開いて、すべてのファンクションキーの定義を表示できます。

- **F2** を押します。
- KEYS コマンドを発行します。
- **Tools** ⇨ **Options** ⇨ **Keys** を選択します。

画面 8.6 *SAS: KEYS (DMKEYS) ウィンドウ*



KEYS ウィンドウを表示せずに単一のキーの定義を表示するには、KEYDEF コマンドを使用して、表示したいキーの定義を指定します。たとえば、次のコマンドでは、F4 キーの定義が表示されます。

```
keydef f4
```

キー定義のカスタマイズの詳細については、“UNIX 環境でキー定義をカスタマイズする” (190 ページ)を参照してください。**Keys** ウィンドウおよび KEYDEF コマンドの詳細については、オンラインの SAS ヘルプとドキュメントを参照してください。

UNIX 環境の SAS ツールボックス

SAS ツールボックスについて

SAS ツールボックスには、次のディスプレイに示す 2 つの部分があります。

画面 8.7 *SAS ツールボックス*

- アクティブな SAS ウィンドウに素早くコマンドを入力できるようにするコマンドウィンドウ。UNIX で使用可能なコマンドの詳細については、“UNIX 版に固有の SAS コマンド” (230 ページ) およびオンラインの SAS ヘルプとドキュメントの Base SAS セクションの SAS コマンドセクションを参照してください。
- 複数のツールアイコンを含むツールバー。ツールアイコンを選択すると、SAS はすぐに、そのアイコンに関連付けられたコマンドを実行します。ツールバーもツールアイコンもいずれもカスタマイズできます。詳細については、“Tool Editor の使用” (185 ページ)を参照してください。

アクティブウィンドウの名前は、SASToolBox のタイトルバーに表示されます。たとえば、**Log** ウィンドウがアクティブであると、タイトルバーには、SASToolBox: Program Editor ではなく、SASToolBox: Log と表示されます。

UNIX では、デフォルトの SASToolBox が、SAS ウィンドウスタックの底部に自動的に表示されます。その構成を管理するには、**Preferences** ダイアログボックスを使用します。(“SAS ToolBox 設定の変更 ” (176 ページ)を参照してください。)

デフォルト SASToolBox のカスタマイズ

デフォルトの SASToolBox は、ツールボックスをカスタマイズするかどうかにかかわりなく、ご使用の SASUSER.PROFILE.DMS.TOOLBOX に自動的にコピーされます。関連付けられた PMENU エントリを持たないアプリケーションを起動すると、そのアプリケーションについてデフォルトツールボックスが表示されます。次に、そのアプリケーション用にツールボックスをカスタマイズすると、そのカスタマイズされたツールボックスは、SASUSER.PROFILE.DEFAULT.TOOLBOX に保存されます。ここでは、DEFAULT は、そのウィンドウまたはアプリケーションの PMENU エントリと同じエントリ名です。

デフォルトの SAS ToolBox のカスタマイズ、複数のツールボックスの作成とそれらの切り替え、そしてアプリケーションのロード時に自動的にロードされるアプリケーション固有のツールボックス(SAS/AF アプリケーションをもつもの)の作成が可能です。一度に表示されるツールボックスは 1 つだけであり、ツールボックス内のツールは、アプリ

ケーションの切り替え時に変わります。詳細については、“UNIX 環境でツールボックスとツールセットをカスタマイズする” (183 ページ)を参照してください。

コマンドウィンドウとツールバーのデフォルト構成

デフォルトでは、ツールバーとコマンドウィンドウは結合され、また次の条件が適用されない限り、SAS 初期化時に自動的に表示されます。

- SAS ジョブを非ウィンドウ環境モードで実行しました。
- `SAS.defaultToolBox` または `SAS.defaultCommandWindow` リソースが **False** に設定されています。デフォルト値は **True** です。ツールボックスを管理するリソースの詳細については、“ツールボックスの動作を制御する X リソース” (184 ページ)を参照してください。
- **Preferences** ダイアログボックス内から **Display tools window**、**Display command window** または **Combine windows** を **ToolBox** タブを選択解除します。

次のディスプレイは、コマンドウィンドウおよびツールバーのデフォルト構成の状態が示しています。

画面 8.8 コマンドウィンドウとツールバーのデフォルト構成

コマンドウィンドウとツールバーを開く、閉じる

次のテーブルには、コマンドウィンドウおよびツールバーの開閉の手順を示しています。

表 8.1 コマンドウィンドウおよびツールバーの開閉の手順

ウィンドウ	開き方	閉じ方
コマンドウィンドウとツールバー	両方のウィンドウを開くには、次のいずれかのステップを実行します。 • COMMAND WINDOW コマンドを発行します。 • TOOLLOAD コマンドを発行します。詳細については、“TOOLLOAD コマンド: UNIX” (245 ページ)を参照してください。 • **Tools** ⇨ **Options** ⇨ **Toolbox** を選択します。	これらのウィンドウを閉じるには、次の手順を実行してください。 • `Close` を **ToolBox** ウィンドウメニューから選択します。 • “TOOLCLOSE コマンド: UNIX” (244 ページ)で記述されているように、TOOLCLOSE コマンドを入力します。 • **Tools** ⇨ **Options** ⇨ **Toolbox** を選択します。これにより、**ToolBox** が選択解除されます。

ウィンドウ	開き方	閉じ方
コマンドウィンドウ	コマンドウィンドウだけを開くには、**Preferences** ダイアログボックスの タブにある **Combine Windows** を選択解除し、次のいずれかの手順を実行します。 • **Preferences** ダイアログボックスの **ToolBox** タブにある **Display command window** を選択します。 • COMMAND WINDOWS コマンドを発行します。	コマンドウィンドウだけを閉じるには、次の手順を実行します。 • **Preferences** ダイアログボックスの **ToolBox** タブにある **Display command window** を選択解除します。 • ウィンドウメニューから **Close** を選択します。
ツールバー	ツールバーだけを開くには、**Preferences** ダイアログボックスの **ToolBox** タブにある **Combine windows** を選択解除し、次のいずれかの手順を実行します。 • **Preferences** ダイアログボックスの **ToolBox** タブにある **Display tools window** を選択します。 • TOOLLOAD コマンドを発行します。“TOOLLOAD コマンド: UNIX” (245 ページ)を参照してください。 • **Tools** ⇨ **Options** ⇨ **Toolbox** を選択します。	ツールバーだけを閉じるには、**Preferences** ダイアログボックスの **ToolBox** タブにある **Combine windows** を選択解除し、次のいずれかのステップを実行します。 • **Preferences** ダイアログボックスの **ToolBox** タブにある **Display tools window** を選択解除します。 • “TOOLCLOSE コマンド: UNIX” (244 ページ)で記述されているように、TOOLCLOSE コマンドを発行します。 • **Tools** ⇨ **Options** ⇨ **Toolbox** を選択します。これにより、**ToolBox** が選択解除されます。

コマンドの実行

コマンドは、コマンドウィンドウまたはツールバーのいずれかから実行できます。次のテーブルには、コマンドの実行方法に関する詳細が記載されています。

表 8.2 *コマンドウィンドウおよびツールバー内でのコマンドの実行*

場所	実行
コマンドウィンドウ	コマンドを実行するには、次の手順を実行します。 1. コマンドウィンドウをクリックします。 2. コマンドを入力します。 3. Enter を押すか、またはチェックマークをクリックします。 コマンドは、アクティブな SAS ウィンドウの中で実行されます。上矢印キーおよび下矢印キーを使用して以前入力したコマンドをスクロールするか、ドロップダウンリストから前のコマンドを選択できます。左マウスボタンを使用して、ドロップダウンリストからコマンドを選択します。右マウスボタンを使用して、リストからコマンドを選択・実行します。

場所	実行
ツールバー	コマンドを実行するには、ツールバーの中のツールアイコンをクリックし、そのアイコンに関連付けられたコマンドを実行します。**SAS.toolBoxTipDelay** リソースで指定された時間、アイコン上にカーソルを置いていると、ポップアップウィンドウが、そのアイコンのコマンドを説明するテキストを表示します。

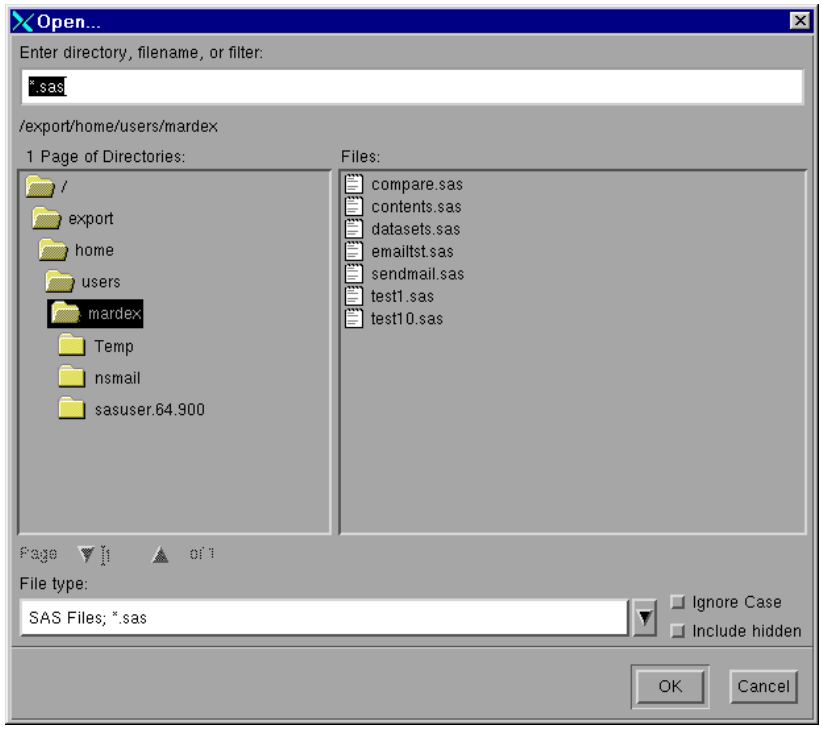
UNIX 環境でファイルを開く

Open ダイアログボックスを開く

ダイアログボックスを開く

Open ダイアログボックスにより、ホストファイルシステムからファイルを選択できます。このダイアログボックスを開くには、**File** ⇨ **Open** を選択します。

画面 8.9 *Open ダイアログボックス*



Open ダイアログボックスの各オプションの説明

次のテーブルには、**Open** ダイアログボックスにある各オプションの説明が記載されています。

表 8.3 Open ダイアログボックスの各オプション

オプション	説明
ディレクトリ、名前、またはフィルタの入力	ここでは、開く対象のディレクトリ、ファイル、またはファイルフィルタの名前を入力できます。 **Filter** フィールドに示されるディレクトリは、現在選択中のディレクトリです。このディレクトリは、**Page of Directories** リストから名前を選択するか、新規名をこのフィールドに直接入力することにより変更できます。このダイアログボックスでは、読み取り不能のディレクトリは、異なるアイコンで表示されます。 ディレクトリ内のすべてのファイルのリストを表示するには、**Filter** フィールドの中にアスタリスク(*)ワイルドカードを入力するか、またはファイルタイプとして **All Files; *** を選択してください。
ディレクトリのページ	**Filter** および **Page** フィールドに指定されたディレクトリの名前が含まれています。
ファイル	選択したディレクトリの中で、指定したフィルタに一致するファイルが含まれています。
ページ	**Page of Directories** リストに表示されたディレクトリを変更できます。**Page of Directories** リスト内のエントリ数が画面の高さの 2 倍を超えた時点で、新規ページが定義されます。ページを変更するには、**Page** フィールドの隣りの右矢印キーまたは左矢印キーを使用します。
ファイルタイプ	**Files** リストに表示したいファイルの種別を選択できます。このフィールドの下矢印キーを選択することで、使用可能なファイルフィルタのリストを表示できます。ファイルフィルタを選択するには、ファイルをクリックしてください。
大文字と小文字を区別しない	大文字の名前と小文字の名前の両方をディスプレイに含めることを指定します。(フィルタとして **All Files; *** を選択すると、**Ignore Case** を選択すると、大文字の名前と小文字の名前の両方が表示されます。)
非表示項目を含める。	選択されなかったファイルおよびディレクトリをグラフィカルディスプレイに含める、または除外します。

SAS リソースを使用して初期フィルタとディレクトリを指定する

File type フィールドの初期フィルタの指定は、`SAS.pattern` リソースに値を割り当てることで行えます。ただし、**Open** ダイアログボックスには、次の起動までの間そのフィルターが保持されるので、`SAS.pattern` リソースは、**Open** ダイアログボックスの初回の起動に対してのみ適用されます。また、`SAS.directory` リソースを使用することで、**Open** ダイアログボックスの初回呼び出し時に必要なディレクトリを指定できます。

SAS リソースの指定に関する詳細については、“X リソースの概要” (170 ページ)を参照してください。

ファイル名の正規表現の使用

Open ダイアログボックスに入力する内容はすべて、正規表現として扱われます。ファイルを開くか、または保存するときで、ファイル名の一部に正規表現の特殊文字を使用したいときは、その前にバックスラッシュ(\)を入力してください。たとえば、$Jan という名前のファイルに書き込むには、ファイル名として\$Jan を入力します。

正規表現の詳細については、UNIX man ページ 5 で regexp を参照してください。

```
man 5 regexp
```

UNIX 環境で作業ディレクトリを変更する

作業ディレクトリについて

作業ディレクトリは、多くの SAS コマンドおよびアクションが適用されるオペレーティングシステムディレクトリです。SAS のデフォルト設定では、SAS セッション開始時に、カレントディレクトリが作業ディレクトリとして使用されます。

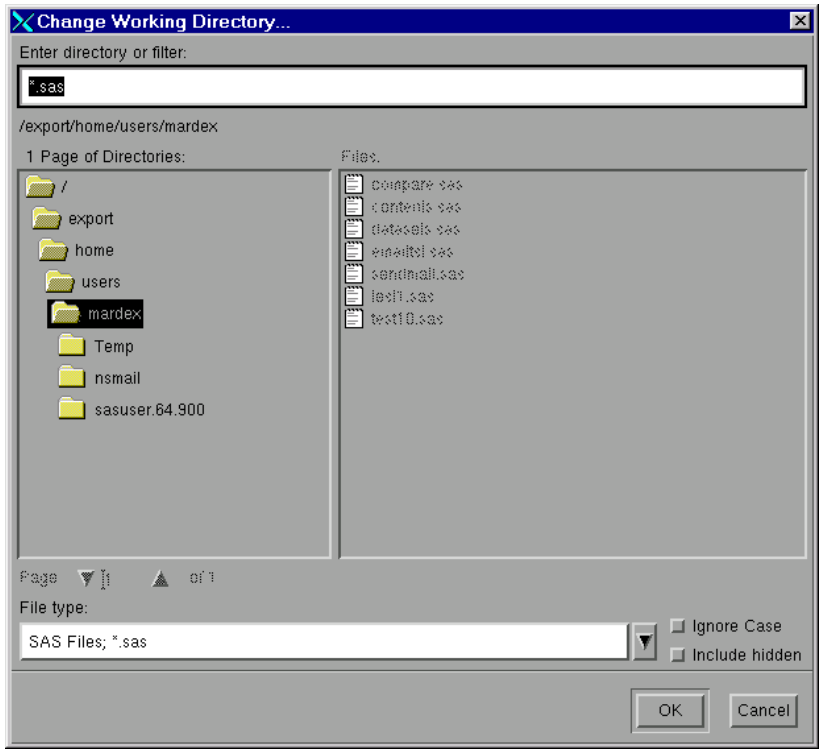
作業ディレクトリを変更する

この作業ディレクトリは、SAS セッション中に変更できます。**Change Working Directory** ダイアログボックスを使用して新しいディレクトリを選択できますが、X コマンド、X ステートメント、CALL SYSTEM ルーチン、または%SYSEXEC マクロステートメントを使用しディレクトリ変更(`cd`)コマンドを発行することもできます。X コマンドおよびステートメント、CALL SYSTEM ルーチン、ならびに%SYSEXEC マクロステートメントの詳細については、“SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)を参照してください。

Change Working Directory ダイアログボックス

Change Working Directory ダイアログボックスを開くには、DLGCDIR コマンドをを発行するか、**Tools** ⇨ **Options** ⇨ **Change Directory** を選択します。

画面 8.10 Change Working Directory ダイアログボックス



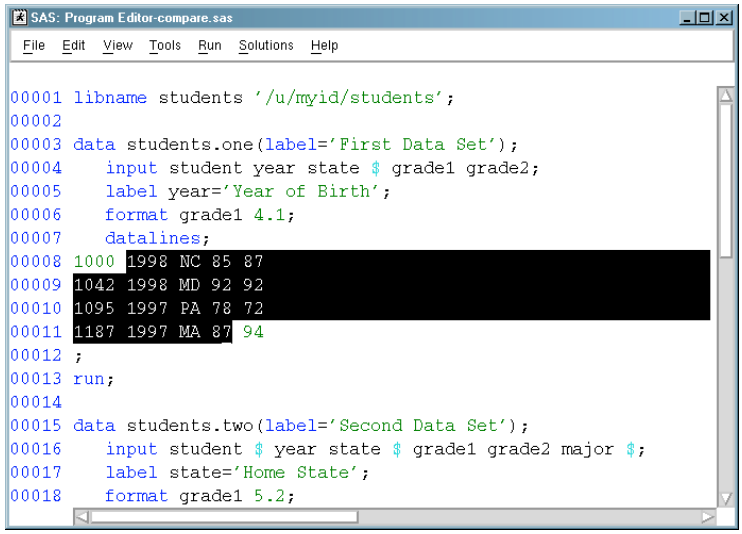
Change Working Directory ダイアログボックスの機能は、リストからファイルを選択できないことを除けば、**Open** ダイアログボックスの機能とまったく同じです。**Change Working Directory** ダイアログボックス内の各オプションの説明については、“Open ダイアログボックスの各オプションの説明” (156 ページ)を参照してください。

UNIX 環境でテキストを選択する(マークを付ける)

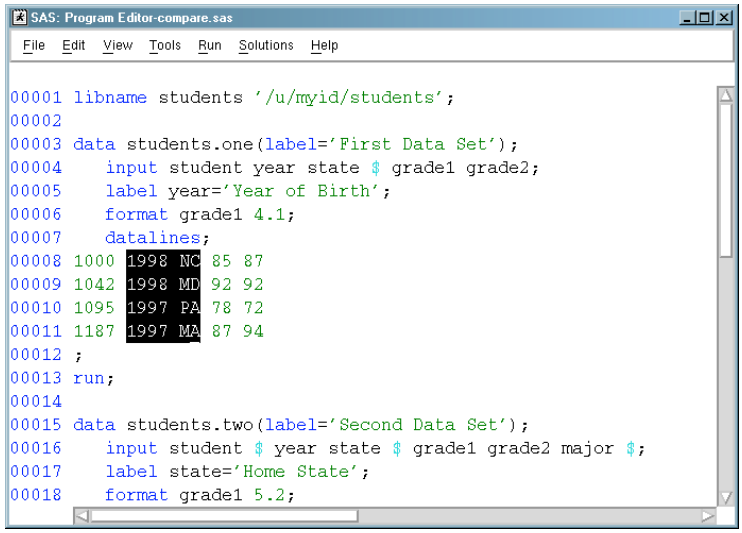
文字列のマーク付けとブロックのマーク付けの相違点

SAS ウィンドウでテキストを選択すると、文字列または文字ブロックを選択できます。文字列には、次のディスプレイにあるように、1 つ以上の行をもつ連続列の中のテキストも含まれます。ブロックとは、画面 8.12 (160 ページ)に示すように、連続行の同じ列を含む長方形のブロックです。

画面 8.11 マークされた文字列をもつ Program Editor ウィンドウ



画面 8.12 マークされたブロックをもつ Program Editor ウィンドウ



テキストの選択法

マウスを使用してテキストを選択する

テキストを選択するには、次の手順を実行してください。

1. カーソルを、マークしたいテキストの開始点に移動します。
2. 左マウスボタンを押したままにします。文字列ではなくブロックを選択する場合は、Ctrl キーを押したままで左マウスボタンを押してください。
3. マークしたいテキスト上でマウスポインタをドラッグします。
4. ALT キー(またはキーボードによっては EXTEND 文字キーまたは META キー)を押したままで、マウスボタンから手を離します。マウスにより生成されたマークは、ドラッグマークと呼ばれます。

マークされたテキストの領域を拡大するには、Shift キーを押したまま、左マウスボタンおよび Alt キー(ブロックをマークするときは Ctrl キー)を使用して、新しい終了位置をマークします。選択したテキストをマーク解除するには、ウィンドウ内でマウスボタンを押します。

MARK コマンドを使用してテキストを選択する

コマンド行から MARK コマンドを発行するか、またはそれをファンクションキーに割り当てることができます。MARK コマンドを使用すると、同一ウィンドウ内のテキストの複数領域を同時に選択できます。MARK コマンドの詳細については、オンラインの SAS ヘルプとドキュメントを参照してください。

テキストを選択するには、次の手順を実行してください。

1. カーソルを、マークしたいテキストの開始点に移動します。
2. MARK コマンドを発行します。文字列でなくブロックを選択する場合は、MARK コマンドに BLOCK 引数を付けてください。
3. カーソルを、マークしたいテキストの終了点に移動します。
4. 2 回目の MARK コマンドを発行します。

選択したテキストをマーク解除するには、UNMARK コマンドを発行します。

編集メニューを使用してテキストを選択する

Edit メニューを使ってテキストを選択するには、次の手順を実行してください。

1. カーソルを、マークしたいテキストの開始点に移動します。
2. **Edit** ⇨ **Select** を選択します。
3. カーソルを、マークしたいテキストの終了点に移動します。
4. 左マウスボタンを押します。

選択したテキストをマーク解除するには、**Edit** ⇨ **Deselect** を選択します。

UNIX 環境での選択したテキストのコピー、切り取り、貼り付け

選択したテキストのコピー法、切り取り法、貼り付け法

テキストをマークした後、それをコピーまたは切り取りをして、他の場所に貼り付けることができます。

- テキストをコピーするには、ツールボックスからコピーアイコンを選択し、STORE または WCOPY コマンドを発行するか、または **Edit** ⇨ **Copy** を選択します。
- テキストを切り取るには、ツールボックスから切り取りアイコンを選択し、CUT または WCUT コマンドを発行するか、また **Edit** ⇨ **Cut** を選択します。
- 切り取りまたはコピーしたテキストを貼り付けるには、ツールボックスから貼り付けアイコンを選択し、PASTE または WPASTE コマンドを発行するか、または **Edit** ⇨ **Paste** を選択します。

CUT、PASTE、および STORE コマンドの詳細については、オンラインの SAS ヘルプとドキュメントを参照してください。

SAS での自動貼り付けバッファの使用法

Alt キーを押したままにせずにマウスボタンから手を離すことによってドラッグマークを終了すると、マーク終了時のアクションを実行します。これにより、STORE コマンドが自動的に生成されて、マークの内容が SAS 貼り付けバッファに保存される場合があります。STORE コマンドが自動的に生成されると、そのテキストを明示的にコピーしなくても、貼り付けることができます。

自動貼り付けバッファの無効化

この自動貼り付けバッファは、次の方法で無効にできます。

- **SAS.markPasteBuffer** リソースを設定します。
- **Automatically store selection** を選択解除します。これは、**Editing** タブを **Preferences** ダイアログボックス内で、**Tools** ⇨ **Options** ⇨ **Preferences** と選択します。

詳細については、“UNIX 環境での切り取りと貼り付けのカスタマイズ” (209 ページ)を参照してください。

SAS とその他の X クライアント間のテキストのコピーと貼り付け

デフォルトの SAS 貼り付けバッファを X 固有の貼り付けバッファに関連付ければ、X クライアント間でテキストを切り取りまたはコピーして貼り付けられます。. たとえば、デフォルトの SAS 貼り付けバッファを貼り付けバッファに関連付けると、xterm ウィンドウと SAS ウィンドウ間でテキストをコピーして貼り付けられます。SAS バッファを X バッファに関連付けるには、**SAS.defaultPasteBuffer** リソースを指定します。

```
SAS.defaultPasteBuffer: XTERM
```

貼り付けバッファの使用の詳細については、“UNIX 環境での切り取りと貼り付けのカスタマイズ” (209 ページ)を参照してください。

UNIX 環境でドラッグアンドドロップを使用する

デフォルトと非デフォルトのドラッグアンドドロップの違い

UNIX 上の SAS ウィンドウ環境には、デフォルトと非デフォルトの 2 種類のドラッグアンドドロップがあります。デフォルトのドラッグアンドドロップでは、テキストを 1 つの場所から他の場所へ移動できます。非デフォルトのドラッグアンドドロップでは、テキストの移動またはコピー、SAS コードをドラッグしている場合はそのテキストの送信、またはドラッグアンドドロップ操作のキャンセルの中から選択できます。デフォルトのドラッグアンドドロップを使用すると、異なるセッション間の SAS ウィンドウ間、そしてドラッグアンドドロップをサポートする Netscape などのような Motif アプリケーションと SAS ウィンドウ間でテキストをドラッグできます。非デフォルトのドラッグアンドドロップは、同じ SAS セッション内のウィンドウ間でのみ使用できます。

UNIX 環境でのドラッグアンドドロップの制限

UNIX 環境では、ファイルまたは RTF(Rich Text Format)テキストをドラッグアンドドロップできません。

テキストのドラッグアンドドロップ法

テキストをドラッグアンドドロップするには、“UNIX 環境でテキストを選択する(マークを付ける)” (159 ページ)に記載されたいずれかの方法で、最初にテキストをマークします。デフォルトのドラッグアンドドロップを使用するには、中央マウスボタンを使用して必要な場所までテキストをドラッグします。非デフォルトのドラッグアンドドロップを使用するには、Alt (または EXTEND CHAR)キーを押したままで、マウスボタンから手を離します。

UNIX 環境でテキスト文字列を検索し、置換する

Find ダイアログボックスと Replace ダイアログボックスについて

Find と **Replace** ダイアログボックスを使用すると、SAS テキストエディタウィンドウ(**Program Editor**、SCL エディタ、または **NOTEPAD**)内で文字列の検索・置換を行えます。

Find ダイアログボックスを開く

文字列を検索するには、DLGFIND コマンドを発行して **Find** ダイアログボックスを開くか、**Edit** ⇨ **Find** を選択します。

Find ダイアログボックスの各オプションの説明

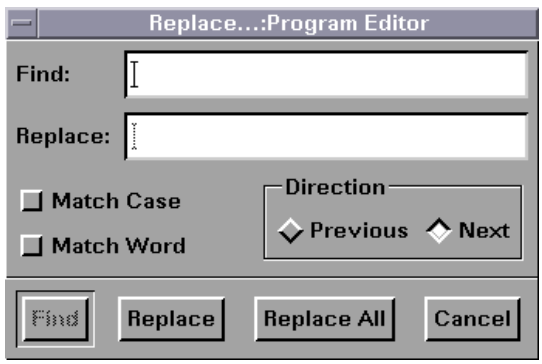
Find ダイアログボックスは、**Replace** ダイアログボックスと同様に使用できますが、**Replace** フィールド、または **Replace** および **Replace All** ボタンがありません。

Find ダイアログボックス内の各オプションの説明は、“Replace ダイアログボックスのオプションの説明 ” (164 ページ)を参照してください。

Replace ダイアログボックスを開く

1 つの文字列を他の文字列で置換するには、DLGREPLACE コマンドを発行して **Replace** ダイアログボックスを開くか、**Edit** ⇨ **Replace** を選択します。

画面 8.13 *Replace ダイアログボックス*



Replace ダイアログボックスのオプションの説明

文字列を検索するには **Find** フィールドに文字列を入力して、**Find** をクリックします。文字列を変更するには、**Find** フィールドに文字列を入力し、**Replace** フィールドに置換文字列を入力してから、**Replace** をクリックします。その文字列をすべてその置換文字列に変更するには、**Replace All** をクリックします。

次のボタンで検索操作と置換操作をカスタマイズできます。

大文字と小文字を区別する
検索機能に、入力したのとまったく同じように大文字と小文字を区別するよう指定します。

単語単位で検索する
スペース、行の末尾、ファイルの末尾の文字により区切りされた指定文字列の検索。

前
現在のカーソル位置からファイルの先頭に向かって検索します。

次
現在のカーソル位置からファイルの末尾に向かって検索します。

UNIX 環境で SAS セッションからメールを送信する

SAS のデフォルト電子メールプロトコル

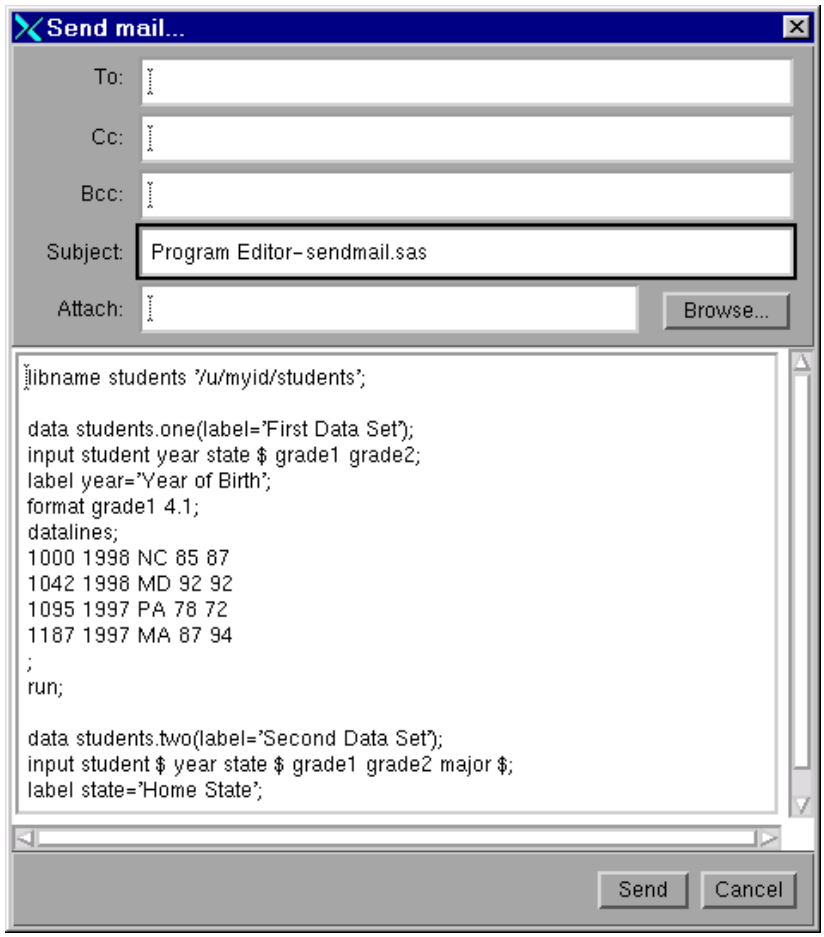
デフォルトでは、SAS は SMTP (Simple Mail Transfer Protocol)を使用して、現在の SAS セッションの中から電子メールを送信します。EMAILSYS システムオプションを使用して、電子メールの送信に使用するスクリプトまたはプロトコルを選択できます。詳細については、“EMAILSYS システムオプション: UNIX” (375 ページ)を参照してください。

SMTP 電子メールインタフェースの詳細については、*SAS 言語リファレンス: 解説編*を参照してください。

Send Mail ダイアログボックスについて

Send Mail ダイアログボックスを使用すると、現在の SAS セッションから出ることなく、電子メールを送信できます。このダイアログボックスを呼び出すには、DLGSMAIL コマンドを発行するか、**File** ⇨ **Send Mail** を発行します。

画面 8.14 Send Mail ダイアログボックス



Send Mail ダイアログボックスを使用した電子メールの送信

電子メールを送信するには、必要に応じて次の手順を実行してください。

1. e-mail 受信者の ID を **To**、**Cc** および **Bcc** フィールドに入力します。複数のアドレスは、スペースまたはカンマで区切ってください。
2. **Subject** フィールド内の入力内容は、必要に応じて編集してください。
3. 送信したいファイル名を **Attach** フィールド に入力します。複数のファイル名はスペースで区切ります。また、**Browse** 機能を使用してファイルを選択できます。

   注: 電子メール添付ファイルの送信をサポートしていない外部スクリプトもあります。
4. メッセージ領域にメッセージを入力するか、アクティブな SAS ウィンドウから取得した内容を編集します。
5. **Send** をクリックします。

メッセージをキャンセルには、**Cancel** をクリックします。

テキストのウィンドウコンテンツを送信する

アクティブな SAS テキストウィンドウ(**Program Editor** または**ログ**など)の内容は、**Send mail** ダイアログボックスを使用してメール送信できます。**Send mail** ダイアログボックスを開くには、**File** ⇨ **Send Mail** を選択します。SAS は、アクティブな SAS ウィンドウの内容を自動的にコピーし、電子メールの本文にそのテキストを含めます。電子メールメッセージの変更または追加は、**Send mail** ダイアログボックスの中で行えます。

アクティブな SAS ウィンドウの内容をメッセージの中に含めたくない場合は、**Edit** ⇨ **Clear All** を選択してから、**Send mail** ダイアログボックスを呼び出してください。

非テキストウィンドウの内容の送信

非テキストウィンドウ(SAS/GRAPH で生成されたグラフ、または PROC REPORT 出力からのイメージ)の内容を送信するには、アクティブな SAS ウィンドウから **File** ⇨ **Send Mail** を選択します。SAS が、そのイメージデータを一時ファイルに自動的にコピーし、そのファイル名を **Send mail** ダイアログボックスの **Attach** フィールドに入力します。この一時ファイルのデフォルトのファイルタイプを変更するには、“デフォルトのファイルタイプの変更” (166 ページ) を参照してください。

SAS は、イメージのうち、アクティブウィンドウ内に表示される部分だけを、ウィンドウフレームおよびタイトルと一緒にコピーします。この動作は、DLGSCRDUMP コマンドを使用したときと同じです。詳細については、“DLGSCRDUMP コマンド: UNIX” (236 ページ)を参照してください。

このイメージを電子メールに添付したくない場合は、**Attach** フィールドの内容をクリアしてください。

注: 電子メール添付ファイルの送信をサポートしていない外部スクリプトもあります。

デフォルトのファイルタイプの変更

SAS により作成される一時ファイルのデフォルトのファイルタイプは、**Preferences** ダイアログボックスを使用することで変更できます。**Preferences** ダイアログボックスを開くには、次の手順を実行してください。

1. **Tools** ⇨ **Options** ⇨ **Preferences** を選択します。
2. **Image type for Email attachments** ボックス **DMS** タブ上で、次のいずれかのファイルタイプを選択します。
   - Portable Network Graphics (.png)
   - Graphics Interchange Format (.gif)
   - Tagged Image File Format (.tif)

UNIX 環境でホストエディタがサポートされるように SAS を構成する

ホストエディタの使用要件

SAS では、Motif インターフェイスをもつホストテキストエディタの使用がサポートされているので、SAS セッションで vi または Emacs などのエディタを使用できます。デフォルトのホストエディタとして設定されたホストエディタはありません。そのため、この機能を使用するにはこれを指定する必要があります。ホストエディタのサポートには、motifxsassm クライアントの利用が必要です。.詳細については、“UNIX 環境の SAS Session Manager (motifxsassm)” (149 ページ)を参照してください。

ホストエディタの呼び出しと使用

ホストエディタの開き方と使用法

SAS でホストエディタを使用するには、次の手順を実行してください。

1. EDITCMD システムオプションを使ってエディタを呼び出すために必要なコマンドを指定します。
2. 必要に応じて、HOSTEDIT コマンドでエディタを呼び出してください。

HOSTEDIT コマンドでは、データが SAS ウィンドウからホストエディタへ渡されます。ホストエディタの中で保存されたデータは、書き込み可能なウィンドウであれば、元の SAS ウィンドウにコピーされます。

SAS テキストエディタウィンドウに戻った後に UNDO コマンドを発行すれば、ホストエディタで加えたすべての変更を元に戻せます。HOSTEDIT コマンドを発行する前のウィンドウの状態に戻るには、UNDO コマンドを再度発行する必要があります。HOSTEDIT コマンドを読み取り専用ウィンドウで発行すれば編集内容を外部ファイルに保存できますが、SAS テキストエディタウィンドウには変更ありません。

詳細については、“EDITCMD システムオプション: UNIX” (375 ページ) および “HOSTEDIT コマンド: UNIX” (240 ページ)を参照してください。

例 1: SAS を起動して HOSTEDIT コマンドで xedit を使用する

xedit と呼ばれる X ベースのエディタをもつシステムもあります。HOSTEDIT コマンドで xedit を使用したい場合は、次のコマンドで SAS を起動してください。

```
sas -editcmd '/usr/local/bin/xedit'
```

例 2: SAS を起動して vi を使用する

vi エディタは、ターミナルウィンドウが必要な、ターミナルベースのエディタです。xterm クライアントの -e オプションは、xterm クライアント起動時にプログラムを実行します。EDITCMD オプションを使用して vi とともに xterm クライアントを表示するには、SAS を次のようにして起動します。

```
sas -editcmd '/usr/bin/X11/xterm -e /usr/bin/vi'
```

テキスト属性の転送のトラブルシューティング

色や強調表示などのテキスト属性は、ホストエディタウィンドウと SAS テキストエディタウィンドウ間で転送されません。HEATTR ON コマンドを発行して、テキストの編集で使用した強調表示や色属性がホストエディタ側で削除されることを警告するダイアログボックスが表示されるようにします。このダイアログボックスでは、続行するか、または HOSTEDIT コマンドを中止することを要求します。このダイアログボックスを非表示にするには、HEATER OFF を指定します。

UNIX 環境でヘルプを利用する

Help メニューは、SAS セッション内で常に利用できます。次に示すのは、**Help** メニューから利用可能なヘルプトピックの説明です。

このウィンドウの使用
アクティブウィンドウに関連するヘルプ情報が提供されます。**Help** ボタンをクリックするか、F1 キーを押すことにより同じ情報にアクセスできます。

SAS ヘルプとドキュメント
SAS の使用法の習得を助けるチュートリアルとサンプルプログラム、サイトにインストールされたすべての製品の総括ドキュメントおよび SAS に他のサポートを依頼する場合の情報を掲載しています。

注: ブラウザの **Preferences** ダイアログボックス内で要求されていないポップアップウィンドウをブロックするオプションを設定する場合には、オンラインの SAS ヘルプとドキュメントが表示されない場合があります。

SAS ソフトウェア入門ガイド
SAS の入門に役立つチュートリアルが開かれます。

Web 上の SAS
カスタマサポートセンタ、FAQ、SAS へのフィードバック送信および SAS ホームページなど、SAS Web サイト上の便利なエリアへのリンクが提供されます。(参照: テクニカルサポート Web サイト)

SAS 9 について
SAS のバージョン情報 SAS ソフトウェア、オペレーティング環境および Motif に関する情報を提供する 9 ダイアログボックス

9 章

SAS ウィンドウ環境のカスタマイズ

X 環境における SAS カスタマイズの概要

SAS ウィンドウ環境では、X ベースのグラフィカルユーザーインターフェイス(GUI)の使用がサポートされます。UNIX 環境において、SAS は、Motif スタイルに基づいた X Window System インターフェイスを提供します。X 環境での SAS の詳細については、“X 環境の SAS について” (146 ページ)を参照してください。

作業環境をカスタマイズするには、X リソースを使います。

X リソースの概要

X リソースについて

通常、X クライアントには、カスタマイズ可能な機能があります。これらのプロパティは X リソースと呼ばれます。SAS は X クライアントとして機能するため、SAS ウィンドウ環境の表示と動作の多くの局面は、X リソースによって制御されます。たとえば、X リソースを使用して、フォント、背景色、ウィンドウのサイズを定義できます。SAS のようなアプリケーションのリソースは、リソースデータベース内に置かれます。

SAS は、リソースデータベースを修正しなくても、正しく機能します。ただし、インターフェイスのデフォルトの動作と表示を変更する場合もあります。カスタマイズの指定には、いくつかの方法があります。特定の X サーバー上で表示されるすべての SAS セッションを変更する方法もあります。特定のホスト上での実行される SAS セッションに影響する方法もあります。また、単一の SAS セッションにのみ、影響する方法もあります。

X Window System のクライアントおよび X リソースについての詳細は、ベンダが提供するドキュメントを参照してください。

X リソースを指定する構文

リソースの指定には、次の形式を使用します。

resource-string: *value*

resource string には、通常 2 つの識別子と 1 つの区切り文字が含まれます。最初の識別子は、クライアント名またはアプリケーション名 (SAS)です。区切り文字は、ピリオド(.)またはアスタリスク(*)文字です。2 番目の識別子は特定のリソースの名前です。*value* は、リソースの種類により、ブール値(True または False)、数字、文字列のいずれかになります。

アプリケーション名とリソース名ともに、インスタンス値またはクラス値を指定できます。クラスの指定は、単一のインスタンスよりも大きな範囲に適用されます。

次にサンプルのリソース指定を示します。

```
SAS.startSessionManager: True
SAS.maxWindowHeight: 100
SAS.awsResizePolicy: grow
```

リソース指定についての詳細は、X Window System ドキュメントを参照してください。

X リソースのカスタマイズ法

次のリストでは、X リソースをカスタマイズするのに使用できる方法を記述します。

- SAS セッションをカスタマイズするには、**Font** ダイアログボックス、**Preferences** ダイアログボックスまたは **Resource Helper** を使用します。これらのツールすべてにより、X リソース定義は次に SAS セッションを起動するときに読み込まれる場所へと書き込まれます。これらのツールの詳細については、“Preferences ダイアログボックスを使用し、X リソースを変更する” (172 ページ)、“Resource Helper を使用し、X リソースを設定する” (178 ページ)、および “UNIX 環境でフォントをカスタマイズする” (198 ページ)を参照してください。

  注: **Preferences** ダイアログボックス内で指定する設定は、あらゆるコマンド行設定よりも優先されます。

- セッション特有のリソースを指定するには、SAS を起動するごとに、コマンド行上で **-xrm** オプションを使用します。たとえば、次のコマンドで、SAS セッションを終了するときに、**Confirm** ダイアログボックスの非表示を指定します。

  ```
  sas -xrm 'SAS.confirmSASExit: False'
  ```

  -xrm オプションは、必要な回数だけ指定できます。各リソースには、**-xrm** オプションを指定する必要があります。

  注: 通常、シェルスクリプトで SAS を起動する場合、バックスラッシュ(\)文字を使用して、シェルから引用符を保護してください。

  ```
  sasscript -xrm \'SAS.confirmSASExit: False\'
  ```

- ホームディレクトリ内のファイルにリソース定義を追加します。アプリケーション起動時に X ツールキットが通常検索するファイル内にリソースを置いておくと、それらのリソースは SAS 起動時にロードされます。X ツールキットのリソース検索場所については、X Window System のドキュメントを参照してください。

また、SAS の初期化後にリソースデータベースにリソースを追加するには、**xrdb** ユーティリティを実行します。たとえば、次のコマンドでは、MyResources ファイル内の定義がリソースデータベースへとマージされます。

```
xrdb -merge myresources
```

- サブディレクトリを作成して、リソース定義を保存します。(通常、このサブディレクトリは **app-defaults** と名付けられます。) XUSERFILESEARCHPATH 環境変数を、このサブディレクトリのパス名に設定します。XUSERFILESEARCHPATH 環境変数の指定時に、**%N** を使用して、アプリケーションクラス名をファイルのかわりにできます。XUSERFILESEARCHPATH 環境変数が、起動する各シェルに定義されることを確認するために、シェル($HOME/.login、$HOME/.cshrc または $HOME/.profile などのファイル)の初期化ファイル内で環境変数の定義を指定します。

  XUSERFILESEARCHPATH に識別されるサブディレクトリ内で **SAS** というファイルを作成します。このファイル内にリソース定義を含めてください。

  注: もう 1 つの方法として、XAPPLRESDIR 環境変数を、リソース定義が保存されているサブディレクトリのパス名に設定できます。XAPPLRESDIR 環境変数および XUSERFILESEARCHPATH 環境変数では、少し異なる構文を使用して、リソース定義の場所が指定されます。XUSERFILESEARCH 環境変数で指定された場所は、XAPPLRESDIR 変数で指定された場所よりも優先されます。詳細については、UNIX X の man ページを参照してください。

- 特定のホストで全ユーザーに使用されるカスタマイズされたリソース定義が必要な場合は、**SAS** というファイルを作成してリソース定義を含め、そのファイルをシステムの **app-defaults** ディレクトリ内に格納します。

X リソースの詳細については、ベンダが提供する X Window System ドキュメントまたは X Window System についてのその他のドキュメントを参照してください。

Preferences ダイアログボックスを使用し、X リソースを変更する

Preferences ダイアログボックスについて

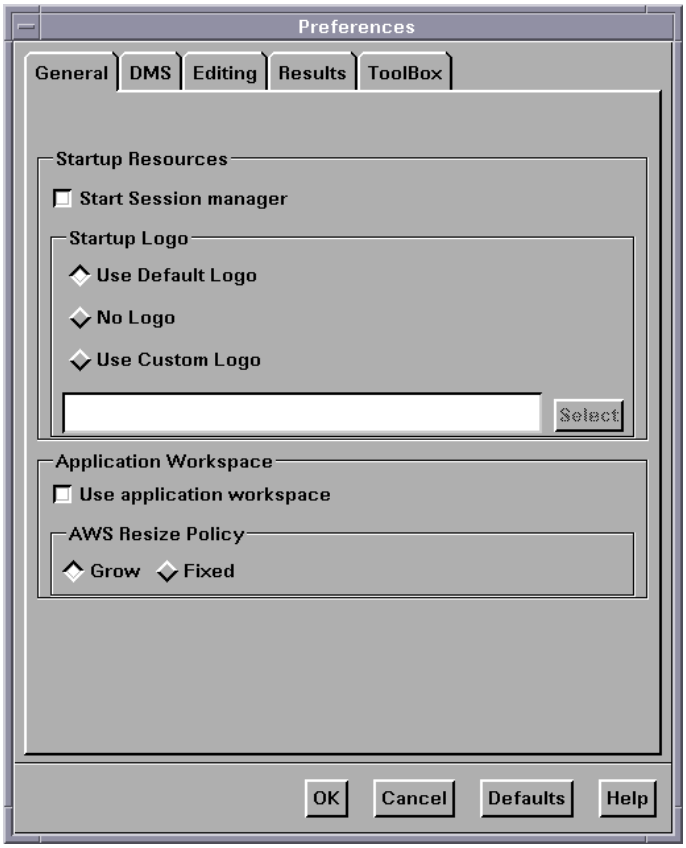
Preferences ダイアログボックスにより、特定の X リソースの設定を管理できます。**Preferences** ダイアログボックスでなされた変更は **General** タブ上のリソースを除いて)、すぐに有効になり、設定は Sasuser ディレクトリ内の SasuserPrefs ファイルに保存されます。

注: **Preferences** ダイアログボックス内で指定する設定は、現在のセッションのあらゆるコマンド行設定よりも優先されます。

Preferences ダイアログボックスを開く

Preferences ダイアログボックスを開くには、DLGPREF コマンドを発行するか、**Tools** ⇨ **Options** ⇨ **Preferences** と選択します。

画面 9.1 Preferences ダイアログボックス



Preferences ダイアログボックスのオプションの説明

General 設定の変更

全般的な設定を変更するには、**Preferences** ダイアログボックス内の **General** タブを選択し、ウィンドウ内で次の項目を選択します。

Start Session manager

SAS セッションを開始するときに、SAS Session Manager が起動するかどうか指定します。SAS セッション内でホストエディタを使うには、SAS Session Manager が実行中である必要があります。SAS Session Manager では、SAS セッションへの割り込みや終了を行ったり、SAS セッション内のすべてのウィンドウを最小化したり元に戻したりできます。詳細については、"UNIX 環境の SAS Session Manager (motifxsassm)" (149 ページ) および "UNIX 環境でホストエディタがサポートされるように SAS を構成する" (167 ページ) を参照してください。**Start Session manager** ボックスをクリックすると `SAS.startSessionManager` リソースが設定されます。

Startup Logo

SAS セッションを初期化中に XPM ファイルを表示させるかを指定します。また、その場合に、どのファイルを使うかも指定します。

Use Default Logo を選択すれば、サイトにはデフォルトのファイルが使用されます。**No Logo** を選択する場合、どのファイルも表示されません。**Use Custom Logo** を選択すると、テキストフィールド内の XPM ファイル名に直接入力するか、**Select** をクリックして **File Selection** ダイアログボックスを表示できます。このボックスを選択すると、`SAS.startupLogo` リソースが設定されます。

Use application workspace
1 つのアプリケーションによる全ウィンドウが単一アプリケーションワークスペースに表示されるように制限します。このボックスを選択すると、`SAS.noAWS` リソースが設定されます。このリソースへの変更を有効にするには、ウィンドウを閉じてからもう一度開きます。

注: UNIX 動作環境では、AWS (アプリケーションワークスペース)はデフォルトで有効に設定されています。**EFI** ウィンドウを使用中でウィンドウの正確な位置とサイズを記憶させるの場合は、AWS を無効に設定します。これをするには、**Tools** ⇨ **Options** ⇨ **Preferences** を選択し、**Use application workspace** を選択解除します。**EFI** ウィンドウでの作業が終了した後、必ず AWS をデフォルト設定に戻したことを確認してください。

AWS Resize Policy
内部ウィンドウが追加/削除される際の、AWS ウィンドウのサイズ変更方法を指定します。(詳細については、"SAS セッションのワークスペースとグラビティ" (147 ページ) および "ウィンドウの種類" (148 ページ)を参照してください。)

Grow
AWS ウィンドウは、内部ウィンドウを拡大または移動するとき常に(すべての内部ウィンドウが見えるように)拡大しようとしますが、使用しない領域を取り除くために縮小することはありません。

Fixed
AWS ウィンドウは、最初の内部ウィンドウのサイズに自身でサイズ調整し、それ以降はサイズ変更しません。

このボックスを選択すると、`SAS.awsResizePolicy` リソースが設定されます。

DMS 設定の変更

DMS タブの設定を変更するには、**Preferences** ダイアログボックス内の **DMS** タブを選択し、ウィンドウ内で次の項目を選択します。

Use menu access keys
メニューのニーモニックを有効にします。ニーモニックが有効のときは、メニュー項目を選ぶには項目内の下線が引かれた単一の文字を入力します。このボックスを選択すると、`SAS.usePmenuMnemonics` リソースが設定されます。

Confirm exit
SAS セッションを終了するときに、**Exit** ダイアログボックスを表示します。このボックスを選択すると、`SAS.confirmSASExit` リソースが設定されます。

Save Settings on Exit
SAS セッションを終了するときに、SAS が WSAVE ALL コマンドを発行するようにします。このコマンドにより、ウィンドウの色と位置のような全体的な設定が保存されます。これらの設定は現在開いているすべてのウィンドウに有効です。これらの設定は、Sasuser.Profile カタログに保存されます。このボックスを選択すると、`SAS.wsaveAllExit` リソースが設定されます。

注: WSAVE コマンドが機能するには、ウィンドウマネージャによって、正確なウィンドウ配置をサポートされている必要があります。ウィンドウマネージャの設定方法を決定するには、ウィンドウマネージャのドキュメントを参照してください。たとえば、Exceed を実行中の場合、**Screen Definition Settings** ダイアログボックスを開き、**Cascade Windows** の選択を解除します。

Backup Documents
現在開いているドキュメントを、`SAS.autoSaveInterval` リソースで指定された間隔で自動的に保存するかどうか指定できます。このボックスを選択すると、`SAS.autoSaveOn` リソースが設定されます。

Image type for Email attachments
電子メールを使って非テキストウィンドウの内容を送る際に SAS によって作成される一時ファイルのデフォルトの種類を指定します。非テキストウィンドウの例には、SAS/GRAPH によって生成されたグラフまたは PROC REPORT 出力からの画像が含まれます。詳細については、“非テキストウィンドウの内容の送信” (166 ページ)を参照してください。

Editing 設定の変更

Editing の設定を変更するには、**Preferences** ダイアログボックス内の **Editing** タブを選択し、ウィンドウ内で次の項目を選択します。

Default paste buffer
デフォルトの SAS バッファのエイリアスを定義します。次の一覧では、貼り付けバッファのエイリアス名と各名前に関連する XPM ファイルバッファについて説明します。

XPRIMARY
X のプライマリ選択(**PRIMARY**)

XSCNDARY
X のセカンダリ選択(**SECONDARY**)

XCLIPBRD
X のクリップボード(**CLIPBOARD**)

XTERM
xteam クライアントで使用されるプロトコルを交換します。

XCUTn
X の切り取りバッファ。*n* は 0 と 7 の間をすべて含む数字です。

このボックスを選択すると、**SAS.defaultPasteBuffer** リソースが設定されます。カットアンドペーストバッファについての詳細は、“UNIX 環境でドロップダウンメニューを制御する” (209 ページ)を参照してください。

Automatically store selection
マウスでテキストの範囲をマークするたびに、STORE コマンドを生成します。このボックスを選択すると、**SAS.markPasteBuffer** リソースが設定されます。

Cursor
SAS テキストエディタウィンドウ内の編集モードを制御します。**Insert** ボックスと **Overtype** ボックスを選択すると、**SAS.insertModeOn** リソースをそれぞれ **True** と **False** に設定します。

Results 設定の変更

Results 設定を変更するには、**Preferences** ダイアログボックス内の **Results** タブをクリックします。**Results** タブの項目は ODS で作成された出力に影響します。(ODS の詳細な説明については、*SAS Output Delivery System: ユーザーガイド*を参照してください。) このダイアログボックス内の項目から、次のものを選択します。

Create Listing
固定出力を生成する、ODS LISTING の出力先を開きます。このボッ k スを選択することは、ODS LISTING SELECT ALL ステートメントを入力することと同義です。

Create HTML
ODS HTML 出力先を開きます。ここで、HTML でフォーマットされる出力が生成されます。HTML はデフォルトの出力の種類です。

Folder
HTML ファイルの出力先ディレクトリを指定します。このフィールド内のディレクトリを指定することは、ODS HTML ステートメント内の PATH オプションでディレクトリを指定することと同義です。

Use WORK Folder
ODS が **Work** ディレクトリへと、すべての HTML ファイルを送るようにします。このボックスを選択することは、HTML ステートメント内の PATH オプションで **Work** ディレクトリのパス名を指定することと同義です。

Style
HTML 出力に使用するスタイルテンプレートを指定します。スタイルテンプレートでは、色、フォント名、フォントサイズなどの要素を制御します。このフィールド内のスタイルを指定することは、ODS HTML ステートメント内の STYLE オプションでスタイルを指定することと同義です。SAS レジストリ内の \ODS\PREFERENCES\STYLES キーで定義される、どのスタイルをも指定できます。SAS レジストリを開くには、REGEDIT 発行するか、または **Solutions** ⇨ **Accessories** ⇨ **Registry Editor** と選択します。

デフォルトのスタイルは HTMLTMLBlue です。

View results as they are generated
出力結果が生成されるときに、自動的にそれを表示するようにします。このボックスを選択する場合、**Password protect HTML file browsing** の選択が解除されていることを確認してください。

Password protect HTML file browsing
ブラウザに HTML ファイルを送る前に、パスワードの入力を促します。このボックスを選択する場合、**View results as they are generated** の選択が解除されていることを確認してください。このボックスを選択すると、**SAS.htmlUsePassword** リソースが設定されます。

Use ODS Graphics
ODS グラフィックをサポートするプロシジャを実行する際に自動的にグラフを生成することができます。**Use ODS Graphics** はデフォルトでは有効になっています。

SAS ToolBox 設定の変更

Preferences ダイアログボックスの **ToolBox** タブ上の項目は、ToolBox とコマンドウィンドウの両方に影響します。これらの設定を変更するには、**Preferences** ダイアログボックス内の **ToolBox** タブを選択します。

Display tools window
デフォルトのツールボックスを表示するかどうかを指定します。このチェックボックスを選択すると、**SAS.defaultToolBox** リソースが設定されます。

Display command window
コマンドウィンドウを表示するかどうかを指定します。このチェックボックスを選択すると、**SAS.defaultCommandWindow** リソースが設定されます。

Auto Complete Commands
以前に入力したコマンドと同じ文字で始まるコマンドをコマンドウィンドウに入力する際、コマンドの残りの文字を SAS が自動的に補うかどうかを指定します。このボックスと **Save Commands** が同時に選択される場合、以前のセッションに入力されたコマンドが自動的に書き込まれます。このチェックボックスを選択すると、**SAS.autoComplete** リソースが設定されます。

Save Commands
コマンドウィンドウに入力するコマンドが保存されるかどうか、および、保存されるコマンドの数を指定します。0 から 50 までの数を指定できます。0 を指定すると、コマンドは一切保存されません。1 以上を指定すると、Sasuser ディレクトリの

commands.hist ファイルに、その数のコマンドが保存されます。このボックスが選択されていると、以前のセッションで入力されたコマンドは SAS が自動的に補います。(**Auto Complete Commands** を参照してください。) このフィールドを選択すると、**SAS.commandsSaved** リソースが設定されます。

Combine windows
ToolBox とコマンドウィンドウを、1 つのウィンドウへと結合します。ToolBox とコマンドウィンドウはデフォルトで結合されています。このチェックボックスを選択すると、**SAS.useCommandToolBoxCombo** リソースが設定されます。

Use arrow decorations
結合した ToolBox とコマンドウィンドウ両方の末尾に矢印を追加します。このチェックボックスを選択すると、**SAS.useShowHideDecorations** リソースが設定されます。

Always on top
ToolBox または結合した ToolBox とコマンドウィンドウをウィンドウの重なりの手前に常に表示させます。デフォルトでは、このチェックボックスは選択されていますが、ウィンドウマネージャとその他のアプリケーションをウィンドウの重なりの手前に表示させるときに問題が発生する可能性があります。このような状態になった場合、この機能を無効にしてください。このチェックボックスを選択すると、**SAS.toolBoxAlwaysOnTop** リソースが設定されます。

Toolbox Persistent
Program Editor を閉じる際、**Program Editor** に関連する ToolBox が開いたままであるかどうかを指定します。デフォルトでは、**Program Editor** ウィンドウを閉じる際には、常に **Program Editor** ToolBox は開いたままになります。このボックスを選択解除すると、**Program Editor** を閉じる際に ToolBox が閉じます。このチェックボックスを選択すると、**SAS.isToolBoxPersistent** リソースが設定されます。

Tools 内の項目は、ToolBox 内の個々のツールに影響します。

Use large tools
ツールアイコンが 24x24 または 48x48 ピクセルで表示されるかどうかを指定します。デフォルトでは 24x24 ピクセルです。このチェックボックスを選択すると、**SAS.useLargeToolBox** リソースが設定されます。

Use tip text
ツールボックス内のツール上にカーソルを置くときに、ツールチップテキストを表示するかどうかを指定します。ウィンドウマネージャの中には、ツールボックスの後ろにツールボックスチップを配置する場合があるものもあります。ツールボックスチップが、使用環境内でツールボックスの後ろに置かれる場合は、このボックスの選択を解除してください。このチェックボックスを選択すると、**SAS.useToolBoxTips** リソースが設定されます。

delay
ツールボックスチップが表示される前の遅延をミリ秒単位で指定します。このチェックボックスを選択すると、**SAS.toolBoxTipDelay** リソースが設定されます。フィールド内に直接、値を入力または、フィールドの右側へと矢印を使って値を変更できます。

Resource Helper を使用し、X リソースを設定する

Resource Helper について

Resource Helper を使用して、キー定義と SAS 対話型インターフェイスの色をカスタマイズできます。Resource Helper により、SAS リソース定義が作成され、Resource Manager により検索できる場所に保存されます。Resource Helper がリソース定義を検索する場所の一覧は、“Resource Helper による X リソースの検索法 ” (182 ページ) を参照してください。Resource Helper で保存されたリソース設定は、SAS セッションの次回起動時に有効になります。

Resource Helper は、SAS セッションまたはシェルプロンプトから開始できます。

Resource Helper の開始方法

SAS セッションから Resource Helper を開始する

SAS Resource Helper を SAS ウィンドウから開始するには、コマンドウィンドウのコマンド行で次のコマンドを入力します。

```
reshelper
```

画面 9.2　Resource Helper のメインウィンドウ

シェルプロンプトから Resource Helper を起動する

Resource Helper は、SAS がインストールされているディレクトリ(**!SASROOT**)の/**utilities/bin** サブディレクトリにインストールされます。実行可能なモジュールの名前は **reshelper** です。たとえば、**/usr/local/sas94** に SAS がインストールされている場合、次のコマンドを入力して Resource Helper を開始します。

```
/usr/local/sas94/utilities/bin/reshelper &
```

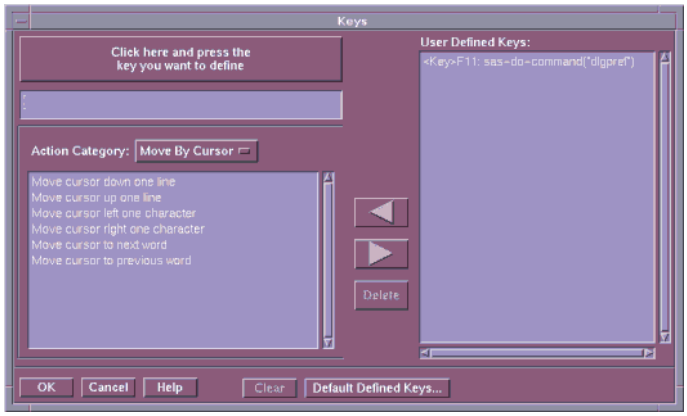
Resource Helper を使用してキーの定義

キーの定義法

キーを定義するには、次の手順に従います。

1. Resource Helper を開始(を参照 (178 ページ))して、Keys アイコンを選択します。

画面 9.3 *Resource Helper の Keys ウィンドウ*



キー定義は、いくつかの **Action Categories** へと分類されます。

- **Move By Cursor**
- **Move By Field**
- **Edit**
- **Miscellaneous**
- **All Actions**

2. **Click here and press the keys you want to define** を選択します。
3. 操作を割り当てるキー、またはキーの組み合わせを押します(たとえば、F12 を押す)。デフォルトの SAS の変換がすでにそのキーの組み合わせに割り当てられていた場合、Resource Helper により、デフォルトの変換が表示されます。
4. Action Category メニューのボタンを選択して、操作カテゴリの一覧を開きます。任意の操作カテゴリを選択します。たとえば、現在のフィールドを削除するためのキーを定義する場合、**Action Category** として **Edit** を選択します。Resource Helper はそのカテゴリの操作のリストを表示します。
5. そのリストから操作を1つ、たとえば **Delete current field** を選択します。Resource Helper では、1 つの変換に対して、1 つのみ操作を割り当てられます。選択する操作が、引数(`sas-action-routine` など)を必要とする場合、Resource Helper により、引数の入力が促されます。

   Resource Helper により、キーの組み合わせと、その新しい定義が表示されます。

   ```
   None<Key>F12: sas-delete()
   ```

   注: `sas-action-routine` sas-function-key 操作ルーチンを選択する場合、キー定義は自動的に **Keys** ウィンドウに表示されます。他の操作ルーチンを選択し、**Keys** ウィンドウにその定義を表示する場合、そのキーのウィンドウラベルを定義する必要があります。**Keys** ウィンドウでのラベルの定義については、"SAS.keysWindowLabels リソースの構文" (194 ページ)を参照してください。
6. 右矢印を選択して、**User-Defined Keys** の一覧へとキー変換を追加します。
7. キー変換の定義が完了した後、**OK** をクリックして、**Keys** ウィンドウを終了します。
8. 永続的に変換を保存する場合は、Resource Helper のドロップダウンメニューから、**File** ⇨ **Save Resources** を選択します。

User-Defined Keys 一覧にすでに存在するキー定義を変更する場合は、その定義を選択し、左矢印を選択して一覧からその定義を削除し、定義を編集します。

User-Defined Keys から定義を削除するには、その定義を選択し、**Delete** をクリックします。

Clear では、キー定義編集ウィンドウが消去されます。

Default Defined Keys では、使用するシステムのデフォルトのキー定義が表示されます。

不適切なキー定義のトラブルシューティング

ほとんどの場合で、Resource Helper の使用はユーザー自身でリソースを定義するよりも簡単かつ高速です。ただし、X Window System はさまざまな場所のリソースを検索するため、ユーザーが定義しようとするキーに対して、Resource Helper が間違ったキー符号を取得する可能性もあります。Resource Helper を使用して予期しない結果が得られたときは、ユーザーによるキーリソースの定義が必要な場合があります。詳細については、“キー変換の定義” (191 ページ)を参照してください。

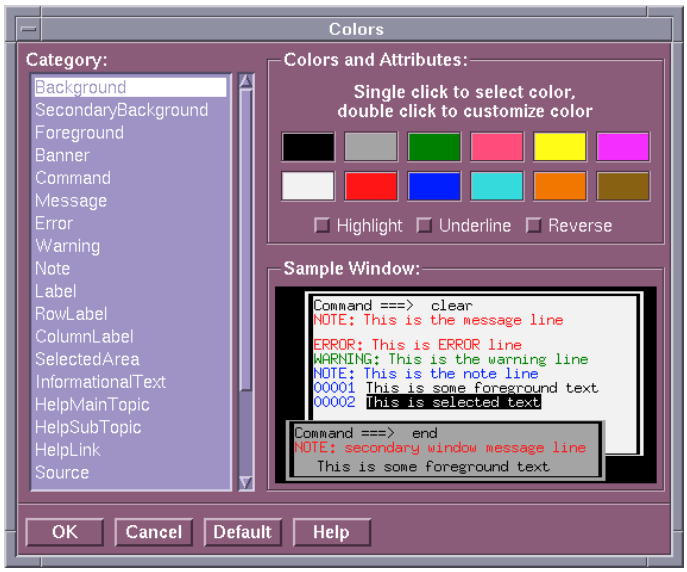
Resource Helper を使用して SAS ウィンドウの色を変更

Color ウィンドウの使用法

次の方法で、SAS ウィンドウの一部の色を変更できます。

1. Resource Helper を開始し、Colors アイコンを選択します。

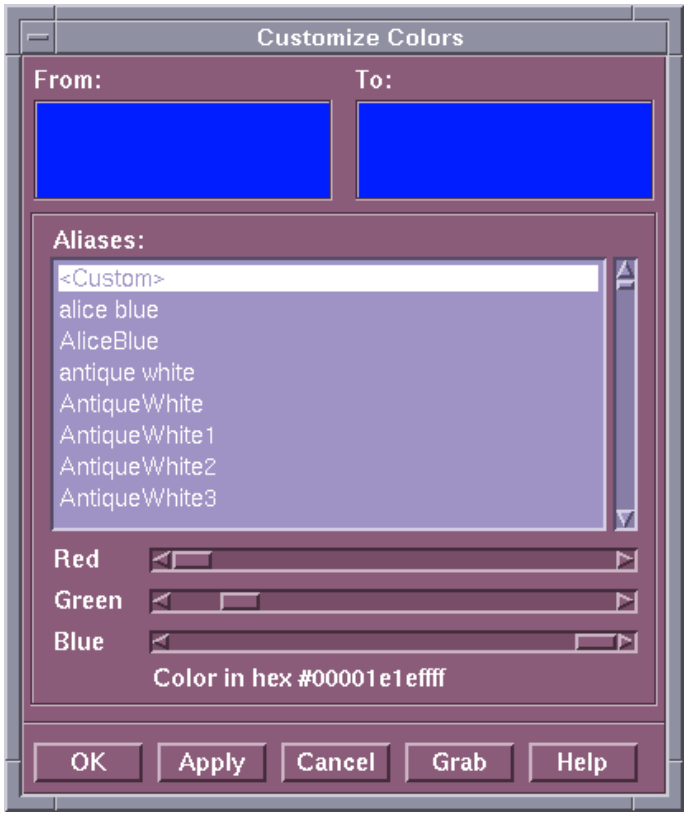
   画面 9.4 *Resource Helper の Colors ウィンドウ*



2. **Category** エリアから、カテゴリを選択します。
3. 次の図にあるように、**Colors and Attributes** ウィンドウで、色または属性をクリック、あるいは、色をダブルクリックして、**Customize Colors** エリアを開きます。

   SAS ウィンドウのいくつかのカテゴリの属性を変えることもできます。属性オプションは、HIGHLIGHT、UNDERLINE または REVERSE です。

画面 9.5 Resource Helper の Colors ウィンドウのカスタマイズ



次のようにして、色をカスタマイズできます。

- 新しい **Alias を選択します。**
- **Red、Blue** または **Green** スライダを移動します。
- **Grab** を選択し、画面上のどこかで色をクリックします。

4. 色設定の定義が完了した後、**OK** をクリックして、**Customize Colors** ウィンドウを終了します。

`Sample Window` 内に結果が表示されます。その色の 16 進数の値がウィンドウの最下部に表示されます。

例: SAS ウィンドウの色の変更

次の例では、SAS ウィンドウの色の変更方法を示します。

1. **Customize Colors** ウィンドウ内の **Red** をダブルクリックします。

   From:欄に現在 SAS ウィンドウ環境に使用されている赤が表示されます。

2. **Aliases** の下の **Aquamarine** をクリックし、**To:**欄内の変化を見ます。
3. マウスで **Red、Green** および **Blue** のスライダを動かし、**To:**欄内の色の変化に注目します。
4. **Apply** をクリックし、Colors エリア内で **Red** として表示される色の違いに注目します。
5. **OK** をクリックして、変更を保存します。

デフォルトの設定に戻す

Defaults をクリックして、色の設定をデフォルトの値に復元します。

色の設定の保存

永続的に色の設定を保存する場合は、Resource Helper のドロップダウンメニューから、**File** ⇨ **Save Resources** を選択します。

Resource Helper による X リソースの検索法

次の一覧では、Resource Helper がリソース定義を検索する場所と、それらの場所を検索する順序について説明します。

1. Resource Helper により、XENVIRONMENT 環境変数で示されたファイル内のリソースがロードされます。XENVIRONMENT が設定されていない場合、Resource Helper により、**~/.Xdefaults-***hostname* ファイル内のリソースがロードされます。*hostname* は Resource Helper が実行されているサーバーの名前です。

2. Resource Helper により、RESOURCE_MANAGER プロパティ内で定義されたリソースがロードされます。Resource Helper がリソースを検索する最初の場所が RESOURCE_MANAGER プロパティである場合、Resource Helper を使用して生成するどのリソースよりも RESOURCE_MANAGER プロパティが優先されます。

   RESOURCE_MANAGER プロパティ内であらゆるリソースが定義されていたかどうかを知るには、次のコマンドを実行します。

   ```
   xrdb -q | more
   ```

   一覧が返ってこない場合は、RESOURCE_MANAGER プロパティは存在しません。この場合、Resource Helper は、**~/.Xdefaults** ファイル内で定義されたリソースをロードします。

3. Resource Helper により、XUSERFILESEARCHPATH 環境変数で示されたファイル内のリソースがロードされます。

   XUSERFILESEARCHPATH 環境変数の指定時に、**%N** を使用して、アプリケーションクラス名をファイルのかわりにできます。たとえば、任意のアプリケーションに対するすべてのリソースの場所として **/usr/local/resources** を指定するには、Bourne シェルまたは Korn シェルで次のコマンドを発行します。

   ```
   export XUSERFILESEARCHPATH=\
   /usr/local/resources/%N
   ```

   C シェルでは、コマンドは次のようになります。

   ```
   setenv XUSERFILESEARCHPATH \
   /usr/local/resources/%N
   ```

   結果として、SAS の起動時に、XUSERFILESEARCHPATH によって指定されるファイルは次のとおりです。

   ```
   /usr/local/resources/SAS
   ```

   SAS は SAS のアプリケーションクラス名です。

4. Resource Helper により、XAPPLRESDIR 環境変数で指定されたファイル内のリソースがロードされます。そのアプリケーションのクラス名は、XAPPLRESDIR 環境変数に付加され、出力される文字列はリソースの検査に使用されます。たとえば、Bourne シェルまたは Korn シェルで次のコマンドを発行できます。

   ```
   export XAPPLRESDIR=/usr/local/app-defaults
   ```

   これを行う場合、SAS の次回起動時に、アプリケーションのクラス名はパスに付加されます。

```
/usr/local/app-defaults/SAS
```

C シェルでは、コマンドは次のようになります。

```
setenv XAPPLRESDIR /usr/local/app-defaults
```

5. Resource Helper は、**~/SAS** という名前のファイル内のリソースをロードします。
6. Resource Helper により、XFILESEARCHPATH 環境変数で指定されたファイルまたは代替物内のリソースがロードされます。

   注: 環境変数が設定されていたか知るには、次のコマンドを実行します。

   ```
   env|grep <environment_variable>
   ```

7. Resource Helper は、**/usr/lib/X11/app-defaults** という名前のファイル内のリソースをロードします。Resource Helper は、このファイルへの書き込みアクセス権限を持つ必要はありませんが、このファイルを読み出せる必要があり、また、書き込みアクセス権限を持つリソースファイルへと SAS リソースを付加する必要があります。このファイルが存在しない場合または Resource Helper がこのファイルを読み出せない場合、Resource Helper によって警告メッセージは生成されません。
8. Resource Helper により、SAS コードで定義される代替システムのリソースがロードされます。

/usr/lib/X11/app-defaults ファイルを除いて、Resource Helper は、SAS リソースが最初に検出された場所と同じディレクトリとファイルへ新しいリソースを書き込もうとします。この場所は、書き込みアクセス権限を持つファイルおよび、書き込みアクセス権限を持つディレクトリである必要があります。Resource Helper がそのファイルに書き込めない場合、そのファイルにある SAS リソースは有効なまま残り、Resource Helper が生成した新規または変更リソースはすべて無効になります。この状況が起こる場合、Resource Helper により、ファイルまたはディレクトリを含むエラーダイアログボックスが表示され、問題解決のための手段が提示されます。

UNIX 環境でツールボックスとツールセットをカスタマイズする

ツールボックスのカスタマイズ法

次の方法でツールボックスをカスタマイズできます。

- **Preferences** ダイアログボックスを使用します。**Preferences** ダイアログボックスにより、ツールボックスの外観と動作をカスタマイズできます。**Preferences** ダイアログボックスの使用については、“Preferences ダイアログボックスを使用し、X リソースを変更する” (172 ページ) および “SAS ToolBox 設定の変更 ” (176 ページ)を参照してください。
- リソースファイル内の SAS リソースを指定します。ツールボックスに影響する SAS リソースの詳細については、“ツールボックスの動作を制御する X リソース” (184 ページ)を参照してください。
- Tool Editor を使用します。Tool Editor により、ツールボックス内の個々のツールをカスタマイズできます。詳細については、“Tool Editor の使用” (185 ページ)を参照してください。

ツールボックスの動作を制御する X リソース

次の SAS リソースを使用して、ツールボックスの動作を制御できます。

SAS.autoComplete: True | False
以前に入力したコマンドと同じ文字で始まるコマンドをコマンドウィンドウに入力する際、コマンドの残りの文字を SAS が自動的に補うかどうかを指定します。デフォルト値は True です。

SAS.commandsSaved: number-of-commands-saved
コマンドウィンドウに入力するコマンドが保存されるかどうか、および、保存されるコマンドの数を指定します。0 から 50 までの数を指定できます。0 を指定すると、コマンドは一切保存されません。1 以上を指定すると、Sasuser ディレクトリの **commands.hist** ファイルに、その数のコマンドが保存されます。このリソースに 1 以上を指定し、**SAS.autoComplete** が True の場合、以前のセッションで入力されたコマンドが、SAS により自動的に書き込まれます。デフォルトの値は 25 です。

SAS.defaultToolBox: True | False
SAS 起動時に、デフォルトのツールボックスを開くかどうか指定します。デフォルトは True です。

SAS.isToolBoxPersistent: True | False
Program Editor を閉じる際、Program Editor に関連するツールボックスが開いたままであるかどうかを指定します。デフォルト値は True です。

SAS.toolBoxAlwaysOnTop: True | False
ツールボックスを重なったウィンドウの一番手前に表示するかどうかを指定します。デフォルトの値 True は、Motif インターフェイスまたは他のアプリケーションではないウィンドウマネージャが、重なったウィンドウの手前に表示しようとすると問題が生じる可能性があります。このような状況の場合は、このリソースを False に設定してください。

SAS.toolBoxTipDelay: delay-in-milliseconds
ツールボックスチップが表示される前の遅延をミリ秒単位で指定します。デフォルトは 750 です。

SAS.useCommandToolBoxCombo: True | False
コマンドウィンドウとツールボックスの結合または分離を指定します。**SAS.defaultToolBox** リソースおよび **SAS.defaultCommandWindow** リソースで、ツールボックスとコマンドウィンドウの表示を制御します。ツールボックスとコマンドウィンドウ両者が同時に表示される場合、このリソースで結合または分離を指定します。デフォルト値は True です。

SAS.useLargeToolBox: True | False
ツールボックス内のツールアイコンが 24x24 または 48x48 ピクセルで表示されるかどうかを指定します。デフォルトは False(24x24 ピクセル)です。

SAS.useShowHideDecorations: True | False
結合したコマンドウィンドウとツールボックスウィンドウが左右に矢印を持つかどうかを指定します。これらの矢印を使用し、必要に応じて、ウィンドウの一部をしたり表示させたりできます。デフォルトの値は False です。

SAS.useToolBoxTips: True | False
ツールボックスのチップテキストの表示を指定します。ウィンドウマネージャの中には、ツールボックスの後ろにツールボックスチップを配置する場合があるものもあります。ツールボックスのチップが、使用環境内でツールボックスの後ろに置かれる場合は、このリソースを False に設定してください。デフォルトは True です。

Tool Editor の使用

ツールセットについて

Tool Editor により、SAS アプリケーションにカスタムツールセットを作成できます。ツールセットは、アプリケーションに関連づけられる、あらかじめ定義されたツールのセットです。ツールセットを使用すると、各ユーザーはそれぞれのアプリケーションツールボックスを簡単にカスタマイズできるようになります。アプリケーションにツールセットを作成する場合、Tool Editor 内の **Actions** を選択し、ツールボックス内で表示させるツールを選べます。これらのツールに、アイコン、コマンド、チップテキスト、ID を定義する必要はありません。

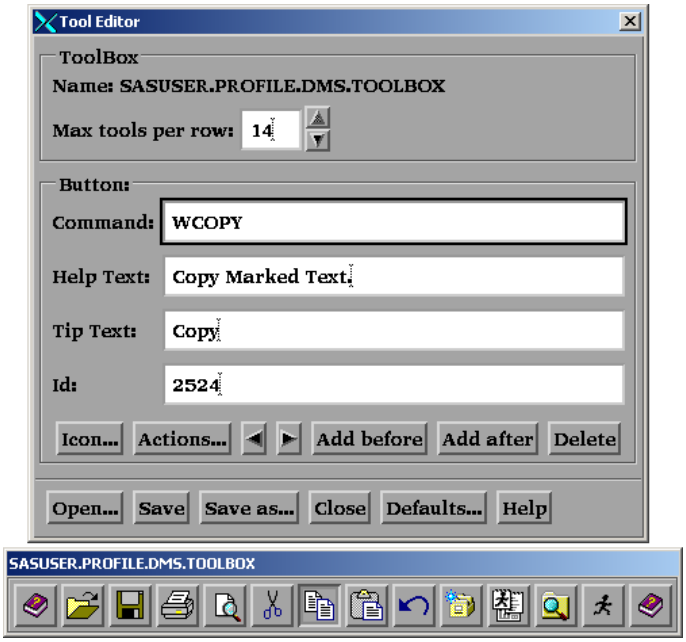
たとえば、ファイルを開く、テキストを切り取り、コピー、貼り付ける、ファイルを保存するなどのためのツールを含むアプリケーションに、デフォルトのツールボックスを定義できます。これらのツールを含むツールセットおよび、**Preferences** ダイアログボックスを開く、**Replace** ダイアログボックスを開く、RECALL コマンドを入力するなどのためのツールを定義できます。ただし、これらの追加のツールは、Tool Editor を使用してツールボックスに加えなければ表示されません。詳細については、“既存のツールの属性の変更” (186 ページ) および “アプリケーションまたはウィンドウ固有のツールセットの作成またはカスタマイズ” (190 ページ)を参照してください。

Tool Editor を呼び出す

ツールボックスの外観と内容を変更するには、Tool Editor を使用します。Tool Editor を呼び出すには、**Tools** ⇨ **Options** ⇨ **Edit Toolbox** と選択します。あるいは、“TOOLEDIT コマンド: UNIX” (245 ページ)内で記述されている TOOLEDIT コマンドを発行できます。

次の図で、**Program Editor** ウィンドウの **Tools** メニューから開かれた **Tool Editor** ダイアログボックスの例を示します。

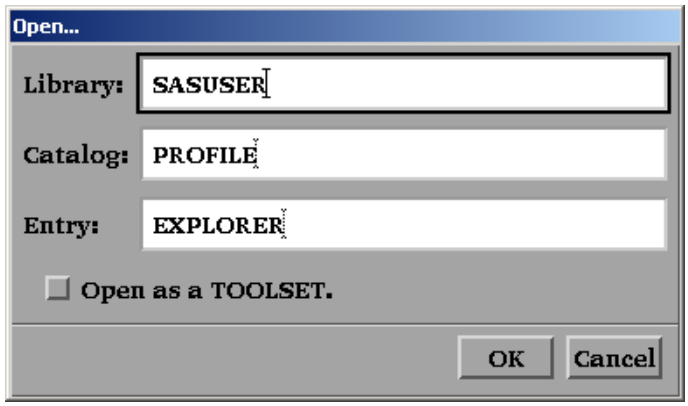
画面 9.6 *Tool Editor ダイアログボックス*



デフォルトでは、**Tool Editor** により、現在のツールボックスを編集できます。異なるツールボックスを編集するには、**Tool Editor** ダイアログボックス内の **Open** ボタンをクリ

ックします。編集するツールボックスの、ライブラリ、カタログおよびエントリ名を指定します。次の図では、**Open** ダイアログボックスを示します。

画面 9.7 *Open ダイアログボックス*



Tool Editor を呼び出した後

Tool Editor の呼び出し後、ツールボックスはプレビューモードで表示されます。プレビューモードで、ツールアイコンをクリックしツールを選択すると、そのアイコンが現在のアイコンになります。そのアイコンに関連するコマンドが、**Command** フィールド内に表示されます。また、ツールの作成や更新時にその情報が追加されたかどうかによって、**Help Text**、**Tip Text**、および **ID** フィールド内の属性が表示される場合もあります。**Tool Editor** ダイアログボックス内のフィールドとボタンについての詳細は、**Help** ボタンをクリックしてください。

ツールボックス全体の表示画面の変更

Tool Editor の **ToolBox** エリア内の項目はツールボックス全体に影響を与えます。

Name
編集中のカタログエントリを表示します。デフォルトのツールボックスの名前は、SASUSER.PROFILE.DMS.TOOLBOX です。

Max tools per row
ツールボックス内のアイコンの配置方法を指定します。デフォルトの値では、ツールボックスは水平になります。行ごとの 1 つのツールにより、ツールボックスは垂直になります。

既存のツールの属性の変更

Tool Editor を開くと、ツールセット内の最初のツールが選択され、このツールの属性は **Tool Editor** ダイアログボックスの **Button** エリアに表示されます。ツール内の別のアイコンをクリックすると、**Tool Editor** により、そのツールの属性が表示されます。

あるいは、**Actions** ボタンをクリックするときに表示されるツールセットから、ツールを選択できます。ツールを選択後に **OK** をクリックすると、**Tool Editor** の **Button** エリア内の属性が、対応する新しいツールへと更新されます。

注: **Actions** ボタンをクリックすると、ツールセットが編集中のツールボックスに関連付けられた場合(同じエントリ名を持つ)のみ、ツールセットが表示されます。詳細については、“ツールボックスまたはツールセットの変更の保存” (188 ページ)を参照してください。

変更するツールがすでに選択されている場合、**Tool Editor** の属性フィールドを選択し、必要な変更を入力します。

ツールの属性を変更するには、次の手順に従ってください。

1. ツールセットから、変更するツールを選択します。
2. **Button** エリアで、ボタンに関連する属性フィールドを選択し、テキストを適切に変更します。

   Command
   アイコンをクリックするときに実行されるコマンドを指定します。UNIX で利用可能なウィンドウ環境コマンドを使用できます。すべての動作環境で有効なコマンドについては、SAS ヘルプとドキュメントを参照してください。コマンドをセミコロン(;)で区切ります。たとえば、アイコンを作成して **Change Working Directory** ダイアログボックスを開くには、DLGCDIR コマンドを使用します。

   Help Text
   Windows 環境で実行するためにデザインされたアプリケーションに使用されます。ツールボックスがこれらのウィンドウにポートされてロードされると、ヘルプテキストが Windows の AWS ステータスバーに表示されます。

   Tip Text
   カーソルをアイコン上に置くときに表示されるテキストを指定します。

   ID
   SAS/AF アプリケーションにツールボックスを作成している場合に役立ちます。ID は、アプリケーション内の対応するメニューの識別子です。この数字は、PROC PMENU 内の ITEM ステートメントの ID オプションにある項目に割り当てられる値です。ID を指定する場合、アプリケーションにより、PMENU 項目の状態が設定され、ツールボックス内のツールの状態と一致します。ID により、PMENU の項目はアクティブ化または非アクティブ化され、ツールボックス内のツールがアクティブか非アクティブなのかに一致します。ID を指定しない場合、ID の初期値は 0 に設定されます。
3. 必要であれば、アイコンを変更します。
   a. **Icon** ボタンをクリックまたはプレビューツールボックス内のアイコンをダブルクリックします。**Tool Editor** により、**Select a pixmap** ダイアログボックスが開かれます。これは SAS に供給されるアイコンを表示します。これらのアイコンは SAS ウィンドウ、データ、分析、数字と記号、ファイル、フォルダといった複数のカテゴリに分類されます。カテゴリを変更するには、**Icon Category** フィールドの右に矢印を選択し、新しいカテゴリを選択します。
   b. 使用するアイコンを選択し、それから **OK** を選択します。

   次の図で、**Select a pixmap** ダイアログボックスを示します。

4. “ツールボックスまたはツールセットの変更の保存” (188 ページ)で記述されているように、変更を保存します。

ツールボックスへのツールの追加

ツールボックスにツールを追加するには、次の手順に従います。

1. 新しいツールを追加する場所の隣のアイコンを選択します。
2. **Add before** または **Add after** を選択します。**Tool Editor** により、新しいアイコンがツールボックスに追加され、ボタンフィールドがクリアされます。
3. “既存のツールの属性の変更” (186 ページ)で説明されているように、ボタンフィールドに適切な情報を入力してください。
4. “既存のツールの属性の変更” (186 ページ)で説明されているように、必要であれば、アイコンの属性を変更します。
5. “ツールボックスまたはツールセットの変更の保存” (188 ページ)で記述されているように、変更を保存します。

ツールボックス内のツールの順序を変更

ツールボックス内のツールの位置を変更するには、ツールアイコンを選択し、それから左または右の矢印をクリックしてツールを動かします。

ツールボックスからのツールの削除

ツールボックスからツールを削除するには、次の手順に従います。

1. 削除するツールを選択します。
2. **Delete** をクリックします。
3. “ツールボックスまたはツールセットの変更の保存” (188 ページ)で記述されているように、変更を保存します。

デフォルトの設定に戻す

現在のツールボックス内のツールをデフォルトの設定に戻すには、**Defaults** をクリックします。**Tool Editor** に確認するよう求められます。**Yes**、**No** または **Cancel** をクリックします。

ツールボックスまたはツールセットの変更の保存

Name フィールドで示されるカタログエントリへと変更を保存、または異なる名前で新しいツールボックスを作成できます。

ウィンドウに特有、あるいはアプリケーションに特有のツールボックスやツールセットを自身で使うためにカスタマイズしている場合、ウィンドウまたはアプリケーションに PMENU エントリと同じエントリ名を使用して、Sasuser.Profile カタログ内にカスタマイズしたツールボックスやツールセットを保存してください。SAS では、最初に Sasuser.Profile カタログ内でツールボックスとツールセットが検索され、次にアプリケーションカタログ内で検索されます。

SAS/AF アプリケーションの開発者あるいはサイト管理者であり、ウィンドウ特有またはアプリケーション特有のツールボックスを、すべてのユーザーがアクセスできるように編集している場合は、ウィンドウまたはアプリケーションの PMENU エントリと同じライブラリ名、カタログ名、エントリ名を使って TOOLBOX エントリを保存してください。ツールセットを特定のツールボックスと関連付けるには、TOOLBOX エントリと同じライブラリ名、カタログ名、エントリ名で TOOLSET エントリを保存します。適切な場所への書き込み許可が必要になります。たとえば、グラフィックエディタのカスタマイズしたツ

ールボックスを保存するには、サイト管理者は SASHELP.GI.GEDIT.TOOLBOX 内にツールボックスを保存する必要があります。

Save ボタンをクリックすると、ツールボックスの情報が **Name** フィールドで示されたカタログエントリへと保存されます。**Save As** ボタンをクリックすると、異なるライブラリ名、カタログ名、エントリ名を入力するように要求されます。ツールセットとしてツールボックスを保存することも選べます。ツールボックスをツールセットとして保存する場合、エントリの種類は TOOLSET になります。そうでない場合、エントリの種類は常に TOOLBOX になります。(ツールのセットを TOOLSET として保存することでは、ユーザーの TOOLBOX エントリは変更されません。) ツールセットの詳細については、"UNIX 環境でツールボックスとツールセットをカスタマイズする" (183 ページ) および "アプリケーションまたはウィンドウ固有のツールセットの作成またはカスタマイズ" (190 ページ)を参照してください。

最初に変更を保存せずに、**Close** ボタンまたは **Open** ボタンをクリックすると、作業を続ける前に、**Tool Editor** により変更を現在のツールボックスまたはツールセットに保存するように要求されます。

ツールボックスまたはツールセットを保存後は、**Tool Editor** はさらなる編集に備えて開いたままになり、**Name** フィールドは(新しい名前を入力していれば)新しいエントリの名前に変わります。

新しいツールボックスの作成

完全に新規のツールボックスを作成するには、次の方法から選んでください。

- **Tool Editor** を使用して既存のツールボックスを編集し、"ツールボックスまたはツールセットの変更の保存" (188 ページ)で記述されているように **Save as** ボタンをクリックしてツールボックスを保存します。
- **Explorer** ウィンドウの Sasuser.Profile カタログを開き、**File** ⇨ **New** ⇨ **Toolbox** を選択して新しいツールボックスを追加します。

アプリケーションまたはウィンドウ固有のツールボックスの作成またはカスタマイズ

アプリケーションの開発者で、既存のアプリケーションツールボックスを作成または編集する場合、次の手順に従います。

1. カスタマイズするウィンドウまたはアプリケーションの Sasuser.Profile 内にある既存の TOOLBOX をすべて削除します。

   Sasuser.Profile 内のツールボックスのコピーを削除すると、Tool Editor の呼び出し時に SAS に供給されるツールボックスのコピーを得られます。

2. "新しいツールボックスの作成" (189 ページ) または "Tool Editor の使用" (185 ページ)で説明されているように、アプリケーションツールボックスを作成または編集します。
3. "ツールボックスまたはツールセットの変更の保存" (188 ページ)で記述されているように、編集したツールボックスを保存します。
4. ユーザーに、ウィンドウツールボックスまたはアプリケーションツールボックスを変更したことを知らせます。

   ユーザーが新規のツールボックスを使用する場合、対応する TOOLBOX エントリを Sasuser.Profile から削除する必要があります。その後では、ウィンドウまたはアプリケーションの起動時に新しいツールボックスが自動的にロードされます。対応する TOOLBOX エントリを Sasuser.Profile から削除しない場合、新しいツールボックスではなく、対応するツールボックスのコピーがロードされます。

TOOLLOAD コマンドおよび TOOLCLOSE コマンドは、SAS/AF アプリケーションの開発時にとても役立ちます。これらのコマンドと EXECCMDI ルーチンを使用すると、開発するアプリケーションがツールボックスを開いたり閉じたりができるようになり、また、アプリケーションのユーザーに作業中において複数のツールボックスへのアクセス権限を与えることができます。EXECCMDI ルーチンの詳細については、*SAS Component Language: Reference* を参照してください。

アプリケーションまたはウィンドウ固有のツールセットの作成またはカスタマイズ

アプリケーション特有またはウィンドウ特有のツールボックスを作成するのと同じ方法で、アプリケーション特有またはウィンドウ特有のツールセットを定義します。これには、2 つの相違点があります。

- 新しいツールセットを作成するには、“新しいツールボックスの作成” (189 ページ) で記述されているようにツールボックスを定義することで開始します。
- ツールボックスを定義した後は、TOOLBOX エントリとしてではなく、TOOLSET エントリとして保存してください。

注: アプリケーションの開発者の場合、開発するアプリケーションを修正する前に、“アプリケーションまたはウィンドウ固有のツールボックスの作成またはカスタマイズ” (189 ページ) で記述されているように、アプリケーション用の既存の TOOLSET エントリを、必ずすべて削除してください。

UNIX 環境でキー定義をカスタマイズする

キー定義のカスタマイズ方法

キー定義をカスタマイズするには次の 4 つの方法があります。

- **Keys** ウィンドウを使用

  Keys ウィンドウを開くには、KEYS コマンドを発行するか、**Tools** ⇨ **Options** ⇨ **Keys** を選択します。プライマリ SAS ウィンドウ環境ウィンドウに **Keys** ウィンドウを使用してキー定義を変更する場合、キー定義はエントリ DMKEYS.KEYS 内の Sasuser.Profile カタログに保存されます。他の SAS ウィンドウのキー定義は、BUILD.KEYS、FSEDIT.KEYS などといった名前のカタログエントリ内に保存されます。

  KEYS コマンドと **Keys** ウィンドウについての詳細は、オンライン SAS ヘルプとドキュメントを参照してください。

- KEYDEF コマンドを使用

  KEYDEF コマンドにより、個々のファンクションキーの再定義ができます。

  ```
  keydef keyname <command|~text-string>
  ```

  たとえば、**`keydef F8 dlgpref`** を指定すると、**F8** キーで **Preferences** ダイアログボックスが開きます。

  KEYDEF コマンドについての詳細は、オンライン SAS ヘルプとドキュメント内の Base SAS セクションを参照してください。

- Resource Helper (reshelper)を使用

Resource Helper により、選択するキーと機能を基にして、SAS リソースの仕様が生成されます。Resource Helper を使用して、**Keys** ウィンドウ内にリストアップされる、任意のキーの機能を変更できます。Resource Helper の詳細については、"Resource Helper を使用し、X リソースを設定する" (178 ページ) および "Resource Helper を使用してキーの定義" (178 ページ)を参照してください。

ほとんどの場合で、Resource Helper はユーザー自身でリソースを定義するよりも簡単かつ高速です。ただし、X Window System はさまざまな場所のリソースを検索するため、ユーザーが定義しようとするキーに対して、Resource Helper が間違ったキー符号を取得する可能性もあります。また、キーに割り当てる操作ルーチンが **sas-function-key** ルーチンではない場合、**Keys** ウィンドウでキーラベルを変更する方法が Resource Helper にはありません。どちらの場合においても、キーリソースの定義を独自に行う必要があります。

- ユーザーのリソースファイル内に **SAS.keyboardTranslations** リソースおよび **SAS.keysWindowLabels** リソースを定義("キー変換の定義" (191 ページ)に説明されています)

キーボード上のほとんどのキーを定義可能です。ただし、キーの中には、関連付けられた専用の機能を持つものが少しながらあります。たとえば、マウスのボタンはカーソル操作とカットアンドペースト操作専用であり、ユーザーはカスタマイズできません。

キー変換の定義

キー変換について

X Window System のキーカスタマイズは、キーシーケンスおよび、そのキーシーケンスがキーボード上で入力されるときに実行される操作からなります。このカスタマイズは、操作へのキーのバインドとして知られます。同時に、キーの操作は変換として参照されます。

SAS.keyboardTranslations リソースについて

SAS.keyboardTranslations リソースには、すべての SAS ウィンドウで SAS が使用するキーバインドのセットを指定します。**SAS.keyboardTranslations** リソースのデフォルトの値は、ディスプレイとして使用している X サーバーによってレポートされる、ベンダの識別文字列を基にした実行時間で決定されます。これらのデフォルト値は、**!SASROOT/X11/resource_files** に含まれるファイル内にリストアップされています。SAS 提供によるデフォルトのバインドを変更するには、**SAS.keyboardTranslations** リソースを変更する必要があります。

注: このリソースで指定される X Toolkit Intrinsics 変換は、ユーザーの領域および、このリソースに影響されるすべての SAS ウィンドウのコマンド行の両方に適用されます。このリソースは、**Command** ウィンドウ、**Open** ダイアログボックスや **Import** ダイアログボックス、その他のドロップダウンメニューダイアログボックスのような Motif インターフェイスリソースに制御されるウィンドウには影響を与えません。

キー定義の作成ステップ

キー定義を作成するには、次の手順に従います。

1. 定義するキーの keysyms を決定します。

   Keysyms とは、キーボード上の各キーに X Window System によって認識される記号のことです。詳細については、"Keysyms の指定" (192 ページ)を参照してください。

2. リソースファイル内の **SAS.keyboardTranslations** リソースを変更または追加して、定義するキーの定義の含めます。

キーボードの動作ルーチンを使い、どの動作をキーが実行するかを定義します。**Keys** ウィンドウの右側の列にある定義では、**sas-function-key** 以外のキーボード動作ルーチンを使用して定義されるキーの機能を現在は制御していません。**Keys** ウィンドウ内のこれらのキーの定義は、無効なラベルになります。詳細については、"SAS.keyboardTranslations リソースの構文" (193 ページ)を参照してください。

3. リソースファイル内の **SAS.keysWindowLabels** リソースを変更または追加します。

   SAS.keysWindowLabels リソースでは、**Keys** ウィンドウに表示される有効なラベルのセットを指定します。新しいラベルを追加するか、**Keys** ウィンドウの左列にある既存のラベルを変更する場合のみ、このリソースを変更してください。

   SAS.keysWindowLabels リソースでは、**Keys** ウィンドウで使用されるニーモニックのみを定義します。特定のキーに操作を実行させるには、そのキーに **SAS.keyboardTranslations** 定義を指定する必要があります。詳細については、"SAS.keysWindowLabels リソースの構文" (194 ページ)を参照してください。

4. SAS セッションを開始して、**Keys** ウィンドウを開きます。

5. **Keys** ウィンドウの右の列で、コマンド名または定義した各キーの他の詳細を入力します。

キー定義の例については、"例 : SAS リソースを使用してキーの定義" (197 ページ)を参照してください。

Keysyms の指定

xev ユーティリティを使用して、キーボード上のキーに関連付けられた keysyms を指定できます。**xev** ユーティリティは、ほとんどの UNIX オペレーティングシステムと一緒に配布されていますが、使用している動作環境に **xev** がインストールされていない場合は、UNIX システム管理者に連絡し、その UNIX 環境で使用可能な別の方法についての情報を入手してください。**xev** ユーティリティは、発生する各 X イベントのメッセージを書き込みます。**KeyPress** イベントでは、押下された各キーに keysym が指定されます。

キーを定義するには、次の手順に従います。

1. キーを定義する X サーバー上で、**xev** ユーティリティを開始します。

   xev クライアントでは、小さい **Event Tester** ウィンドウが表示され、発生する X イベントがリストされます。(**xev** クライアントは大量の出力を生成するので、後で検討するために、その出力をファイルに保存する必要のある場合があります。UNIX の **script** コマンドを発行して、ファイルへと出力を保存できます。)

2. 必要であれば、**Event Tester** ウィンドウ上でマウスポインタをクリックして、そのウィンドウにキーボードフォーカスを与えます。

3. 定義するキーを押して、リストされる **KeyPress** イベントに注目します。

   その一覧には、カンマで区切られた多くの項目が含まれます。**KeyPress** イベント内のフィールドの 1 つには、押下されたキーに関連付けられた keysym の名前が表示されます。

   たとえば、NumLock をオンにして、Dell PC の 105 キーボードのキーパッド上の 0 キーを押すと、次の出力が生成されます。

```
KeyPress event, serial 32, synthetic NO,
window 0x1a00001, root 0x5d, subw 0x1a00002,
time 600120687, (37,41), root:(240,458),
state 0x10, keycode 90 (keysym 0xffb0, KP_0),
```

```
same_screen YES,
XLookupString gives 1 bytes: (30) "0"
XmbLookupString gives 1 bytes: (30) "0"
XFilterEvent returns: False
```

この例では、keysym の名前は **KP_0** です。

注: **SAS.defaultVirtualBindings** リソースで、仮想 keysyms のセットが定義されます。仮想 keysyms はすべて、**osfPageDown**、**osfClear**、**osfPrimaryPaste** などのように、**osf** で始まっています。SAS が提供する初期設定を使わずに、これらの仮想バインドを再配置する場合、予期せぬ結果になる可能性があります。指定したキー変換が機能しない場合、仮想 keysym にバインドされるキーを再定義する必要がある可能性があります。この場合、**xev** ユーティリティが表示する keysym のかわりに、**SAS.keyboardTranslations** リソースに仮想 keysym を指定する必要があります。Resource Helper を開始し、**Keys** をクリックし、そして定義するキーまたはキーの組み合わせを押すと、キーにバインドされる仮想 keysym を指定できます。Resource Helper は仮想 keysym の名前を表示します。SAS がインストールされているディレクトリ(**!SASROOT**)内の**/X11/resource_files** にあるキー定義ファイル、および VirtualBinding または xmbind の UNIX man ページを参照してください。

SAS.keyboardTranslations リソースの構文

注: ほとんどの SAS ドキュメントでは、山かっこ(<>)が使われて任意の構文が表示されます。ただし、このトピックでは、任意の構文は角かっこ([])で表されます。このトピック内での山かっこは、構文の一部であり、表示されるとおりに入力する必要があります。

ここに **SAS.keyboardTranslations** リソースの構文を示します。

SAS.keyboardTranslations: #override \

[*modifier*] <Key>*keysym* : *action-routine* \n\

[*modifier*] <Key>*keysym* : *action-routine*

#override
この定義が、他のキーとは関係なく定義される特定のキーの既存のバインドよりも優先されることを示します。*#override* ディレクティブを省略する場合、新しいバインドはデフォルトのバインドすべてを置き換え、キーボード上の他のキーはすべて利用できなくなります。

注: *#augment* および*#replace* ディレクティブの詳細については、X Window System のドキュメントを参照してください。

modifier
次のいずれかになります。

- Alt
- Ctrl
- Meta
- Shift
- Lock
- Mod1
- Mod2
- Mod3
- Mod4

- Mod5
- None
- blank space

有効な修飾子のリストは、使用するキーボードにより変わります。使用するキーボードに有効な修飾子のリストを表示するには、**xmodmap** UNIX コマンドを入力します。詳細については、**xmodmap** の UNIX man ページを参照してください。

<Key>
必要とされます。keysym の先頭を示します。

keysym
定義するキーに X により認識されるキーの記号です。詳細については、"Keysyms の指定" (192 ページ)を参照してください。

action-routine
キーに行わせる操作です。"SAS キーボード操作名" (194 ページ)で記述されている中の任意の操作ルーチンを指定できます。

\n
X 変換マネージャにより、変換シーケンスが終了して次のシーケンスが開始する場所を指定できます。最後の変換終了後には、\n を入力しません。

\
行の終わりの改行文字が、定義の一部として解釈されないようにします。この文字の使用は文体の慣例であり、各変換を別の行に表示されます。最後の変換終了後には、バックスラッシュを入力しません。

注: SAS により、**SAS.keyboardTranslations** リソース内の無効なキーの指定を阻害されることはありません。場合によっては、無効なキーにより、シェルウィンドウに警告が出されます。

SAS.keysWindowLabels リソースの構文

注: 次の構文内の角かっこ([])は、*(InternalKeyName)*が任意であることを示します。

ここに **SAS.keysWindowLabels** リソースの構文を示します。

SAS.keyWindowLabels: \
KeyWindowLabel [(*InternalKeyName*)] \n\
KeyWindowLabel [(*InternalKeyName*)]

KeyWindowLabel
Keys ウィンドウ内に表示させるラベル(1 から 8 文字まで)です。

InternalKeyName
対応する **SAS.keyboardTranslations** キーバインドの **sas-function-key** 動作ルーチンに渡される文字列です。(*InternalKeyName* は SAS に使用され、**Keys** ウィンドウエントリを、SAS カタログからロードされたまたは SAS**Keys** ウィンドウで定義された KEYS モジュール内のキー定義と関連付けます。)
InternalKeyName が指定されていない場合、SAS により、*KeyWindowLabel* は *InternalKeyName* として使用されます。

\n および \
SAS.keyboardTranslations リソース内と同じ目的で使用します。詳細については、"SAS.keyboardTranslations リソースの構文" (193 ページ)を参照してください。

SAS キーボード操作名

注: ほとんどの SAS ドキュメントでは、山かっこ(<>)が使われて任意の構文が表示されます。ただし、このトピックでは、任意の構文は角かっこ([])で表されます。このトピック内での山かっこは、構文の一部であり、表示されるとおりに入力する必要があります。

SAS では、X の初期化中にキーボード操作のセットが示されます。これらのキーボード操作を単純な機能として見なすこともできます。操作が実行される場合、現在キーボード入力フォーカスを持つウィンドウ上で行われます。

次のキーボード操作のリストは、Motif インターフェイスによって X ツールキットキーボードのイベント変換での使用に登録された操作ルーチンを示します。

sas-cursor-down()
SAS ウィンドウ内でカーソルを 1 行下げます。カーソルは、SAS ウィンドウ内の最下部に達しても、元の位置には戻りません。

sas-cursor-left()
SAS ウィンドウ内でカーソルを 1 文字分左に動かします。カーソルは、SAS ウィンドウ内の左側に達しても、元の位置には戻りません。

sas-cursor-right()
SAS ウィンドウ内でカーソルを 1 文字分右 i に動かします。カーソルは、SAS ウィンドウ内の右側に達しても、元の位置には戻りません。

sas-cursor-up()
SAS ウィンドウ内でカーソルを 1 行上げます。カーソルは、SAS ウィンドウ内の最上部に達しても、元の位置には戻りません。

sas-delete()
現在のフィールド内のすべてのテキストを削除します。

sas-delete-begin()
現在のカーソル位置から現在のテキストフィールドの先頭までのテキストを削除します。

sas-delete-char()
テキストカーソルの下の文字を削除し、カーソルを所定の位置に残します。

sas-delete-end()
現在のカーソル位置から現在のテキストフィールドの末尾までのテキストを削除します。

sas-delete-prev-chr()
テキストカーソルの左へと文字を削除し、カーソルを 1 空白分戻します。

sas-delete-prev-word()
現在のカーソル位置から直前の単語の先頭までのテキストを削除します。その操作が行われるときにカーソルが単語の内部にある場合、カーソルの位置からその単語の先頭までのテキストが削除されます。

sas-delete-word()
現在のカーソル位置から現在または次の単語までのテキストを削除します。

sas-do-command()
操作が行われるとき、SAS コマンドとして解釈される 1 つ以上のテキスト文字列パラメータが実行されるのを認めます。操作は複数のパラメータで行われる場合があります。それらのパラメータは、パラメータ間に **sas-do-command** 操作が提供するセミコロンの区切り文字で連結されます。連結した SAS コマンド文字列は、それからサブミットされ実行されます。たとえば、次の変換構文を使用して、すべての SAS ウィンドウ環境ウィンドウに HOME と SUBMIT キーシーケンスを定義できます。

```
<Key>KP_F3: sas-do-command(HOME;SUBMIT)
```

sas-function-key("InternalKeyName")
InternalKeyName ラベルに識別されたファンクションキーに関連する SAS コマンドを呼び出します。*InternalKeyName* は、keysWindowLabels リソースに渡される、(1 から 8 文字までの長さの)文字列です。*InternalKeyName* は引用符で囲んでください。内部キーの名前についての詳細は、“キー変換の定義” (191 ページ)を参照してください。

sas-home-cursor()
HOME コマンドの同等物です。HOME 操作が、すべての SAS ウィンドウ環境ウィンドウへ定義されるように、便宜上提供されます。

sas-insert-char(["InsertionString"])
テキストカーソルの下の入力フィールドへと入力された文字を挿入または上書きします。挿入あるいは上書きの動作は、**sas-toggle-insert** 操作によって決まります。その操作には、表示されるテキストカーソルのスタイルが反映されるモードがあります。ブロックカーソルは、上書きモードであることを示し、下線カーソルは、挿入モードであることを示します。通常、**sas-insert-char** は、XKeyEvent を適切な文字に変換し、その文字を SAS テキストカーソルのある場所に挿入します。パラメータを指定する場合、このパラメータが表すテキスト文字列は、SAS テキストカーソルの場所に挿入されます。文字列を二重引用符で囲まないと、文字列の中の余白は、X ツールキットによって、パラメータ区切り文字として解釈されます。文字列パラメータへの引用符の埋め込みについては、X Window System ドキュメントを参照してください。逸脱した引用符を含めるには、次の構文を使用します。

```
Shift<Key>KP_1: sas-insert-char("One\\"1\\"")
```

この構文では、SAS テキストカーソルの位置に、テキスト文字列 **One"1"**が作成されます。

sas-kp-application()
ワークステーションの数字キーパッドを設定して、ファンクションキーの変換を元に戻します。この操作は、**sas-function-key()** 操作にバインドされたキーパッドのキーにのみ有効です。その他の操作へバインドしているキーパッドは、この変換の影響を受けません。

sas-kp-numeric()
ワークステーションのキーパッドが、以前のファンクションキーの割り当てではなく、数値文字を出力するように設定します。この操作は、**sas-function-key()** 操作にバインドされたキーパッドのキーにのみ有効です。その他の操作へバインドしているキーパッドは、この変換の影響を受けません。

sas-move-begin()
カーソルを現在のテキストフィールドの先頭に移動します。

sas-move-end()
カーソルを現在のテキストフィールドの末尾に移動します。

sas-new-line()
呼び出し時に、行末イベントを発生します。この操作は、コンテキストに依存します。SAS コマンド行でこの操作を入力すると、入力されたテキストはサブミットされて実行されます。SAS アプリケーションのクライアント領域で呼び出すと、この操作はテキストカーソル下のテキストエリアの属性によって変化します。簡単に言えば、この操作は入力フィールドの標準的な行ターミネータです。

sas-next-field()
SAS アプリケーションを、SAS ウィンドウのクライアント領域の次のフィールドへと進めます。

sas-next-word()
テキストカーソルを、現在のテキストフィールドの次の単語の先頭まで進めます。**sas-next-word** によって、現在のテキストフィールド内の単語の先頭が検出され

ない場合、次の SAS アプリケーションフィールドまで進めます。ウィンドウの SAS コマンド行領域に入力している場合は、カーソルが SAS ウィンドウクライアント領域にまで折り返すことはありません。

sas-page-down()
現在のウィンドウの内容を 1 ページ単位で下にスクロールします。

sas-page-end()
テキストカーソルを現在のページの末尾に移動します。

sas-page-top()
テキストカーソルを現在のページの先頭に移動します。

sas-page-up()
現在のウィンドウの内容を 1 ページ単位で上にスクロールします。

sas-prev-field()
SAS アプリケーションを、SAS ウィンドウのクライアント領域の前のフィールドへと戻します。

sas-prev-word()
テキストカーソルを、現在のテキストフィールドの前の単語の先頭まで戻します。**sas-prev-word** によって、現在のテキストフィールド内の前の単語の先頭が検出されない場合、前の SAS アプリケーションフィールドまで戻します。ウィンドウの SAS コマンド行領域に入力している場合は、カーソルが SAS ウィンドウクライアント領域にまで折り返すことはありません。

sas-to-bottom()
テキストカーソルをウィンドウのテキスト範囲の最下段まで移動します。

sas-to-top()
テキストカーソルをウィンドウのテキスト範囲の最上段まで移動します。

sas-toggle-insert()
関連するウィンドウの行編集の動作を、挿入モードと上書きモード間で切り替えます。この切り替えは、SAS コマンド行と SAS ウィンドウクライアント領域にのみ適用されます。現在のモードは、使用中のカーソルのスタイルで判別できます。ブロックカーソルは上書きモードであることを示し、下線カーソルは挿入モードであることを示します。

sas-xattr-key(<KeyType>[,<KeyParam>])
SAS の拡張属性キーを処理します。*KeyType* パラメータの値は、XACOLOR、XAATTR または XACLEAR である必要があります。*KeyType* XACOLOR では、12 の DMS の色の名前が有効なパラメータです。*KeyType* XAATTR では、有効な値は HIGHLIGHT、REVERSE、BLINK および UNDERLINE です。XACLEAR にはパラメータは必要ありません。Motif インターフェイスでは、BLINK 属性はサポートされません。ただし、BLINK 属性が指定されている場合、カタログが他の動作環境へポートされるときに表示されます。

例 : SAS リソースを使用してキーの定義

注: ほとんどの SAS ドキュメントでは、山かっこ(<>)が使われて任意の構文が表示されます。ただし、この例では、任意の構文は角かっこ([])で表されます。この例での山かっこは、構文の一部であり、表示されるとおりに入力する必要があります。

次の例では、**sas-do-command** 操作ルーチンにより、COMMAND コマンドが **KP_0** のすべての既存の定義よりも優先されることが指定されます。

```
SAS.keyboardTranslations: #override \n\
None<Key>KP_0: sas-do-command(COMMAND)
```

他のすべてのキーは現在の定義を保持します。

次の例では、キーシーケンス CTRL-K が KEYS コマンドにバインドされ、CTRL-D はカーソル下の文字を削除することを指定しています。CTRL-K と CTRL-D の **Keys** ウィンドウに入力されたコマンドは有効になりません。

```
SAS.keyboardTranslations: #override\
Ctrl<Key>k: sas-do-command(keys)\n\
Ctrl<Key>d: sas-delete-char()
```

次の例では、keysym**hpClearLine** に関連付けられたキーが、**Keys** ウィンドウの **MyClrLn** ラベルの隣に入力されたコマンドを実行することを指定しています。

```
SAS.keyboardTranslations: #override \
<Key>hpClearLine : sas-function-key("ClearLn")
SAS.keysWindowLabels: MyClrLn(ClearLn)
```

SAS.keysWindowLabels リソースのかっこ内に表示される文字列は、**sas-function-key** ルーチンへパラメータとして入力される文字列と一致する必要があります。ラベル (**MyClrLn**)は、任意の文字列でなり得ます。そして、keysym **hpClearLine** は、使用するキーボードに有効な keysym である必要があります。

UNIX 環境でフォントをカスタマイズする

システムフォントとウィンドウ環境で使用されるフォントの相違点

SAS では、主に 2 つの種類のフォントが使用されます。

- システムフォントが、ほとんどのダイアログボックスとメニューで使用されます。CDE ***.systemFont** リソースで定義されたシステムフォントが継承されます。このリソースが定義されない場合、Helvetica フォントが使用されます。
- DMS フォントは SAS ウィンドウで使用されます。SAS のフォントを変更するには、**Fonts** ダイアログボックスを使うか、またはリソースファイル内のリソースを指定します。フォントは、固定フォントまたは固定スペースフォントである必要があります。

注: アプリケーションの起動前にフォントを変更するのが最善です。アプリケーション実行中にフォントを変更すると、予期せぬ動作を起こすことがあります。

SAS による使用フォントの決定法

SAS では、標準(太字ではない)のデフォルトのフォントを次のように指定します。

1. **Font** ダイアログボックスを使って SASUSER.PROFILE.DMSFONT.UNXPREFS にフォントを保存していた場合、このフォントはデフォルトの通常フォントとして使用されます。
2. **Font** ダイアログボックスを使って、フォントを保存していなかったが、**SAS.DMSFont** リソースを設定していた場合、このリソースで指定されたフォントがデフォルトのフォントとして使用されます。
3. **SAS.DMSFont** リソースを設定していなかった場合、パターン*Font に一致する任意のフォントが使用されます。このパターンは定義または継承されることがあります。
4. *Font に一致するリソースを指定も継承もしていないが、**SAS.DMSFontPattern** リソースを設定していた場合、このリソースを使って、どのフォントが使用されるかが決まります。*Font に一致する任意のリソースが継承または定義される場合、**SAS.DMSfontPattern** リソースは有効ではありません。

5. どのリソースも設定されていなかった場合、SAS によって、X サーバーで利用可能なフォントからフォントが選択されます。

SAS.DMSboldFont リソースに値を指定していなかった場合、デフォルトの標準フォントを使って、デフォルトの太字フォントが指定されます。標準 **SAS.DMSFont** に関連付けられた XLFD 名がある場合、SAS によって一致する太字フォントが選択され、ロードされます。SAS が自動的に太字フォントを選択/ロードできない場合、その太字フォントにも標準フォントが使用されます。

多くの場合、フォント名はエイリアスを与えられ、より短い名前で、関連する XLFD 名を持つフォントを参照できるようになります。太字フォントを指定するときに使用される名前は、標準フォントの XA_FONT フォントプロパティを基にしています。

Fonts ダイアログボックスを使用したフォントのカスタマイズ

Fonts ダイアログボックスについて

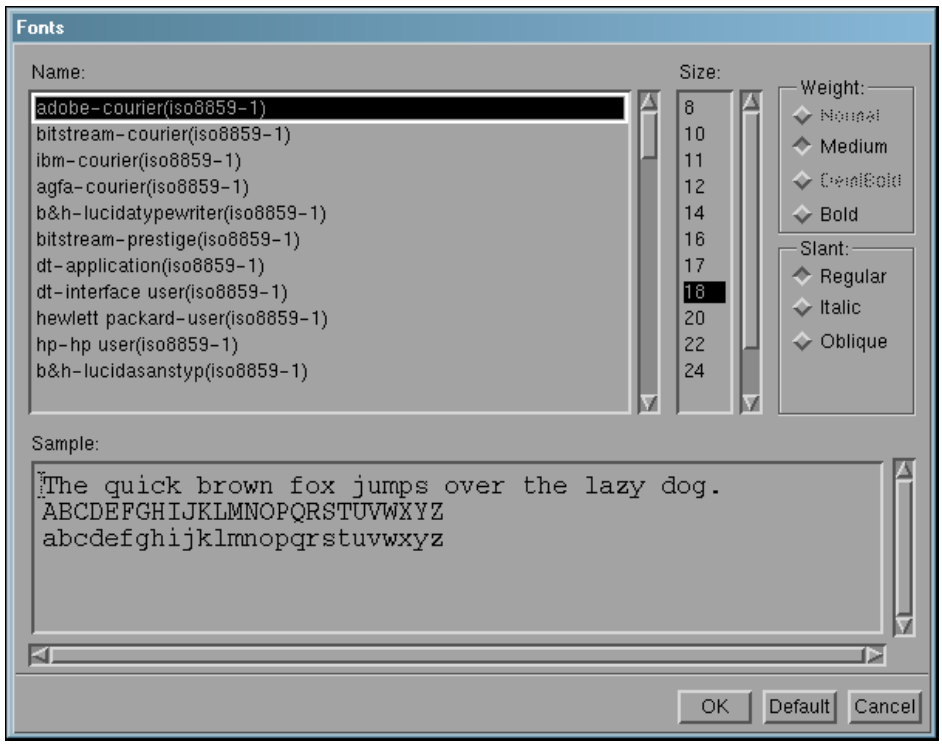
Fonts ダイアログボックスにより、SAS セッション全体のウィンドウ環境フォントを変更できます。フォントを変更する場合、選択するフォントは SASUSER.PROFILE.DMSFONT.UNXPREFS に保存され、将来の SAS セッションで使用されます。

デフォルトフォントの変更法

デフォルトのフォントを変更するには、**Fonts** ダイアログボックスを開きます。**Fonts** ダイアログボックスを開くには、次の方法の中から 1 つを使います。

- コマンドウィンドウで DLGFONT コマンドを実行します。
- **Tools** ⇨ **Options** ⇨ **Fonts** を選択します。

画面 9.8 *Fonts ダイアログボックス*



- フォント名を選び、必要に応じて、サイズ、重さ、傾きも選びます。(すべてのフォントが、すべてのサイズ、重さ、傾きで利用できません。) **Sample** フィールドは、選択したフォントがどのように見えるかを示します。
- **OK** をクリックして、既存のフォントを選択したフォントに変更します。

デフォルトのフォントに戻すには、**Default** をクリックします。

変更を取り消し、**Fonts** ダイアログボックスを終了するには、**Cancel** をクリックします。

フォントのリソースの指定

SAS ウィンドウ環境で使用されるフォントを次のリソースを使ってカスタマイズできます。

SAS.DMSFont: font-name
デフォルトの通常フォントとして使用されるフォントを指定します。デフォルトの通常フォントは Courier です。

SAS.DMSboldFont: font-name
デフォルトの太字フォントとして使用されるフォントを指定します。

SAS.DMSDBfont: font-name
マルチバイト文字セットをサポートする動作環境の SAS ウィンドウシステムに使用される、マルチバイトの通常文字セットフォントを指定します。

SAS.DMSDBboldFont: font-name
マルチバイト文字セットをサポートする動作環境の SAS ウィンドウシステムに使用される、マルチバイトの太字文字セットフォントを指定します。

SAS.DMSfontPattern: XLFD-pattern
使用する XLFD (X Logical Font Description)パターンを指定し、ウィンドウ環境フォントを決定します。X Window System 内のフォントのほとんどは、ハイフン(–)文字で区切られる多くの異なるフィールドを含む XLFD と関連しています。XLFD 内のフィールドにより、フォントファミリ名、太さ、サイズ、解像度といったプロパティと、そのフォントがプロポーショナルなのか等幅なのかがわかります。X で使用する XLFD およびフォント名についての詳細は、X Window System ドキュメントを参照してください。

SAS.DMSfontPattern に指定する *XLFD-pattern* は、XLFD と同じ数のフィールドを含む必要があります。アスタリスク(*)文字は、その特定のフィールドに対して、あらゆる値が許容できることを意味します。たとえば、次のパターンは、一般的な傾きを持ち、太字ではなく、等幅かつ iso8859 であるフォントに一致します。

```
SAS.DMSFontPattern: -*-*-*-r-*--*-*-*-*-m-*-iso8859-1
```

SAS により、*XLFD-pattern* を使って次のようにフォントが選ばれます。

- SAS は、**SAS.DMSfontPattern** リソースに一致するフォントのリストを X サーバーに照会します。
- 現在の X ディスプレイとは異なる X と Y 解像度を持つすべてのフォント、(プロポーショナルフォントのような)変数文字セルサイジングを持つすべてのフォントおよび、8 ポイント未満または 16 ポイント以上のポイントサイズを持つすべてのフォントは、SAS によって除外されます。この手順で結果的に空白のリストができる場合、SAS により一般的な(かつ通常は固定の)フォントが選ばれます。
- 残りのリストから、最大のポイントサイズを持つフォントが選ばれます。

SAS.fontPattern: XLFD-pattern
SAS のグラフィックフォントのリクエストを解決するために使用される、候補となるフォントを記述している XLFD フォントパターンを指定します。このパターンを使用すると、ユーザーはさまざまなグラフィックアプリケーションの状況で、X フォントの使

用を最適化または制御できます。*のデフォルト値 は通常、パフォーマンスに著しく影響を及ぼすことはありません。極端な数のフォントがあるまたは限定されたパフォーマンス環境で作動しているサーバー上で SAS を実行中の場合は、フォントの検索に制限をかけてください。

SAS.systemFont: font-name
システムフォントを指定します。SAS フォントは、SAS ウィンドウで使用されます。システムフォントが、ほとんどのダイアログボックスとメニューで使用されます。通常、システムフォントは、CDE (Common Desktop Environment)や KDE (K Desktop Environment)のような X Window 環境で設定されるフォントリソースから継承されます。***.systemFont** リソースの場合は、12 ポイントの Helvetica フォントが使用されます。

フォントのエイリアスの指定

フォントのエイリアス

使用するサーバーが、SAS が供給するすべてのフォントに一致するフォントを提供しない場合、フォントエイリアスのリソースを使用して、システムで利用可能なフォントを置き換えできます。(利用できるフォントについては、システム管理者に尋ねてください。) 次の構文を使い、リソースファイル内にあるフォントエイリアスを指定します。

SAS.*supplied-font*Alias: *substitute-family*

supplied-font は、SAS 提供のフォント名です。*substitute-family* は、置き換えるフォントのファミリ名です。

注意:
SAS フォントを、フォントエイリアスとして指定しないでください。 SAS が提供するフォントを、フォントエイリアスとして指定すると、次の例のように、不一致が起こる場合があります。

```
SAS.timesRomanAlias: symbol
```

この値をフォントエイリアスに割り当てると、フォント選択ダイアログボックスを使って、シンボルフォントを選択することができなくなります。このシンボルフォントは、Times Roman エイリアスとして指定されているからです。

次の表では、SAS フォントエイリアスのリソース名を示します。

表 9.1 *SAS フォントエイリアスのリソース*

リソース名	クラス名
SAS.timesRomanAlias	TimesRomanAlias
SAS.helveticaAlias	HelveticaAlias
SAS.courierAlias	CourierAlias
SAS.symbolAlias	SymbolAlias
SAS.avantGardeAlias	AvantGardeAlias
SAS.bookmanAlias	BookmanAlias

リソース名	クラス名
SAS.newCenturySchoolbookAlias	NewCenturySchoolbookAlias
SAS.palatinoAlias	PalatinoAlias
SAS.zapfChanceryAlias	ZapfChanceryAlias
SAS.zapfDingbatsAlias	ZapfDingbatsAlias

例 : Lucida フォントを Palatino のかわりに使用

システムが Palatino フォントを持たないが、次の Lucida フォントを持つと仮定します。

```
b&h-lucida-bold-r-normal-sans-
10-100-75-75-p-66-iso8859-1
```

Lucida フォントを Palatino フォントに置き換えるには、リソースファイル内に、次の行を含めてください。

```
SAS.palatinoAlias: lucida
```

UNIX 環境で色をカスタマイズする

SAS セッションの色の設定のカスタマイズ法

SAS では、すべての SAS ウィンドウの構成要素にデフォルトの色と属性設定のセットが提供されます。次の方法で、SAS セッションの色をカスタマイズできます。

- Resource Helper (reshelper)を使用します。

  Resource Helper を使用して、任意の色をカスタマイズできます。詳細については、“Resource Helper を使用し、X リソースを設定する” (178 ページ) および “Resource Helper を使用して SAS ウィンドウの色を変更” (180 ページ)を参照してください。

- “SASCOLOR ウィンドウを使用して色をカスタマイズする” (203 ページ)で記述されているように、**SASCOLOR** ウィンドウを使用します。

  SASCOLOR ウィンドウを使って、ほとんどの SAS ウィンドウのウィンドウ構成要素をカスタマイズできます。

- “COLOR コマンドの構文” (203 ページ)で記述されているように、COLOR コマンドを使用します。

  COLOR コマンドは、アクティブなウィンドウの指定された構成要素にのみ影響を与えます。COLOR コマンドを使っての変更は、ここで記述されている他のあらゆる方法で入力された変更よりも優先されます。

- 自身で色リソース指定を入力します。

  色を制御する X リソースに、特定の RGB 値または色の名前を入力できます。詳細については、“色のリソースの定義” (204 ページ)を参照してください。

SASCOLOR ウィンドウを使用して色をカスタマイズする

SASCOLOR ウィンドウを使用して、SAS ウィンドウの特定の構成要素の色と強調表示を変更できます。**SASCOLOR** ウィンドウを開くには、SASCOLOR コマンドを発行するか、**Tools** ⇨ **Options** ⇨ **Colors** を選択します。

画面 9.9 *SASCOLOR ウィンドウ*

ウィンドウ構成要素の色を変更するには、構成要素名を選択し、それから、その構成要素に割り当てる色と属性を選択します。

BLINK 属性はサポートされません。HIGHLIGHT 属性により、テキストが太字で表示されます。

Save をクリックすると、カタログエントリ SASUSER.PROFILE.SAS.CPARMS に変更が保存されます。

注: 新しい色の設定を有効にするには、すべてのアクティブなウィンドウを閉じて、もう一度開いてください。

SASCOLOR ウィンドウについての詳細は、オンライン SAS ヘルプとドキュメントを参照してください。

COLOR コマンドの構文

COLOR コマンドを使って、アクティブなウィンドウの特定の構成要素に色を設定できます。

```
color field-type <color|NEXT <highlight>>
```

field-type
背景、コマンド、境界線、メッセージなどといったウィンドウの領域を指定します。

color
青(B に短縮できます)、赤(R)、緑(G)、シアン(C)、ピンク(P)、黄(Y)、白(W)、黒(K)、マゼンタ(M)、グレー(A)、茶(B)、オレンジ(O)といった色を指定します。

NEXT
色を、次の使用可能な色に変更します。

highlight
H (テキストを太字フォントで表示)、U (下線)、R (色の反転)のどれかになります。BLINK 属性はサポートされません。

変更を保存するには、WSAVE コマンドを実行してください。
SASUSER.PROFILE.window.WSAVE へと変更は保存されます。

注: WSAVE コマンドは、すべての SAS ウィンドウで利用可能でない場合もあります。たとえば、SAS/FSP を使うと、変更は EDPARMS ウィンドウまたは **PARMS** ウィンドウのいずれかを使って保存されます。(WSAVE が、特定の SAS ウィンドウで使用可能かどうかを調べるには、製品ドキュメントを参照してください。)

COLOR コマンドと WSAVE コマンドについての詳細は、オンライン SAS ヘルプとドキュメントを参照してください。

色のリソースの定義

色のリソースの種類

色のリソースは、2 つのカテゴリに分類されます。

- 前景と背景の定義

  これらのリソースにより、12 の DMS の色を定義するのに使用される RGB の値をカスタマイズできます。それぞれの色は背景色または前景色として使用されることがあるため、それぞれの用途のそれぞれの色に、異なる RGB 値または色の名前を指定できます。たとえば、青が前景色として使用されるときは、#0046ED の使用を、青が背景色として使用されるときは CornflowerBlue の使用を指定できます。

- ウィンドウ構成要素の定義

  CPARMS リソースとして参照される、これらのリソースにより、各ウィンドウ構成要素に、12 の DMS の色からどれを使うか指定できます。たとえば、メッセージのテキストのマゼンタでの表示を指定できます。

これらの 2 種類のリソースは同時に機能します。CPARMS の色の値は、現在の前景と背景の定義を使用します。たとえば、次のリソースはプライマリウィンドウの背景を CornflowerBlue に指定しています。

```
SAS.blueBackgroundColor: CornflowerBlue
SAS.cparmBackground: DmBlue
```

前景リソースと背景リソースの RGB 値と色名の指定

SAS では、**SAS.systemBackground**、**SAS.systemForeground**、および次の表にリストしたリソースを使用して、SAS ウィンドウに使用する色を決定します。

SAS.systemForeground: color
SASCOLOR ウィンドウ内の前景システム色の色を指定します。

SAS.systemBackground: color
SASCOLOR ウィンドウ内の背景システム色の色を指定します。

SAS.systemSecondaryBackground: color
システムのセカンダリ背景色を設定し、**SASCOLOR** ウィンドウ内のセカンダリ背景システム色の色を指定します。

MediumVioletRed のような色名または#0000FF のような RGB 値を、すべての前景および背景リソースに指定できます。RGB 色の値については、X Window System ドキュメントを参照してください。

次の表では、すべての前景色と背景色のリソースと、それらのクラス名を示します。これらのリソースのすべては、String の種類です。

表 9.2 前景色と背景色のリソース

リソース名	クラス名
SAS.systemForeground	SystemForeground
SAS.systemBackground	SystemBackground
SAS.systemSecondaryBackground	Background
SAS.blackForegroundColor	BlackForegroundColor
SAS.blueForegroundColor	BlueForegroundColor
SAS.brownForegroundColor	BrownForegroundColor
SAS.cyanForegroundColor	CyanForegroundColor
SAS.grayForegroundColor	GrayForegroundColor
SAS.greenForegroundColor	GreenForegroundColor
SAS.magentaForegroundColor	MagentaForegroundColor
SAS.orangeForegroundColor	OrangeForegroundColor
SAS.pinkForegroundColor	PinkForegroundColor
SAS.redForegroundColor	RedForegroundColor
SAS.whiteForegroundColor	WhiteForegroundColor
SAS.yellowForegroundColor	YellowForegroundColor
SAS.blackBackgroundColor	BlackBackgroundColor
SAS.blueBackgroundColor	BlueBackgroundColor
SAS.brownBackgroundColor	BrownBackgroundColor
SAS.cyanBackgroundColor	CyanBackgroundColor
SAS.grayBackgroundColor	GrayBackgroundColor
SAS.greenBackgroundColor	GreenBackgroundColor
SAS.magentaBackgroundColor	MagentaBackgroundColor
SAS.orangeBackgroundColor	OrangeBackgroundColor
SAS.pinkBackgroundColor	PinkBackgroundColor

リソース名	クラス名
SAS.redBackgroundColor	RedBackgroundColor
SAS.whiteBackgroundColor	WhiteBackgroundColor
SAS.yellowBackgroundColor	YellowBackgroundColor

ウィンドウ要素の色と属性の定義(CPARMS)

特定のウィンドウ構成要素に色と属性を定義するには、CPARMS という SAS リソースに値を割り当てます。各 CPARMS のリソースにより、セカンダリウィンドウ内の背景やプライマリウィンドウの境界線などの特定のウィンドウ構成要素の色と属性が指定されます。

同一のリソース定義内に複数の色と属性の名前を指定できますが、最後に定義された色と属性のみが使用されます。

```
SAS.cparmResource: DmColorName|DmAttrName\
<+DmColorName|DmAttrName>
```

Resource は、次の表内の CPARMS リソースのいずれかになります。これらのリソースすべては、DmColor タイプで、そのデフォルト値は動的です。つまり、デフォルトの値は実行時に決定されます。

表 9.3 *SAS CPARMS リソース*

リソース名	色と属性の設定	クラス名	デフォルトの色
SAS.cparmBackground	SAS セッションで表示されるすべてのプライマリウィンドウ内の背景。	CparmBackground	DmWhite
SAS.cparmBanner	ウィンドウ内のバナー。	CparmForeground	DmBlack
SAS.cparmBorder	プライマリウィンドウの境界線。	CparmBackground	DmBlack
SAS.cparmByline	**Output** ウィンドウに書き込まれる BY 行。	CparmForeground	DmBlue
SAS.cparmColumn	列情報のテキストラベル。SAS エディタ内で、このリソースを使い、編集中の行を識別でき、また、スプレッドシートウィンドウ内で、スプレッドシートにラベルを付けられます。	CparmForeground	DmBlue/ Underline
SAS.cparmCommand	メニューが無効のときのコマンドデータエントリフィールド。	CparmForeground	DmBlack
SAS.cparmData	**Log** ウィンドウまたは **Output** ウィンドウへと書き込まれる一般的な行。	CparmForeground	DmBlack

リソース名	色と属性の設定	クラス名	デフォルトの色
SAS.cparmError	**Log** ウィンドウまたは **Output** ウィンドウへと書き込まれる ERROR の行。	CparmForeground	DmRed
SAS.cparmFootnote	**Output** ウィンドウに書き込まれる FOOTNOTES 行。	CparmForeground	DmBlue
SAS.cparmForeground	編集可能な SAS ウィンドウ環境ウィンドウ内のすべてのテキストフィールド。	CparmBackground	DmBlack
SAS.cparmHeader	**Output** ウィンドウに書き込まれる HEADERS 行。	CparmForeground	DmBlue
SAS.cparmHelpLink	ヘルプシステム内の追加の情報レベルへのリンク。	CparmForeground	DmGreen/ Underline
SAS.cparmHelpMainTopic	ヘルプシステム内のトピックワードまたはフレーズ。	CparmForeground	DmBlack
SAS.cparmHelpSubTopic	ヘルプシステム内のトピックワードまたはフレーズ。	CparmForeground	DmBlack
SAS.cparmInfo	ユーザーへの補助としてウィンドウに表示されるテキスト。たとえば、次のようになります。 `Press Enter to continue`	CparmForeground	DmBlack
SAS.cparmLabel	ウィジェットの前に来るテキスト。たとえば、次の例内のテキスト **Name:** は、ラベルです。 `Name: _____`	CparmForeground	DmBlack
SAS.cparmMark	FIND、CUT、COPY のような操作に選択された領域。	CparmForeground	DmBlack/ DmReverse
SAS.cparmMessage	メッセージフィールド用。	CparmForeground	DmRed
SAS.cparmNote	**Log** ウィンドウまたは **Output** ウィンドウへと書き込まれる NOTE の行。	CparmForeground	DmBlue
SAS.cparmSecondaryBackground	セカンダリウィンドウ内の背景。	CparmForeground	DmGray
SAS.cparmSecondaryBorder	セカンダリウィンドウの境界線。	CparmForeground	DmBlack
SAS.cparmSource	**Log** ウィンドウへと書き込まれる SAS ソース行。	CparmForeground	DmBlack

リソース名	色と属性の設定	クラス名	デフォルトの色
SAS.cparmText	行情報のテキストラベル。SAS エディタ内で、このリソースを使い、編集中の行を識別でき、また、スプレッドシートウィンドウ内で、スプレッドシート行にラベルを付けられます。	CparmForeground	DmBlue
SAS.cparmTitle	**Output** ウィンドウに書き込まれる TITLE 行。	CparmForeground	DmBlue
SAS.cparmWarning	**Log** ウィンドウまたは **Output** ウィンドウへと書き込まれる WARNING の行。	CparmForeground	DmGreen

DmColorName は、次のいずれかになります。

- DmBLUE
- DmRED
- DmPINK
- DmGREEN
- DmCYAN
- DmYELLOW
- DmWHITE
- DmORANGE
- DmBLACK
- DmMAGENTA
- DmGRAY
- DmBROWN

DmAttrName は、次の属性のいずれかになります。

- DmHIGHLIGHT
- DmUNDERLINE
- DmREVERSE

たとえば、次のリソースでは、すべての背景色は灰色に、すべての前景色は黒に指定されます。

```
SAS.cparmBackground: DmGRAY
SAS.cparmForeground: DmBLACK
```

これらのリソースでは、エラーは反転表示の赤色で表示、警告は反転表示の黄色と太字で表示を指定します。

```
SAS.cparmError: DmRED + DmREVERSE
SAS.cparmWarning: DmHIGHLIGHT + DmYELLOW + DmREVERSE
```

SAS により、デフォルトの CPARMS リソースは 2 箇所で検索されます。

- SAS の現場サポート要員が、SASHELP.BASE.SAS.CPARMS カタログエントリ内に色と属性の設定を入力した場合、これらの設定はサイトでのデフォルトになります。
- SASUSER.PROFILE.SAS.CPARMS 内に設定を保存した場合、これらの設定はサイトで指定された設定よりも優先されます。

コントラストの制御

色の組み合わせによっては、テキストフィールド、ボタン、チェックボックス、その他の前景カテゴリが見えなくなることもあります。**SAS.dmsContrastCheck** リソースは、これらのカテゴリを判読可能にします。

SAS.dmsContrastCheck: True | False
コントラストマッピングが、SAS ウィンドウ内の非グラフィック前景色に適用されるかどうか指定します。デフォルトの値は False です。True の値は、必要であれば DMS 前景色が再びマッピングされて、コントラストが作られる指定です。グラフィックの操作を基にした色の使用が、このリソースに影響を受けないこともあります。

UNIX 環境でドロップダウンメニューを制御する

ドロップダウンメニューは、次のリソースによって制御されます。

SAS.pmenuOn: True | False
WSAVE コマンドで保存された情報にかかわらず、全体的な PMENU 状態を強制的に有効にします。個々のウィンドウの WSAVE の状態は、全体的な状態よりも優先されます。デフォルトは True です。(また、PMENU ON コマンドと PMENU OFF コマンドを使って、ドロップダウンメニューを有効/無効にできます。)

SAS.usePmenuMnemonics: True | False
現在の SAS セッションのドロップダウンメニューに、ニーモニックを加えるかどうかを指定します。デフォルトは True です。

UNIX 環境での切り取りと貼り付けのカスタマイズ

テキストの切り取り法、貼り付け法

テキストの切り取りと貼り付けの説明は、“UNIX 環境でテキストを選択する(マークを付ける)” (159 ページ) および “UNIX 環境での選択したテキストのコピー、切り取り、貼り付け” (161 ページ)を参照してください。

貼り付けバッファの種類

4 つの SAS 貼り付けバッファが存在します。それぞれの SAS 貼り付けバッファは、X 貼り付けバッファに関連付けられます。

XPRIMARY
X プライマリの選択(PRIMARY)に関連します。

XSCNDARY
X セカンダリの選択(SECONDARY)に関連します。

XCLIPBRD
X のクリップボードの選択(CLIPBOARD)に関連します。この貼り付けバッファでは、SAS に MIT X Consortium X クリップボードが使用できます。

XTERM
xterm クライアントで使われる貼り付けバッファに関連します。XTERM がデフォルトのバッファです。DEFAULT は、XTERM のエイリアスです。XTERM バッファへとテキストをコピーまたは切り取りする場合、そのテキストは実際は 4 つすべての貼り付けバッファにコピーまたは切り取りされます。XTERM バッファからテキストを貼り付けるとき、そのテキストは XPRIMARY バッファから貼り付けられます。

XCUT*n*
0 <= *n* <= 7 である X 切り取りバッファ *n* と関連します。

貼り付けバッファの選択

他の X クライアントが使用しているのが、どの X データ交換プロトコルなのか確かでない場合に、XTERM 貼り付けバッファを使用してください。
SAS.defaultPasteBuffer リソースを使って、デフォルトの貼り付けバッファを指定できます。

```
SAS.defaultPasteBuffer: XTERM
```

使用するワークステーション環境の X クライアントすべてが、データの交換に X PRIMARY 選択を使うのがわかっている場合は、XPRIMARY 貼り付けバッファを使用してください。

```
SAS.defaultPasteBuffer: XPRIMARY
```

この指定により、SAS リソースと X リソースの両方がより効果的に使われ、クライアント間のオンデマンドのデータ転送が提供されます。

Solaris OpenWindows デスクトップクライアントでは、コピーアンドペースト操作の土台として CLIPBOARD 選択が使用されます。SAS XCLIPBRD 貼り付けバッファを使用する場合、これらのクライアントと直接にテキストを交換できます。

また、SAS XCLIPBRD 貼り付けバッファを使用して、テキストの王冠に Motif クリップボード機構を使用する Motif クライアントと情報をやりとりできます。このクリップボード機構により、xclipboard のような専用クライアントは不要になります。たとえば、xmeditor**Edit** ドロップダウンメニューから、**Cut**、**Copy**、**Paste** の項目を選ぶとき、XCLIPBRD を使用して、Motif の xmeditor アプリケーションと直接テキストを交換できます。

Motif のクイックコピーデータ交換機構と Motif のクリップボードデータ交換機構は、Motif インターフェイスツールキットに特有のもので、SAS の貼り付けバッファとしては現在、サポートされていません。ただし、**File Selection** ダイアログボックスなどのいくつかのダイアログボックスでは、Motif インターフェイスのテキストウィジェットを使用します。これらのダイアログボックス内では、Motif のクイックコピーおよびクリップボードデータ交換機構が使用できます。

貼り付けバッファを使用してテキストを操作する

毎回、マウスでテキストの範囲をマークするときに貼り付けバッファ内に選択したテキストを自動的にコピーされるようにする場合、**SAS.markPasteBuffer** リソース内の貼り付けバッファ名を指定するべきです。

```
SAS.markPasteBuffer: XTERM
```

他の手段としては、DEFAULT は XTERM のエイリアスであるため、次のように指定もできます。

```
SAS.markPasteBuffer: DEFAULT
```

この **SAS.markPasteBuffer** の定義により、テキストを選択するときはいつでも、SAS によって自動的に STORE コマンドが発行されます。

STORE コマンドは、CUT コマンドと PASTE コマンド同様、使用バッファを指定する BUFFER=オプションをサポートします。これらのコマンドが、BUFFER=オプションを含まないファンクションキーまたはドロップダウンメニューから発行されるとき、**SAS.markPasteBuffer** リソースが定義されていない場合に、これらのコマンドでは BUFFER=DEFAULT が使用されます。このリソースが定義されている場合、これらのコマンドでは BUFFER=*buffer-name* が使用されます。

通常の切り取り、コピー、または貼り付けキーをカスタマイズして、BUFFER=オプションを使って、これらのコマンドのどれでも発行できます。たとえば、**osfCopy** キーおよび **osfPaste** キーの **SAS.keyboardTranslations** 定義より、次の指定を優先できます。

```
SAS.keyboardTranslations: #override \
<Key>osfCopy: sas-do-command(\"STORE BUFFER=XCLIPBRD\") \n\
<Key>osfPaste: sas-do-command(\"PASTE BUFFER=XCLIPBRD\")
```

キーのカスタマイズに関する詳細については、“UNIX 環境でキー定義をカスタマイズする” (190 ページ)を参照してください。

テキストと属正情報の保存についての注

テキストを切り取り、またはコピーして、XTERM、XPRIMARY または XSCNDARY 貼り付けバッファを使って SAS セッション間にテキストを貼り付ける場合、色と属性の情報が保存されます。ただし、vi エディタを使用中に、同じテキストを xterm ウィンドウへとコピーし貼り付ける場合、色と属性の情報は失われます。
SAS.defaultPasteBuffer および **SAS.markPasteBuffer** の定義を XCUT0 へ変更すると、2 つの SAS セッション間でテキストをコピーして貼り付けるときに、テキストと色の属性は保持されません。

xclipboard クライアントを使用する場合、SAS のテキスト属性は SAS セッション間で作られた交換内には保存されません。ただし、xclipboard クライアントのようなクリップボードマネージャなしで XCLIPBRD 貼り付けバッファを使用する場合、SAS のテキスト属性は、SAS セッション間の交換に保存されます。

UNIX 環境でセッションワークスペース、セッショングラビティ、ウィンドウサイズをカスタマイズする

SAS では、次のリソースを使用して、セッションワークスペースのサイズ、ワークスペースのグラビティ、ウィンドウのサイズが指定されます。これらのリソースのデフォルト値は、表 9.4 (216 ページ)にリストされています。

SAS.awsResizePolicy: grow | fixed
内部ウィンドウが追加/削除される際の、AWS ウィンドウのサイズ変更方法を指定します。次の値が有効です。

grow
AWS ウィンドウは、内部ウィンドウを拡大または移動するとき常に、すべての内部ウィンドウを表示するため拡大しようとしますが、使用しない領域が削除されても縮小することはありません。

fixed
AWS ウィンドウは、最初の内部ウィンドウのサイズに自身でサイズ調整し、それ以降はサイズ変更しません。

SAS.maxWindowHeight: units
ウィンドウの最大の高さを単位数で指定します。単位は、**SAS.windowUnitType** リソースによって指定されます。

SAS.maxWindowWidth: units
ウィンドウの最大の幅を単位数で指定します。単位は、**SAS.windowUnitType** リソースによって指定されます。

SAS.noAWS: True | False
アプリケーションの各ウィンドウが、アプリケーションワークスペース(AWS)ではなく、自身のネイティブウィンドウに表示されるかを制御します。デフォルトは True です。各アプリケーションは、自身のネイティブウィンドウ内で実行されます。

SAS.scrollBarSize: pixels
スクロールバーのデフォルトのサイズをピクセル単位で指定します。

SAS.sessionGravity: value
SAS がウィンドウを配置しようとする画面の領域を制御します。このリソースは、ウィンドウマネージャの構成によっては、無視されることもあります。使用可能な値は次のとおりです。

- **CenterGravity**
- **EastGravity**
- **WestGravity**
- **SouthGravity**
- **NorthGravity**
- **SouthEastGravity**
- **NorthEastGravity**
- **SouthWestGravity**
- **NorthWestGravity**

SAS.sessionGravityXOffset: offset
グラビティ領域にウィンドウが置かれるときに追加される x オフセットを指定します。

SAS.sessionGravityYOffset: offset
グラビティ領域にウィンドウが置かれるときに追加される y オフセットを指定します。

SAS.windowHeight: units
ウィンドウのデフォルトの高さを単位数で指定します。単位は、**SAS.windowUnitType** リソースによって指定されます。

SAS.windowUnitType: character | pixel | percentage
SAS.windowWidth、**SAS.windowHeight**、**SAS.maxWindowWidth**、**SAS.maxWindowHeight** の単位の種類を指定します。使用可能な値は次のとおりです。

character
行数と列数で指定します。

pixel
ピクセル数で指定します。

percentage
画面に対するパーセントで指定します。

SAS.windowWidth: units
ウィンドウのデフォルトの幅を単位数で指定します。単位は、**SAS.windowUnitType** リソースによって指定されます。

UNIX 環境でユーザー定義のアイコンを指定する

ユーザー定義のアイコンを指定する理由

SAS が提供するアイコンに、ユーザー自身のアイコンを追加できます。たとえば、ツールボックスで独自色のアイコンを使用する場合、**SAS.colorUiconPath**、**SAS.colorUiconCount**、および **SAS.sasUiconx** リソースを定義します。Tool Editor でツールを定義する場合、Tool Editor は、各ツールで選択可能なアイコンの表示の中にユーザーのアイコンを含めます。

SAS によるユーザー定義のアイコンの検索法

ユーザーアイコン数 *x* にアイコンビットマップファイル名を検索するために使用されるリソースの名前は、**SAS.sasUiconx** です。たとえば、次のリソースでは、ユーザーアイコン 1 にファイル名 **myicon** が定義されます。

```
SAS.sasUicon1: myicon
```

リソース名が定義されない場合、SAS によって、**sasui*nnn*.xbm** または **sasui*nnn*.xpm** の形式のファイル名が作成されます。**SAS.uiconPath** リソースまたは **SAS.colorUiconpath** リソースからのパス構成要素は、アイコンファイルが見つかるまで、または検索パスが底を突くまで、順に検索されます。

たとえば、X リソースの次のセットでは、色アイコンの集合が定義されます。

```
SAS.colorUiconPath: /users/jackaroe/pixmaps/
SAS.colorUiconCount: 7
SAS.sasUicon1: adsetup
SAS.sasUicon2: adverse
SAS.sasUicon3: altmenu
SAS.sasUicon4: batch
SAS.sasUicon5: is
SAS.sasUicon6: patgrps
SAS.sasUicon7: pctchg
```

Motif インターフェイスは、**/users/jackaroe/pixmaps/adsetup.xpm** という名前のファイル内でアイコン **sasUicon1** を検索します。

ユーザー定義のアイコンを指定する X リソース

SAS では、次のリソースを使って、利用可能なユーザー定義のアイコン数とその位置が指定されます。

SAS.colorUiconPath: search-path
ユーザー定義の色アイコンファイルを検索するための、ファイル検索パスを指定します。この文字列リソースにより、アイコンファイルを検索するためのディレクトリパスが指定されます。これらのファイルは、X Pixmap (xpm)形式である必要があります。カンマを使って、個々のディレクトリパス名を分離します。たとえば、次の文字

列では、最初に**/usr/lib/X11/pixmaps** ディレクトリでアイコンファイルを検索し、次に**/usr/lib/X11/pixmaps/SAS** ディレクトリ内で検索します。

```
SAS.colorUiconPath : /usr/lib/X11/pixmaps, \
/usr/lib/X11/pixmaps/SAS
```

SAS.colorUiconCount: num-icons
SAS が使用するために利用できるユーザー定義の色アイコンの数を指定します。

SAS.uiconCount: num-icons
SAS セッションで使用するために利用できるユーザー定義のアイコンの数を指定します。

SAS.uiconPath: search-path
ユーザー定義のアイコンビットマップファイルを検索するための、ファイル検索パスを指定します。この文字列リソースにより、アイコンファイルを検索するためのディレクトリパスが指定されます。これらのファイルは、X Bitmap (xbm)形式である必要があります。カンマを使って、個々のディレクトリパス名を分離します。たとえば、次の文字列では、最初に**/usr/lib/X11/bitmaps** ディレクトリでビットマップファイルを検索し、次に**/usr/lib/X11/bitmaps/SAS** ディレクトリ内で検索します。

```
SAS.uiconPath : /usr/lib/X11/bitmaps,\
/usr/lib/X11/bitmaps/SAS
```

SAS.sasUiconx: name
値を X Bitmap ファイルまたは Pixmap ファイルのファイル名に関連付けます。*x* は、そのファイルに割り当てられた数字です。ファイル拡張子は**.xbm** または**.xpm** が自動的に提供されます。

UNIX 環境の各種リソース

次のリソースをカスタマイズすることもできます。

SAS.altVisualId: ID
ビジュアルタイプ ID を指定します。

SAS.autoSaveInterval: minutes
Program Editor ウィンドウからのデータが保存される頻度を分数で指定します。

SAS.autoSaveOn: True | False
Program Editor ウィンドウからのデータが、**SAS.autoSaveInterval** リソースで指定される間隔でファイルに保存されることを指定します。

SAS.confirmSASExit: True | False
DLGENDR コマンドを入力または **File** ⇨ **Exit** を選択時に、SAS によって、**Exit** ダイアログボックスが表示さるかどうかを指定します。デフォルトは True です。

SAS.defaultCommandWindow: True | False
SAS セッションを開始するときに、コマンドウィンドウを呼び出すかどうかを指定します。デフォルトは True です。

SAS.directory: directory-pathname
初めて **Open** ダイアログボックスを呼び出すときに、必要なディレクトリを指定します。デフォルトでは、**Open** ダイアログボックスは、カレントディレクトリを使用します。

SAS.helpBrowser: pathname
オンラインヘルプを表示するために使用するワールドワイドウェブブラウザのパス名またはいつ WBROWSE コマンドを発行するかを指定します。デフォルトのブラウザは、Netscape です。

SAS.htmlUsePassword: True | False
ブラウザに HTML ファイルを送る前に、パスワードの入力を要求するかどうかを指定します。デフォルト値は True です。

SAS.insertModeOn: True | False
SAS エディタウィンドウ内の編集モードを指定します。デフォルトの値は False(上書き)です。

SAS.noDoCommandRecall: True | False
sas-do-command() 操作ルーチンを使ってサブミットされる SAS コマンドが、コマンド再現バッファ内に記録されるかどうかを指定します。デフォルトの値 True では、コマンドはコマンド再現バッファから除外されます。False の値では、コマンドは記録されます。

SAS.pattern: default-pattern
初めて **Open** ダイアログボックスおよび **Import Image** ダイアログボックスを呼び出すときにファイルフィルタとして使用されるデフォルトのパターンを指定します。このパターンは、ダイアログボックスの最上段のテキストフィールド内に表示されます。デフォルトでは、ダイアログボックスは **File** タイプリストの最初にあるフィルタを使用します。パターンのリソースは、**File** タイプフィールドには影響を与えません。

SAS.selectTimeout: seconds
X Toolkit の選択変換のタイムアウトの値を秒単位で指定します。このタイムアウトの値は、SAS が、X Toolkit 選択を変換して完了するための要求を待機する時間の合計を決定します。デフォルトの値はほとんどの場合で適切である必要があります。

SAS.startupLogo: xpm-filename | None | ""
SAS が初期化されるときに表示する XPM ファイルを指定します。文字列が空の場合、デフォルトのロゴが使用されます。

SAS.startSessionManager: True | False
新規の SAS セッションが開始されるときに、SAS Session Manager が自動的に開始されるかどうかを指定します。SAS にユーザー自身のホストエディタを使用するときは、SAS Session Manager が実行されている必要があります。デフォルトは True です。

SAS.suppressMenuIcons: True | False
チェックボックス、およびカスケード表示またはポップアップ表示メニュー内のトグルボタン以外のメニューアイコンが表示されるかどうかを指定します。アイコンを非表示にするとメモリ使用量が減少するため、動作が遅くなってきた X サーバーでメニューの表示速度が改善されます。デフォルトは False です。

SAS.suppressTutorialDialog: True | False
Getting Started Tutorial ダイアログボックスを SAS セッションの開始時に表示させるかどうかを指定します。True の値は、ダイアログボックスの表示を抑制します。すでに SAS を使ったことがある場合、このダイアログボックスを抑制する場合もあります。デフォルトは False です。

SAS.useNativeXmTextTranslations: True | False
XmText ウィジェットの変換が、SAS X Motif ユーザーインターフェイスが使用する Text、Combo Box および Spin Box ウィジェットのすべてのインスタンスに継承されるかどうかを指定します。値が False の場合、SAS のキーウィンドウ変換は、あらゆるユーザーが提供する、あるいはシステムが提供する XmText 変換に優先します。デフォルト値は True です。

次の例では、SAS XmText 変換を示します。

```
SAS*XmText*translations: #override \n\
Ctrl<Key>e:end-of-line()\n\
Ctrl<Key>u:delete-to-start-of-line()\n\
Ctrl<Key>k:delete-to-end-of-line()\n\
Ctrl<Key>f:forward-character()\n\
Ctrl<Key>b:backward-character()\n\
Ctrl<Key>a:beginning-of-line()\n\
Ctrl<Key>c:copy-clipboard()\n\
Ctrl<Key>v:paste-clipboard()\n\
```

SAS.wsaveAllExit: True | False
セッションを終了するときに、SAS が WSAVE ALL コマンドを発行するかどうかを指定します。このコマンドにより、ウィンドウの色と位置のような全体的な設定が保存されます。これらの設定は現在開いているすべてのウィンドウに有効です。デフォルトは False です。

注: WSAVE コマンドが機能するには、ウィンドウマネージャによって、正確なウィンドウ配置をサポートされている必要があります。ウィンドウマネージャの設定方法を決定するには、ウィンドウマネージャのドキュメントを参照してください。たとえば、Exceed を実行中の場合、**Screen Definition Settings** ダイアログボックスを開き、**Cascade Windows** の選択を解除します。

UNIX 環境で SAS が使用する X リソースのまとめ

次の表では、インスタンス名とクラス名、タイプおよび多数の SAS リソースのデフォルトの値を示します。特定の種類の追加のリソースについては、次のセクションを参照してください。

- “フォントのエイリアスの指定” (201 ページ)
- “色のリソースの定義” (204 ページ)
- “ウィンドウ要素の色と属性の定義(CPARMS)” (206 ページ)

表 9.4 *SAS リソース*

リソース名	クラス名	種類	デフォルト
SAS.altVisualId	AltVisualId	整数	NULL
SAS.autoComplete	AutoComplete	ブール	True
SAS.autoSaveInterval	AutoSaveInterval	整数	10
SAS.autoSaveOn	AutoSaveOn	ブール	True
SAS.awsResizePolicy	AWSResizePolicy	文字列	grow
SAS.colorUiconCount	UiconCount	整数	0
SAS.colorUiconPath	UiconPath	文字列	NULL

リソース名	クラス名	種類	デフォルト
SAS.commandsSaved	CommandsSaved	整数	25
SAS.confirmSASExit	ConfirmSASExit	ブール	True
SAS.defaultCommandWindow	DefaultCommandWindow	ブール	True
SAS.defaultPasteBuffer	DefaultPasteBuffer	文字列	XTERM
SAS.defaultToolBox	DefaultToolBox	ブール	True
SAS.directory	Directory	文字列	NULL
SAS.dmsContrastCheck	DmsContrastCheck	ブール	False
SAS.DMSDBFont	Font	文字列	*dynamic*
SAS.DMSDBboldFont	Font	文字列	*dynamic*
SAS.DMSboldFont	Font	文字列	*dynamic*
SAS.DMSFont	Font	文字列	*dynamic*
SAS.DMSfontPattern	DMSFontPattern	文字列	-*-*-*-r-*--*-*-*-*-m-*-iso8859-1
SAS.fontPattern	FontPattern	文字列	*
SAS.helpBrowser	HelpBrowser	文字列	Netscape
SAS.htmlUsePassword	HtmlUsePassword	ブール	True
SAS.insertModeOn	InsertModeOn	ブール	False
SAS.isToolBoxPersistent	IsToolBoxPersistent	ブール	True
SAS.keyboardTranslations	KeyboardTranslations	変換	*dynamic*
SAS.keysWindowLabels	KeysWindowLabels	文字列	*dynamic*
SAS.markPasteBuffer	MarkPasteBuffer	文字列	XTERM
SAS.maxWindowHeight	WindowHeight	ディメンション	95
SAS.maxWindowWidth	WindowWidth	ディメンション	95
SAS.noAWS	NoAWS	ブール	True
SAS.noDoCommandRecall	NoDoCommandRecall	ブール	True
SAS.pattern	Pattern	文字列	NULL

リソース名	クラス名	種類	デフォルト
SAS.pmenuOn	PmenuOn	ブール	True
SAS.sasUicon	SasUicon	文字列	NULL
SAS.scrollBarSize	ScrollBarSize	ディメンション	17
SAS.selectTimeout	SelectTimeout	整数	60
SAS.sessionGravity	SASGravity	文字列	NorthWestGravity
SAS.sessionGravityXOffset	SASGravityOffset	整数	0
SAS.sessionGravityYOffset	SASGravityOffset	整数	0
SAS.startSessionManager	StartSessionManager	ブール	True
SAS.startupLogo	StartUpLogo	文字列	NULL
SAS.suppressMenuIcons	SuppressMenuIcons	ブール	False
SAS.suppressTutorialDialog	SuppressTutorialDialog	ブール	False
SAS.systemFont	SystemFont	文字列	“-adobe-helvetica-medium-r-normal--12–*–*–*–*–*–*–*”
SAS.toolBoxAlwaysOnTop	ToolBoxAlwaysOnTop	ブール	True
SAS.toolBoxTipDelay	ToolBoxTipDelay	整数	750
SAS.uiconCount	UiconCount	整数	0
SAS.uiconPath	UiconPath	文字列	NULL
SAS.useCommandToolBoxCombo	UseCommandToolBoxCombo	ブール	True
SAS.useLargeToolBox	UseLargeToolBox	ブール	False
SAS.useNativeXmTextTranslations	UseNativeXmTextTranslations	ブール	False
SAS.usePmenuMnemonics	UsePmenuMnemonics	ブール	True
SAS.useShowHideDecorations	UseShowHideDecorations	ブール	False
SAS.useToolBoxTips	UseToolBoxTips	ブール	True
SAS.wsaveAllExit	WsaveAllExit	ブール	False
SAS.windowHeight	WindowHeight	ディメンション	50

リソース名	クラス名	種類	デフォルト
SAS.windowWidth	WindowWidth	ディメンション	67
SAS.windowUnitType	WindowUnitType	文字列	percentage

3 部

データに関する注意点

10 章

データ表現

UNIX 環境の数値変数の長さと精度

SAS データセットの数値変数の長さは 8 バイトです。(DATA ステップの LENGTH または ATTRIB ステートメントにて SAS 数値変数の長さを設定することができます)。

数値精度の発行はほとんど全ての SAS 計算機能の応答値と SAS プロシジャから戻る多く数値に影響します。UNIX のための SAS の数値は IEEE 倍精度浮動小数点数として表現されます。完全 8 バイトの数字の小数精度は実質小数 15 桁です。

次の表は SAS 数値変数に保存することが可能な有効桁と最大の整数を指定します。

表 10.1 *UNIX での SAS 変数のための有効桁と最大整数長さ*

バイト長さ	有効桁 維持される	最大整数 正確に表現される
3	3	8,192
4	6	2,097,152
5	8	536,870,912
6	11	137,438,953,472
7	13	35,184,372,088,832
8	15	9,007,199,254,740,992

変数長さを指定しているときに、変数の長さは使用されるディスクスペースの容量とデータセットの読み書きに必要な I/O 操作の数に影響を与えることに留意してください。

数値変数の値が-8192 と 8192 の間の整数に含まれていることがわかっている場合、3 の長さにてその数値を保存しデータセットのスペースを節約することが可能です。例えば、

```
data mydata;
length num 3;
...more SAS statements...
run;
```

数字ダミー変数(0 か 1 を保持するためだけの目的を持つ変数) は 3 バイトの長さである変数に保存することが可能です。

注意:

LENGTH ステートメントを使用して、常に整数を値とする変数のみの長さを短くします。 分数の数字は切り捨てられると精度を失います。さらに、変数の値は指定するバイト数で正確に表現されるようにする必要があります。TRUNC 機能を使って DATA ステップにてプログラムで行うことが可能です。LENGTH ステートメントで長さを指定することでデータの切り捨てとなった場合、警告やエラーは発生しません。

変数長の指定とシステム動作の最適化の詳細については、*SAS 言語リファレンス: 解説編*を参照してください。

UNIX 環境の欠損値

UNIX での SAS では、値の欠如は、IEEE の非数字の値によって表現されます。IEEE の非数字の値は、有効な数値以外にて表現される IEEE 浮動小数点ビットパターンです。これらの数字は計算では導くことはできません。

UNIX 環境でのバイナリデータの読み込みと書き込み

数値のバイナリデータの保存方法は、コンピュータによってさまざまです。バイナリデータをフラットファイルにて互換性のないシステム間で移動しようとする場合、問題が起きます。データを移動する安全な方法は SAS データセットを使用する方法です。

SAS はバイナリデータの処理のための入力形式と出力形式を提供します。これらの入力形式および出力形式はホストに依存します。たとえば、IB*w.d*、PD*w.d*、PIB*w.d*、および RB*w.d* の入力形式と出力形式はネイティブモードでデータの読み書きを行います。つまり、コンピュータで標準であるバイト順システムを使用しています。

入力形式および出力形式の詳細については、*SAS 出力形式と入力形式: リファレンス*を参照してください。

UNIX 日時値を SAS 日時値に変換する

UNIX 日時値は、1970 年 1 月 1 日からの秒数で保存されます。SAS 日時値は、1960 年 1 月 1 日からの秒数で保存されます。UNIX 日時値を SAS 日時値に変換するには、10 年分の秒数を UNIX 日時値に追加する必要があります。

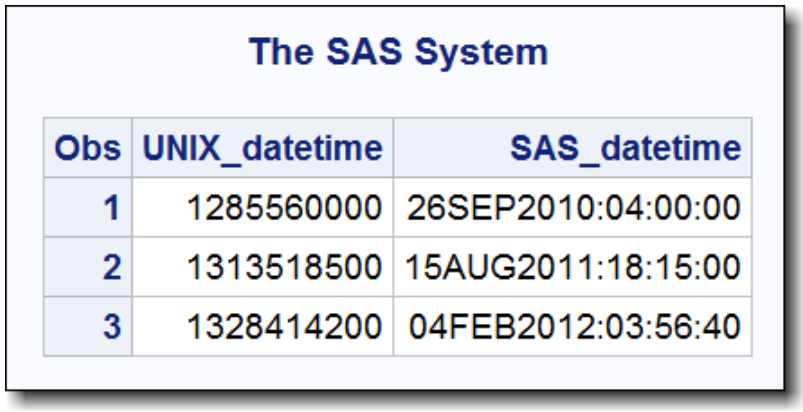
INTNX 機能は、次の例で示すように、UNIX 日時値を SAS 日時値に変更します。

```
data UNIX_to_SAS;
input UNIX_datetime;
```

```
/* The INTNX function accounts for leap years. */
SAS_datetime = intnx('DTyear',UNIX_datetime,10,'s');
format SAS_datetime datetime20.;
datalines;
1285560000
1313518500
1328414200
;
proc print data=UNIX_to_SAS;
run;
```

次の出力は結果を表示します。

画面 10.1 *UNIX 日時値から SAS 日時値への変換*



The SAS System

Obs	UNIX_datetime	SAS_datetime
1	1285560000	26SEP2010:04:00:00
2	1313518500	15AUG2011:18:15:00
3	1328414200	04FEB2012:03:56:40

詳細については、“INTNX Function” (*SAS Functions and CALL Routines: Reference*)を参照してください。

4 部

SAS 言語のホストに固有の機能

11 章

UNIX 版に固有のコマンド

UNIX 版に固有の SAS コマンド

このセクションでは、SAS のウィンドウ環境でコマンド行に入力できるコマンドについて説明します。ここで説明するコマンドには、UNIX 環境に固有の動作または構文を含んでいます。各コマンドの説明には、そのコマンドのどの要素が UNIX 版に固有であるのかを簡単に説明する"UNIX 固有"セクションが含まれています。"UNIX 固有"の情報が"すべて"であれば、そのコマンドは UNIX 動作環境に適用され、これを説明するのは本書のみとなります。

次のコマンドは UNIX 環境ではサポートされません。

CASCADE	RESIZE	WGROW
DCALC	SCROLLBAR	WMOVE
ICON	SMARK	WSHRINK
PCLEAR	TILE	ZOOM

ディクショナリ

AUTOSCROLL コマンド:UNIX

出力を表示するために Log と Output ウィンドウをスクロールする頻度を指定します。

UNIX 固有: 有効な引数およびデフォルト値

構文

AUTOSCROLL<*n*>

オプション引数

n
ウィンドウに収まらないデータ行を受け入れるときにスクロールする行数を指定します。

詳細

AUTOSCROLL コマンドは、**Log** ウィンドウと **Output** ウィンドウに書き込まれるときの行のスクロールを管理します。**Log** ウィンドウと **Output** ウィンドウでの AUTOSCROLL のデフォルト値は `1` です。AUTOSCROLL が一度に 1 行ずつ表示する場合は処理が遅くなります。処理を迅速にするには、autoexec.sas ファイルの AUTOSCROLL 値をより大きく指定します。0 の値を指定すると処理が最適化され、スクロールが最速になります(xterm ウィンドウのジャンプスクロールと同様)。
AUTOSCROLL コマンドを autoexec.sas ファイルに追加するには、DM コマンドを使用する必要があります。次の例は、**Log** ウィンドウと **Output** ウィンドウでのスクロール幅を最大化したものを示しています。

```
dm 'output; autoscroll 0; log; autoscroll 0; pgm;';
```

CAPS コマンド:UNIX

テキストの大文字小文字のデフォルトの設定を変更します。

UNIX 固有: すべて

構文

CAPS <ON | OFF>

オプション引数

ON
大文字化を有効にします。

OFF
大文字化を無効にします。

COLOR コマンド: UNIX

ウィンドウの選択部分の色と強調表示を指定します。

UNIX 固有: 有効なフィールドタイプおよび属性

構文

COLOR *field-type color* | NEXT <*highlight*>

必須引数

field-type
背景、コマンド、境界線、メッセージなどといったウィンドウの領域を指定します。

color
ブルー(略して B)、レッド(R)、グリーン(G)、シアン(C)、ピンク(P)、イエロー(Y)、ホワイト(W)、ブラック(K)、マゼンタ(M)、グレー(A)、ブラウン(B)、オレンジ(O)などから 1 色を指定する。

NEXT
色を、次の使用可能な色に変更します。

オプション引数

highlight
H (テキストを太字フォントで表示)、U (下線)、R (色の反転)のどれかになります。BLINK 属性はサポートされません。

詳細

UNIX では、BORDER、MENU、MENUBORDER、SCROLLBAR、TITLE などのフィールドタイプで色を変更する場合に COLOR コマンドを使用できません。また、H (HIGHLIGHT)属性と B (BLINK)属性もサポートされていません。COLOR コマンドの

詳細については、**Program Editor** ウィンドウに関するオンラインヘルプを参照してください。

DLGABOUT コマンド: UNIX

About SAS ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGABOUT

詳細

About SAS ダイアログボックスは、実行している SAS のリリース情報、サイト番号、オペレーティングシステム、使用している Motif のバージョン情報、各自の PC の色情報などを表示します。

メニューからこのダイアログボックスにアクセスするには、**Help** ⇨ **About SAS 9** を選択してください。

DLGCDIR コマンド:UNIX

Change Working Directory ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGCDIR

詳細

Change Working Directory ダイアログボックスでは、新しい作業ディレクトリを選択できます。メニューからこのダイアログボックスにアクセスするには、**Tools** ⇨ **Options** ⇨ **Change Directory** を選択してください。

DLGENDR コマンド: UNIX

Exit ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGENDR

詳細

Exit ダイアログボックスでは、SAS の終了を確認されます。**OK** を選択すると、SAS セッションが終了します。**SAS.confirmSASExit** というリソースを **False** に設定すると、このコマンドは BYE コマンドに相当するようになります。メニューからこのダイアログボックスにアクセスするには、**File** ⇨ **Exit** を選択してください。

関連項目:

"UNIX 環境の各種リソース" (214 ページ)

DLGFIND コマンド: UNIX

Find ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGFIND

詳細

Find ダイアログボックスでは、テキスト文字列の検索が可能です。メニューからこのダイアログボックスにアクセスするには、**Edit** ⇨ **Find** を選択してください。

関連項目:

コマンド:

- "DLGREPLACE コマンド: UNIX" (235 ページ)

DLGFONT コマンド: UNIX

Fonts ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGFONT

詳細

Font ダイアログボックスでは、SAS のフォントを動的に変更できます。メニューからこのダイアログボックスにアクセスするには、**Tools** ⇨ **Options** ⇨ **Fonts** を選択してください。

関連項目:

コマンド:

- “SETDMSFONT コマンド: UNIX” (243 ページ)

その他の参照資料:

- “UNIX 環境でフォントをカスタマイズする” (198 ページ)

DLGOPEN コマンド: UNIX

Open ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGOPEN <FILTERS='*filters*' <IMPORT> <SUBMIT|NOSUBMIT> <VERIFY>>

オプション引数

FILTERS='*filters*'
ファイルを表示するときに検索基準として使用するために、1 つ以上のファイルフィルタを指定します。たとえば次のコマンドは、カレントディレクトリ内の **.sas** 拡張子がついているファイルをすべて表示し、***.txt** をこのダイアログボックスの **File type** ボックスに追加します。

```
DLGOPEN FILTERS="*.sas *.txt"
```

複数のフィルタを指定できます。指定したフィルタはすべてこのボックスに表示されます。どのフィルタも指定しない場合は、このダイアログボックスにはデフォルトのリストが表示されます。デフォルトのファイルパターンの指定については、“UNIX 環境の各種リソース” (214 ページ)の **SAS.pattern** リソースの説明を参照してください。

IMPORT
Import Image ダイアログボックスを呼び出します。このダイアログボックスでは、SAS/GRAPH アプリケーションにグラフィックファイルをインポートできます。

SUBMIT|NOSUBMIT
ファイルが開いた後で SUBMIT コマンドをを押すかどうかを指定します。

VERIFY
DLGOPEN コマンドがアクティブウィンドウに適しているかを確認します。

詳細

Open ダイアログボックスと **Import Image** ダイアログボックスでは、アクティブウィンドウに読み込むファイルを選択できます。アクティブウィンドウが SAS/GRAPH ウィンドウの場合、**Import Image** ダイアログボックスが表示されます。別のウィンドウの場合は、**Open** ダイアログボックスが表示されます。メニューからこれらのダイアログボックスにアクセスするには、**File** ⇨ **Open** または **File** ⇨ **Import Image** を選択してください。

詳細については、*SAS/GRAPH: Reference* の“Specifying Images in SAS/GRAPH Programs”を参照してください。

DLGPREF コマンド: UNIX

Preferences ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGPREF

詳細

Preferences ダイアログボックスでは、特定のリソース設定を動的に変更できます。メニューからこのダイアログボックスにアクセスするには、**Tools** ⇨ **Options** ⇨ **Preferences** を選択してください。

関連項目:

"Preferences ダイアログボックスを使用し、X リソースを変更する" (172 ページ)

DLGREPLACE コマンド: UNIX

Change ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGREPLACE

詳細

Change ダイアログボックスでは、テキスト文字列の検索と置換が可能です。メニューからこのダイアログボックスにアクセスするには、**Edit** ⇨ **Replace** を選択してください。

関連項目:

コマンド:

- "DLGFIND コマンド: UNIX" (233 ページ)

DLGSAVE コマンド:UNIX

Save As または **Export** ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGSAVE <FILTERS='*filters*' <EXPORT> <VERIFY>>

オプション引数

FILTERS='*filters*'
ファイルを表示するときに検索基準として使用するために、1 つ以上のファイルフィルタを指定します。たとえば、次のコマンドは、カレントディレクトリ内の **.sas** 拡張子がついているファイルをすべて表示し、***.txt** をこのダイアログボックスの **File type** ボックスに追加します。

```
DLGSAVE FILTERS="*.sas *.txt"
```

複数のフィルタを指定できます。指定したフィルタはすべてこのボックスに表示されます。どのフィルタも指定しない場合は、このダイアログボックスにはデフォルトのリストが表示されます。

EXPORT
Export ダイアログボックスを呼び出します。このダイアログボックスでは、各自の SAS セッションのグラフィックファイルをエクスポートできます。

VERIFY
DLGSAVE コマンドがアクティブウィンドウに適しているかを確認します。.

詳細

メニューからこれらのダイアログボックスにアクセスするには、**File** ⇨ **Save as** または **File** ⇨ **Export as Image** を選択してください。

DLGSCRDUMP コマンド: UNIX

アクティブな SAS/GRAPH ウィンドウを、指定するファイル名とファイルタイプを使用して、画像ファイルとして保存します。

UNIX 固有: すべて

構文

DLGSCRDUMP <'*filename.ext*' 'FORMAT=*file-type*'>

詳細

DLGSCRDUMP は、アクティブな SAS/GRAPH ウィンドウを、指定するファイル名とファイルタイプを使用して、画像ファイルとして保存します。引数を指定しない場合、DLGSCRDUMP により **Export** ダイアログボックスが開き、そこでファイル名とファイルタイプを選択できます。SAS/GRAPH にサポートされている画像形式でディスプレイを保存します。画像の指定の詳細については、19 章: “Adding Images To SAS/GRAPH Output” (*SAS/GRAPH: Reference*)を参照してください。

DLGSMAIL コマンド: UNIX

Send Mail ダイアログボックスを開きます。

UNIX 固有: すべて

構文

DLGSMAIL

詳細

Send Mail ダイアログボックスでは、SAS での作業中に電子メールを送信できます。メニューからこのダイアログボックスにアクセスするには、**File** ⇨ **Send mail** を参照してください。

関連項目:

システムオプション:

- “EMAILSYS システムオプション: UNIX” (375 ページ)

その他の参照資料:

- “FILENAME ステートメント(EMAIL)を使用し、電子メールを送信する” (82 ページ)
- “UNIX 環境で SAS セッションからメールを送信する” (164 ページ)

FILE コマンド: UNIX

現在のウィンドウのコンテンツを外部ファイルに書き込みます。

UNIX 固有: 次の項目の有効な値: *encoding-value* および *host-options*

構文

FILE <*file-specification*> <ENCODING='*encoding-value*'> <*portable-options*> <*host-options*>

オプション引数

file-specification
次のいずれかになります。

single filename
SAS はカレントディレクトリに対象ファイルを書き込みます。ファイル名を引用符で囲む場合、SAS では指定するとおりのファイル名が使用されます。ファイル名を引用符で囲まず、ファイル名の拡張子を特に指定しない場合、SAS では、**Program Editor**,ウィンドウ、**Log** ウィンドウ、**Output** ウィンドウからこのコマンドを発行するかどうかによって、.sas、.log または.lst が使用されます。

entire pathname
パス名を引用符で囲まない場合でも、SAS はファイル名の拡張子を一切予測しません。

fileref
SAS は外部ファイルに割り当てるファイル参照名を指定します。

ENCODING='*encoding-value*'
出力ファイルへの書き込み時に使用するエンコーディングを指定します。ENCODING=の値は、出力ファイルのエンコーディングが、現在のセッションのエンコーディングとは異なることを示します。

出力ファイルにデータを書き込むときは、セッションのエンコーディングから指定されたエンコーディングへ、SAS によってデータがトランスコードされます。

有効なエンコーディング値については、"Overview to SAS Language Elements That Use Encoding Values" (*SAS National Language Support (NLS): Reference Guide* 22 章)を参照してください。

portable-options
FILE コマンドのオプションであり、あらゆる動作環境で有効です。このオプションの詳細については、*SAS システムオプション: リファレンス*を参照してください。

host-options
UNIX 環境に固有です。これらのオプションでは次のいずれもが使用できます。

BLKSIZE=BLK=
1 回の I/O 操作で物理的に書き込まれるバイト数を指定します。デフォルト値は 8K です。最大値は 1G–1 です。

LRECL=
論理レコード長を指定します。この値は、有効なレコード形式によって異なります(RECFM)。SAS 9.4 では、LRECL=のデフォルト値は 32,767 です。固定長レコード(RECFM=F)を使用する場合、LRECL=のデフォルト値は 256 になります。最大レコード長は 1G です。

- RECFM=F の場合、LRECL=オプションの値により各出力レコードの長さが決定されます。出力レコードは、切り捨てられるか、あるいは特定のサイズに合うようにスペースで埋め込まれます。
- RECFM=N の場合、LRECL=オプションの値は 256 以上にする必要があります。
- RECFM=V の場合、LRECL=オプションの値により最大レコード長が決定されます。指定された長さを超えるレコードは切り捨てられます。

NEW | OLD
新規ファイルが出力のために開かれることを示しています。対象ファイルがすでに存在している場合は、削除されて再作成されます。これはデフォルトのアクションです。

RECFM=
レコード形式を指定します。次に示すのは、RECFM=オプションの値です。

D	デフォルトのレコード形式(可変長形式)。
F	固定長形式。つまり、各レコードの長さは同じです。キャリッジコントロール文字が含まれる外部ファイルには、RECFM=F を使用しないでください。
N	バイナリ形式。ファイルはバイトストリームで構成され、レコード境界はありません。
P	プリント形式。SAS で、キャリッジコントロール文字が書き込まれます。
V	可変長形式。各レコードは改行文字で終わります。
S370V	可変長 S370 レコード形式(V)。
S370VB	可変長ブロック S370 レコード形式(VB)。

S370VBS　スパンレコード可変長ブロック S370 レコード形式(VBS)。

UNBUF
SAS に対し、その後の FILE ステートメントでファイルへのバッファ付き書き込みを実行しないよう伝えます。このオプションは特に、データ収集デバイスに書き込んでいるときに適用されます。

詳細

ファイル仕様を入力しない場合は、SAS は以前の FILE コマンドまたは INCLUDE コマンドからファイル名を使用します。この場合、まず SAS からこのファイルを上書きするかどうか尋ねられます。FILE コマンドも INCLUDE コマンドも発行していない場合は、デフォルトのファイルが存在しないことを示すエラーメッセージが表示されます。

FILL コマンド: UNIX

FILL を指定します。

UNIX 固有: デフォルト文字

構文

FILL <*fill-character*>

オプション引数

fill-character
1 つの行への入力に使用される文字を指定します。

UNIX では、デフォルトの FILL は下線です(_)。

FONTLIST コマンド: UNIX

Select Font ウィンドウを開きます。このウィンドウには使用可能なソフトウェアフォントが一覧表示されます。

UNIX 固有: すべて

構文

FONTLIST

詳細

FONTLIST コマンドにより、各自の動作環境で使用可能なソフトウェアフォントすべてを一覧表示するウィンドウが開きます。この機能は、FONT=オプションや FTEXT=オプションなど、SAS プログラムで使用するフォントを選択する場合に役立ちます。

FONTLIST コマンドを SAS コマンド行から発行すると、**Select Font** ウィンドウが開きます。このウィンドウには **Copy** ボタンと **System** ボタンという 2 つのボタンが含まれています。**System** をクリックすると、**Fonts** ウィンドウが開きます。使用可能なシステムフォントの選択とプレビューは、このウィンドウから行います。好きなフォントとフォントの属性を選択したら、**OK** をクリックしてください。**Select Font** ウィンドウが再度開いて、選択したフォント名が表示されます。**Copy** をクリックすると、コピーバッファにフォント名

が配置され、選択したフォント名を各自の SAS プログラムに貼り付けられるようになります。

GSUBMIT コマンド: UNIX

ペーストバッファに保存されている SAS コードを送信します。

UNIX 固有: 有効なバッファ名

構文

GSUBMIT BUF=*buffer-name* | "*statement-1*; *statement-n*...;"

必須引数

buffer-name
XPRIMARY、XSCNDARY、XCLIPBRD、XTERM、XCUT*n* のいずれかを指定できます。ここでは、0<=*n*<=7 です。詳細については、"UNIX 環境での切り取りと貼り付けのカスタマイズ" (209 ページ)を参照してください。

statement
SAS ステートメントです。

HOME コマンド: UNIX

カーソル位置を、カーソル現在位置とコマンド行に切り替えます。

UNIX 固有: キーボード版

構文

HOME

詳細

キーボードは UNIX 動作環境により異なります。HOME コマンドに割り当てるキーを決定するには、**Keys** ウィンドウにて確認してください。**Keys** ウィンドウを開くには、KEYS コマンドを発行してください。

関連項目:

"UNIX 環境でキー定義をカスタマイズする" (190 ページ)

HOSTEDIT コマンド: UNIX

EDITCMD システムオプションで指定した UNIX エディタを現在のウィンドウで起動します。

別名: HED

UNIX 固有: すべて

構文

HOSTEDIT

詳細

SAS テキストエディタウィンドウから HOSTEDIT コマンドを発行すると、そのウィンドウのバッファのコンテンツが/tmp ディレクトリの一時ファイルに書き込まれます。EDITCMD で指定したホストエディタを呼び出すコマンドが、SAS Session Manager に渡されます。SAS Session Manager により、このコマンドが動作環境に発行されて、一時ファイルのエディタが呼び出されます。

HOSTEDIT コマンドで使用される X ディスプレイは、各自の SAS セッションで使用されるものと同じです。

関連項目:

システムオプション:

- “EDITCMD システムオプション: UNIX” (375 ページ)

その他の参照資料:

- “UNIX 環境でホストエディタがサポートされるように SAS を構成する” (167 ページ)

INCLUDE コマンド: UNIX

外部ファイルのコンテンツ全体を、現在のウィンドウにコピーします。

UNIX 固有: 次の項目の有効な値: *encoding-value* および *portable-options*

構文

INCLUDE <*file-specification*> <ENCODING='*encoding-value*'> <*portable-options*> <*host-options*>

オプション引数

file-specification
次のいずれかになります。

- 単独のファイル名。SAS によりカレントディレクトリ内でファイルが検索されます。ファイル名を引用符で囲む場合、SAS では指定どおりのファイル名が使用されます。ファイル名を引用符で囲まず、ファイル名の拡張子も指定しない場合、SAS では.sas.が検索されます。
- パス名全体。パス名を引用符で囲まない場合でも、SAS はファイル名の拡張子を一切予測しません。
- ファイル参照名

ENCODING='*encoding-value*'
外部ファイルからの読み取り時に使用するエンコーディングを指定します。ENCODING=の値は、外部ファイルのエンコーディングが、現在のセッションのエンコーディングとは異なることを示します。

外部ファイルのデータを読み取るときは、指定されたエンコーディングからセッションのエンコーディングへ、SAS によってデータがトランスコードされます。

有効なエンコーディング値については、“Overview to SAS Language Elements That Use Encoding Values” (*SAS National Language Support (NLS): Reference Guide* 22 章)を参照してください。

portable-options
INCLUDE コマンドのオプションであり、あらゆる動作環境で有効です。これらのオプションについては、*SAS システムオプション: リファレンス*を参照してください。

host-options
UNIX 環境に固有です。これらのオプションでは次のいずれもが使用できます。

BLKSIZE=
BLK=
1 回の I/O 操作で物理的に読み込まれるバイト数を指定します。デフォルト値は 8K です。最大値は 1G–1 です。

LRECL=
論理レコード長を指定します。この値は、有効なレコード形式によって異なります(RECFM)。SAS 9.4 では、LRECL=のデフォルト値は 32,767 です。固定長レコード(RECFM=F)を使用する場合、LRECL=のデフォルト値は 256 になります。最大レコード長は 1G です。

- RECFM=F の場合、LRECL=オプションの値により、1 レコードとして読み取られるバイト数が決定されます。
- RECFM=N の場合、LRECL=オプションの値は 256 以上にする必要があります。
- RECFM=V の場合、LRECL=オプションの値により最大レコード長が決定されます。指定された長さを超えるレコードは切り捨てられます。

RECFM=
レコード形式を指定します。次に示すのは、RECFM=オプションの値です。

D デフォルトのレコード形式(可変長形式)。

F 固定長形式。つまり、各レコードの長さは同じです。

N バイナリ形式。ファイルはバイトストリームで構成され、レコード境界はありません。

P プリント形式。

V 可変長形式。各レコードは改行文字で終わります。

詳細

そのファイル名にファイル仕様を入力しない場合は、SAS は以前の FILE コマンドまたは INCLUDE コマンドからファイル名を使用します。この場合、まず SAS からこのファイルを上書きするかどうか尋ねられます。FILE コマンドも INCLUDE コマンドも発行していない場合は、デフォルトのファイルが存在しないことを示すエラーメッセージが表示されます。

SETAUTOSAVE コマンド: UNIX

自動保存のオンとオフを切り替えます。

UNIX 固有: すべて

構文

SETAUTOSAVE <ON | OFF>

詳細

SETAUTOSAVE コマンドは、**Program Editor** に対して自動保存のオンとオフを切り替えます。ただし、**Preferences** ダイアログボックスの自動保存の数値設定は優先されます。**Preferences** ダイアログボックスを開くには、**Tools** ⇨ **Options** ⇨ **Preferences** を選択します。自動保存の管理は、**DMS** タブの **Backup Documents** チェックボックスで行います。このタブには、これらバックアップの時間間隔を指定できるフィールドもあります。

SETAUTOSAVE コマンドを使用して自動保存を有効にし、**Backup Documents** チェックボックスを選択すると、SAS により、**DMS** タブで指定した間隔で、カレントディレクトリ内の `pgm.asv` というファイルに Program Editor のコンテンツが自動保存されます。

このコマンドを発行するが、ON や OFF を指定しない場合、SAS では現在の自動保存設定が表示されます。

関連項目:

- “UNIX 環境の各種リソース” (214 ページ)
- “DMS 設定の変更” (174 ページ)

SETDMSFONT コマンド: UNIX

現在のセッションに対し、ウィンドウ環境フォントを指定します。

UNIX 固有: すべて

構文

SETDMSFONT "*font-specification*"

必須引数

font-specification
ウィンドウ環境を決定する場合に SAS で使用する X Logical Font Description (XLFD)パターンを指定します。

詳細

X Window System の大部分のフォントは、XLFD に関連付けられており、XLFD は、ダッシュ(-)文字で区切られたさまざまなフィールドを数多く含んでいます。XLFD 内のフィールドにより、フォントファミリ名、太さ、サイズ、解像度といったプロパティと、そのフォントがプロポーショナルなのか等幅なのかがわかります。X Window System で使用される XLFD およびフォント名の詳細については、X Window System ドキュメントを参照してください。

関連項目:

コマンド:

- “DLGFONT コマンド: UNIX” (233 ページ)

SETENV コマンド: UNIX

環境変数を定義し、値を割り当てます。

構文

SETENV <*variable-name*> <*variable-value*>

UNSETENV *variable-name*

必須引数

variable-name
設定可能な UNIX 環境変数を指定します。UNSETENV コマンドを使用する場合は、この値が必須です。

オプション引数

variable-value
UNIX 環境変数の値を指定します。

詳細

SETENV コマンドを使用すると、環境変数を定義し、その変数に値を割り当てることができます。環境変数の値は、autoexec 処理中に SYSGET 関数を使用して、SAS セッション内から取得できます。(コマンド `x setenv a/tmp;`による設定: `a=/tmp`、コマンド `x echo $a;`による結果値: `/tmp`)

UNSETENV コマンドにより、環境変数が削除されます。エントリと環境変数のメモリが解放されます。

TOOLCLOSE コマンド: UNIX

ツールボックスを閉じます。

UNIX 固有: すべて

構文

TOOLCLOSE

詳細

TOOLCLOSE コマンドはツールボックスを閉じます。

関連項目:

コマンド:

- “TOOLLOAD コマンド: UNIX” (245 ページ)

TOOLEDIT コマンド: UNIX

指定のツールボックスの Tool Editor を開きます。

UNIX 固有: すべて

構文

TOOLEDIT <*library.catalog.entry*>

詳細

エントリ名を指定しない場合は、Tool Editor で、アクティブウィンドウのツールボックスを編集します。

TOOLLARGE コマンド: UNIX

SAS ToolBox ウィンドウのサイズを切り替えます。

UNIX 固有: すべて

構文

TOOLLARGE <ON | OFF>

必須引数

ON
SAS ToolBox のアイコンのサイズを 48x48 に設定します。

OFF
SAS ToolBox のアイコンのサイズを 24x24 に設定します。

詳細

ON も OFF も指定しない場合は、TOOLLARGE コマンドにより SAS ToolBox のサイズが切り替わります。SAS ToolBox のサイズの変更は現在のセッションに対してのみ有効となり、新しいサイズは保存されません。

また、メニューを使って **Preferences** ダイアログボックスから SAS ToolBox のサイズを変更することも可能です。**Tools** ⇨ **Options** ⇨ **Preferences** を選択します。**ToolBox** タブを選択し、**Use large tools** を選択してください。**Preferences** ダイアログボックスで SAS ToolBox のサイズを変更する場合は、新しいサイズが保存され、その後のセッションでは SAS のツールボックスが大きく表示されます。

TOOLLOAD コマンド: UNIX

特定のツールボックスをロードします。

UNIX 固有: すべて

構文

TOOLLOAD <*library.catalog.entry*>

詳細

エントリ名を指定しない場合は、TOOLLOAD によりアクティブウィンドウに対してツールボックスがロードされます。

関連項目:

コマンド:

- “TOOLCLOSE コマンド: UNIX” (244 ページ)

TOOLTIPS コマンド: UNIX

ツールボックスのアイコンの ToolTip テキストを切り替えます。

UNIX 固有: すべて

構文

TOOLTIPS <ON | OFF>

必須引数

ON
ツールボックスのアイコンにカーソルを移動させたとき、ToolTip テキストが表示されるよう指定します。

OFF
ToolTip テキストを非表示に指定します。

詳細

ON も OFF も指定しない場合は、現在の設定によって、TOOLTIPS コマンドが ToolTip テキストの表示をオンまたはオフにします。

また、**Tools** ⇨ **Options** ⇨ **Preferences** を選択し、**Preferences** ダイアログボックスを使用して、ToolTip テキストを表示するかどうかを指定することもできます。**ToolBox** タブを選択し、**Use tip text** を選択します。

関連項目:

“既存のツールの属性の変更” (186 ページ)

WBROWSE コマンド: UNIX

World Wide Web (WWW)ブラウザを開きます。

UNIX 固有: すべて

構文

WBROWSE <"*url*">

詳細

WBROWSE により、リソース **SAS.helpBrowser** で指定するウェブブラウザが起動します。URL を指定すると、その URL が認識するドキュメントが自動表示されます。URL を指定しない場合は、SAS のホームページが表示されます。

関連項目:

"UNIX 環境の各種リソース" (214 ページ)

WCOPY コマンド: UNIX

アクティブウィンドウ内のマークしたコンテンツをデフォルトバッファにコピーします。

UNIX 固有: すべて

構文

WCOPY

詳細

Base SAS ウィンドウでは、このコマンドは STORE コマンドを実行します。STORE コマンドについては、オンラインの SAS ヘルプおよび SAS ドキュメントを参照してください。

WCUT コマンド: UNIX

アクティブウィンドウ内のマークしたコンテンツをデフォルトバッファに移動します。.

UNIX 固有: すべて

構文

WCUT

詳細

Base SAS ウィンドウでは、このコマンドは CUT コマンドを実行します。

このコマンドが有効なのは、**Program Editor** や **NOTEPAD** などのテキストエディタウィンドウがアクティブウィンドウである場合のみです。

CUT コマンドについては、オンラインの SAS ヘルプと SAS ドキュメントを参照してください。

WDEF コマンド: UNIX

アクティブウィンドウを再定義します。

UNIX 固有: 動作管理は SAS.awsResizePolicy リソースによる

構文

WDEF *starting-row starting-column number-rows number-columns*

詳細

WDEF コマンドは、SAS セッションに割り当てられているアプリケーションワークスペースで動作します。AWS コンテナウィンドウでは WDEF は動作しませんが、AWS コンテナウィンドウに含まれている SAS ウィンドウが見えるように AWS コンテナウィンドウを拡大する必要がある場合は WDEF は動作します。AWS のサイズ変更動作は **SAS.awsResizePolicy** リソースで管理します。

関連項目:

- “UNIX 環境の各種リソース” (214 ページ)
- “X Window マネージャ” (147 ページ)

WPASTE コマンド: UNIX

デフォルトバッファのコンテンツをアクティブウィンドウに貼り付けます。

UNIX 固有: すべて

構文

WPASTE

詳細

Base SAS ウィンドウでは、このコマンドは PASTE コマンドを実行します。PASTE コマンドについては、オンラインの SAS ヘルプおよび SAS ドキュメントを参照してください。

WUNDO コマンド: UNIX

テキストエントリ行を元に戻す、または最後に行われた切り取り動作、コピー動作、貼り付け動作を取り消します。

UNIX 固有: すべて

構文

WUNDO

詳細

Base SAS ウィンドウでは、このコマンドは UNDO コマンドを実行します。SAS/GRAPH ウィンドウでは、WUNDO は無効です。

WUNDO コマンドの 1 回の実行で同時に元に戻されるテキストエントリは 1 行のみです。WUNDO コマンドを再度実行すると、以前のテキスト行が元に戻ります。

CC コマンドを使ってテキストブロックをコピーして貼り付けてから WUNDO コマンドを発行すると、コピーし貼り付けしたテキストブロックが削除されます。DD コマンドを使ってテキストブロックを削除してから WUNDO コマンドを発行すると、削除したテキストブロックが復元されます。

注: WUNDO コマンドは SUBMIT コマンドが削除する行を置換できません。サブミットした SAS ステートメントの影響を元には戻せません。

X コマンド: UNIX

SAS セッションを終了しないで UNIX コマンドを入力できます。

UNIX 固有: すべて

構文

X *command*

X '*command-1*; *command-2*....<; *command-n*>'

必須引数

command
UNIX コマンドを指定します。

詳細

X コマンドを入力すると、SAS ではシェルが起動して、指示したコマンドを実行します。入力するコマンドの処理は、入力するコマンドが 1 つなのか 2 つ以上なのかによって異なります。

関連項目:

"SAS セッションからオペレーティングシステムコマンドの実行" (15 ページ)

XSYNC コマンド: UNIX

SAS セッション中に X 同期を変更します。

UNIX 固有: すべて

構文

XSYNC <ON | OFF>

詳細

このコマンドは、通常は X Window System により実行されるバッファリングを無効にします。デフォルトでは X 同期は無効になっています。パフォーマンスは大きく低下しますが、アプリケーションのデバッグ中に X 同期を有効にすると有用です。

ON や OFF を指定しない場合は、XSYNC により X 同期が切り替わります。XSYNC コマンドはあらゆる SAS ウィンドウで有効です。

12 章

UNIX 版に固有のデータセットオプション

UNIX 版に固有の SAS データセットオプション

このセクションでは UNIX 環境にのみ存在する SAS データセットオプションおよびその動作または構文が UNIX 固有のオプションについて説明します。各データセットオプションの説明はデータセットオプションのどの点が UNIX 固有かを説明する“UNIX 固有の点”を含みます。UNIX に特有な動作や構文を持つデータセットオプションについては、*SAS データセットオプション: リファレンス*のオプションの完全な記述を参照してください。

SAS ステートメントのデータセット名に続くデータセットオプションを次にように指定します。

...data-set-name(*option-1=value-1 option-2=value-2*, *…*)

いくつかのデータセットオプションは SAS システムオプションでもあります(例えば BUFSIZE=)。同じオプションがシステムオプションとデータセットオプションの両方として指定されている場合、SAS はデータセットオプションにて与えられた値を使用します。SAS システムオプションの詳細については、“システムオプションを使用し、SAS セッションをカスタマイズする” (18 ページ) および “UNIX 版に固有の SAS システムオプション” (360 ページ)を参照してください。

UNIX にて利用できる全てのデータセットオプションのテーブルを表示するには、“UNIX 環境で使用される SAS データセットオプションのまとめ” (252 ページ)を参照してください。

UNIX 環境で使用される SAS データセットオプションのまとめ

SAS データセットオプションは次のテーブルにてリストアップされています。このテーブルは各オプション名、簡単な説明、開かれたデータセットが入力、出力、または更新に使用できるかどうか、そしてオプションが有効なエンジンをリストします。**See** 列はオプションの詳細情報の場所を示します。次の凡例を使って追加情報を見つけます。

COMP
このセクションのデータセットオプションの記述を参照してください。

DS
*SAS データセットオプション: リファレンス*を参照してください。

NLS
*SAS 各国語サポート(NLS): リファレンスガイド*を参照してください。

表 12.1 *SAS データセットオプションの概要*

オプション名	説明	使用場所	エンジン	参照
ALTER=	SAS ファイルにパスワードを指定して、ユーザーが SAS ファイルを差し替えたり消去したりするのを防ぎますが、Read と Write のアクセスは許可します。	出力、更新	V9, V8, V6	DS, COMP
BUFNO=	SAS データセットをプロセスするのに割り当てるバッファの数を指定します。	入力、出力、更新	V9, V8, V6	DS, COMP
BUFSIZE=	出力 SAS データセットのための永久バッファページの大きさを指定します。	アウトプット	V9, V8	DS, COMP
CNTLLEV=	SAS データセットへの共有アクセスのレベルを指定します。	入力、更新	V9, V8	DS
COMPRESS=	新しい出力 SAS データセットでのオブザベーションの圧縮を設定します。	アウトプット	V9, V8, V6	DS
DLDMGACTION=	SAS ライブラリの SAS データセットが破損していると検知された場合にどのような処理をするかを指定します。	入力、出力、更新	V9, V8	DS
DROP=	入力データセットでは指定した変数をプロセスから除去します。出力データセットでは、指定された変数をデータセットへ書き込みから除去します。	入力、出力、更新	すべて	DS
ENCODING=	入力または出力 SAS データセットのエンコードを上書きします。	入力、出力	V9, V8	NLS
ENCRYPT=	出力 SAS データセットを暗号化するかどうかを指定します。	アウトプット	すべて	DS

オプション名	説明	使用場所	エンジン	参照
FIRSTOBS=	SAS データセットにて最初に SAS がプロセスするオブザベーションを指定します。	入力、更新	すべて	DS
GENMAX=	SAS データセットの作成をリクエストし、バージョンの最大数を指定します。	出力、更新	V9, V8	DS
GENNUM=	特定の SAS データセットの作成を指定します。	入力、出力、更新	V9, V8	DS
IDXNAME=	SAS に対して WHERE 表現の条件を満たすため、特定のインデックスを使用するよう指示します。	入力、更新	V9, V8, V6	DS
IDXWHERE=	WHERE 表現の条件を満たすため、SAS に対してインデックスまたは連続検索を使用するかどうかを指定します。	入力、更新	V9, V8, V6	DS
IN=	論理変数を作成し、データセットが現在のオブザベーションへ貢献データかどうかを指示します。	入力、更新	すべて	DS
INDEX=	新しい出力 SAS データセットのインデックスを定義します。	アウトプット	V9, V8, V6	DS
KEEP=	入力データセットに対してプロセスする変数を指定します。出力データセットに対してデータセットに書き込む変数を指定します。	入力、出力、更新	すべて	DS
LABEL=	SAS データセットのラベルを指定します。	入力、出力、更新	すべて	DS
OBS=	データセットにて SAS がプロセスする最後のオブザベーションを指定します。	入力、更新	すべて	DS
OBSBUF=	DATA ステップビューのプロセスのためのビューバッファの大きさを決定します。	入力	V9, V8	DS
OUTREP=	出力 SAS データセットのためのデータ表現を指定します。	アウトプット	V9, V8	DS, NLS
POINTOBS=	圧縮した SAS データをオブザベーション番号または連続アクセスのどちらでプロセスするかを設定します。	アウトプット	V9, V8	DS
PW=	SAS ファイルに READ、WRITE、または ALTER のパスワードを割り当てて、パスワードで保護されたファイルのアクセスを有効にします。	入力、出力、更新	V9, V8, V6	DS, COMP
PWREQ=	SAS データセットパスワードのダイアログボックスを表示するかどうか指定します。	入力、出力、更新	V9, V8, V6	DS

オプション名	説明	使用場所	エンジン	参照
READ=	SAS ファイルにパスワードを割り当て、読み取り保護された SAS ファイルへのアクセスを有効にします。	入力、出力、更新	V9, V8, V6	DS
RENAME=	変数名を変更します。	入力、出力、更新	すべて	DS
REPEMPTY=	新しい空のデータセットが同じ名前を持つ既存の SAS データセットを上書き可能かどうかを指定します。	アウトプット	V9, V8	DS
REPLACE=	新しい SAS データセットが同じ名前を持つ既存の SAS データセットを上書き可能かどうかを指定します。	アウトプット	すべて	DS
REUSE=	圧縮された SAS データセットの空き容量に新しいオブザベーションが書き込み可能かどうかを指定します。	アウトプット	V9, V8, V6	DS
SORTEDBY=	SAS データセットの現在の保存方法を示します。	入力、出力、更新	V9, V8, V6	DS
SPILL=	DATA ステップビューの非連続プロセスのために予備ファイルを作成するかどうかを指定します。	アウトプット	V9, V8	DS
TOBSNO= (REMOTE エンジンの方法にて SAS Server を通してアクセスしたデータセットのみに有効です)。	クライアント/サーバー転送にて送るオブザベーションの数を指定します。	入力、出力、更新	V9, V8	DS
TYPE=	特別に構築された SAS データセットのためのデータセットタイプを指定します。	入力、出力、更新	すべて	DS
USEDIRECTIO=	指定したファイルのダイレクトファイル I/O をオンにします。このデータセットオプションを使用するには、ライブラリ参照名が割り当てられた LIBNAME ステートメントにて ENABLEDIRECTIO ステートメントオプションを指定する必要があります。	入力、出力、更新	V9, V8	COMP
WHERE=	指定された条件に合う SAS データセット内のオブザベーションを選択します。	入力、出力、更新	すべて	DS
WHEREUP=	WHERE 表現と照らし合わせて新しいオブザベーションおよび更新されたオブザベーションを評価するかどうかを指定します。	出力、更新	V9, V8, V6	DS
WRITE=	SAS ファイルに WRITE パスワードを割り当て、書き込み保護された SAS ファイルへのアクセスを有効にします。	出力、更新	V9, V8, V6	DS

注: TOBSNO= オプションは REMOTE エンジンから SAS Server を通してアクセスしたデータセットのみに有効です。

ディクショナリ

ALTER= データセットオプション: UNIX

SAS ファイルにパスワードを指定して、ユーザーが SAS ファイルを差し替えたり消去したりするのを防ぎますが、Read と Write のアクセスは許可します。

該当要素: DATA ステップおよび PROC ステップ

カテゴリ: データセット設定

デフォルト: なし

エンジン: V9、V8、V6

参照項目: “ALTER= Data Set Option” (*SAS Data Set Options: Reference*)

構文

ALTER=*alter-password*

必須引数

alter-password
有効な SAS 名である必要があります。“Words in the SAS Language” (*SAS Language Reference: Concepts* 3 章)を参照してください。

詳細

ALTER= オプションはカタログ以外の全てのタイプの SAS ファイルに適用されます。このオプションを使用して、SAS ファイル自体、または読み取り保護ファイル、書き込み保護ファイル、変更保護 SAS ファイルへのアクセスに *alter-password* を割り当てることが可能です。

BUFNO= データセットオプション: UNIX

SAS データセットの処理用に割り当てるバッファ数を指定します。

該当要素: DATA ステップおよび PROC ステップ

カテゴリ: データセット設定

デフォルト: 1

エンジン: V9, V8, V6

UNIX 固有: デフォルト値

参照項目: “BUFNO= Data Set Option” (*SAS Data Set Options: Reference*)

構文

BUFNO=*n* | *n*K | *hex*X | MIN | MAX

必須引数

***n* | *n*K**
バッファの数を 1 (バイト)や 1,024 (キロバイト)の倍数で指定します。例えば、値 `8` は 8 バッファを指定し、値 `1k` は 1024 バッファを指定します。

***hex*X**
バッファ数を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、値が `2dx` の場合は 45 バッファが指定されます。

MIN
バッファの最小数を0に設定し、操作環境にて SAS が最小最適値を使用できるようにします。

MAX
操作環境での最大限の数字をバッファ数として指定します。数字は最大の 4 バイト符号付整数、つまり 2^{31}–1 または 20 億です。

詳細

バッファ数はデータセットの永久属性ではなく、現在の SAS ステップにのみ有効です。BUFNO=は、入力、出力または更新用に開かれた SAS データセットに適用されます。

関連項目:

データセットオプション:

- “ BUFSIZE= データセットオプション: UNIX” (256 ページ)

システムオプション:

- “BUFNO システムオプション: UNIX” (368 ページ)

BUFSIZE= データセットオプション: UNIX

出力 SAS データセットの永久バッファページのサイズを指定します。

該当要素: DATA ステップおよび PROC ステップ
カテゴリ: データセット設定
デフォルト: 0
エンジン: V9, V8
UNIX 固有: 有効範囲
参照項目: “BUFSIZE= Data Set Option” (*SAS Data Set Options: Reference*)

構文

BUFSIZE=*n* | *n*K | *n*M | *n*G | *hex*X | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
バッファの大きさを 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、または 1,073,741,824 (ギガバイト)の倍数で指定します。たとえば、値 `8` は 8 バイトを指定し、値 `3m` は 3,145,728 バイトを指定します。

バッファの大きさの範囲は 1K から 2G–1 です。1G より大きな値には、*n*M オプションを使用します。

***hex*X**
16 進数値にてページの大きさを指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、`2dx` の場合はページサイズが 45 バイトに設定されます。

MAX
操作環境での最大限の数字をバッファ数として指定します。数字は最大の 4 バイト符号付整数、つまり 2^{31}–1 またはおよそ 20 億バイトです。

詳細

BUFSIZE= データセットオプションは作成するデータセットのバッファの大きさを指定します。このオプションは　出力データセットのみに有効です。

SAS データセットを作成するときにデフォルト値 (0) を使用した場合、エンジンは CPU と I/O 使用に最適なバッファの大きさを計算します。大きさは 80 のオブザベーションを保持できる 8K の最小倍数ですが、64K より大きくはなりません。

SAS データセットを作成する時にゼロ以外の値を指定した場合、エンジンはその値を使用します。その値が最低1つのオブザベーションを保持できない場合や有効なバッファの大きさでない場合、エンジンは 1K の倍数の値に端数を切り上げます。

関連項目:

システムオプション:

- “BUFSIZE システムオプション: UNIX” (369 ページ)

PW= データセットオプション: UNIX

SAS ファイルに READ、WRITE、または ALTER のパスワードを割り当てて、パスワードで保護されたファイルのアクセスを有効にします。

該当要素: DATA ステップおよび PROC ステップ

カテゴリ: データセット設定

デフォルト: なし

エンジン: V9、V8、V6

参照項目: “PW= Data Set Option” (*SAS Data Set Options: Reference*)

構文

PW=*パスワード*

必須引数

パスワード
有効な SAS 名である必要があります。“Words in the SAS Language” (*SAS Language Reference: Concepts* 3 章)を参照してください。

詳細

PW= オプションはカタログ以外の全てのタイプの SAS ファイルに適用されます。このオプションを使用して SAS ファイルにパスワードを割り当てたり、パスワードで保護された SAS ファイルにアクセスすることができます。

USEDIRECTIO= データセットオプション: UNIX

ENABLEDIRECTIO オプションが適用されたファイルを含むライブラリのダイレクトファイル I/O をオンにします。

該当要素: DATA ステップ

カテゴリ: データセット設定

デフォルト: オフ

エンジン: V9, V8

UNIX 固有: このオプションを使用するには、ライブラリ参照名が割り当てられた LIBNAME ステートメントにて ENABLEDIRECTIO オプションを使用する必要があります。

構文

USEDIRECTIO=YES | NO

必須引数

YES | NO
USEDIRECTIO=オプションをオンにするかどうかを指定します。

詳細

USEDIRECTIO=データセットオプションは、DATA ステートメントにリストされたデータセットのダイレクトファイル I/O をオンにします。関連するライブラリ参照名は LIBNAME ステートメントの ENABLEDIRECTIO オプションにて定義されている必要があります。

LIBNAME ステートメントで ENABLEDIRECTIO を使用すると、そのライブラリのデータセットに対するダイレクトファイル I/O が可能になります。ダイレクト I/O 自体はオンになりません。USEDIRECTIO=オプションを使用してダイレクトファイル I/O を作成する必要があります。

ダイレクトファイル I/O をオンにする方法が2つあります。

- ENABLEDIRECTIO および USEDIRECTIO= オプションの両方を LIBNAME ステートメントにて使用します。

  ```
  libname libref-name '.' ENABLEDIRECTIO USEDIRECTIO=yes;
  ```

  この場合、SAS はライブラリ参照名 *libref-name* を使用して開いたすべての SAS I/O データセットでダイレクトファイル I/O を使用します。

- LIBNAME ステートメントで ENABLEDIRECTIO を使用し、DATA ステートメントで USEDIRECTIO=を使用します。

```
libname libref-name '.' ENABLEDIRECTIO;
data libref-name.data-set-name (USEDIRECTIO=yes);
```

この場合、*libref-name.data-set-name* はダイレクトファイル I/O 用に開かれます。*libref-name* で参照した他の SAS I/O データセットはダイレクトファイル I/O は使用しません。

USEDIRECTIO= 自体は影響を与えません。次のステートメントのどちらもダイレクトファイル I/O 用にデータセットを開きません。

```
libname libref-name '.' USEDIRECTIO=yes;
data libref-name.data-set-name (USEDIRECTIO=yes);
```

例

次の例では、ENABLEDIRECTIO LIBNAME オプションを使用し、ライブラリ参照名 **test** に関連付けられたファイルをダイレクト I/O のために開けるようにします。USEDIRECTIO= データセットオプションはダイレクト I/O のために **test.file1** を開きます。**test.file2** はダイレクト I/O のためには開かれません。

```
LIBNAME test'.'ENABLEDIRECTIO;
data test.file1(USEDIRECTIO=yes);
... more SAS statements ...
run;
data test.file2;
... more SAS statements ...
run;
```

関連項目:

ステートメント:

- “LIBNAME ステートメント: UNIX” (344 ページ)

13 章

UNIX 版に固有の出力形式

UNIX 版に固有の SAS 出力形式

このセクションは UNIX 環境に固有の動作や構文を持つ SAS 出力形式を説明します。各出力形式の説明には、そのデータセットのどの要素が UNIX 版に固有であるのかを簡単に説明する"UNIX 固有"セクションが含まれています。各出力形式はこのドキュメントと *SAS 出力形式と入力形式: リファレンス*に説明されています。

ディクショナリ

HEX*w.* 出力形式: UNIX

バイナリ実数(浮動小数点)数を 16 進表現に変換します。

カテゴリ: 数値

配置: 左

デフォルト: 8

範囲: 1 から 16

UNIX 固有: 浮動小数点表現

参照項目: “HEXw. Format” (*SAS Formats and Informats: Reference*)

詳細

HEX*w.* 出力形式は実数バイナリ(浮動小数点)を 16 進表現に変換します。幅の値 1 から 15 を指定したときに、バイナリ実数は 16 進に変換される前は固定小数点整数に切り捨てられます。16 を幅と指定したときに、SAS は数字の浮動小数点値を書き込みますが、切り捨てはしません。

注: UNIX システムは浮動小数点表現にて大きく変わります。詳細については、"UNIX 環境でのバイナリデータの読み込みと書き込み" (224 ページ)を参照してください。

$HEX*w.* 出力形式: UNIX

文字値から 16 進表現への変換をします。

カテゴリ: 文字

配置: 左

デフォルト: 2

範囲: 1 から 32767

UNIX 固有: ASCII コードを生成します。

参照項目: "$HEXw. Format" (*SAS Formats and Informats: Reference*)

詳細

UNIX では、$HEX*w.* 出力形式は文字の ASCII コードを 16 進表現で生成します。各バイトで 2 カラムが必要とされます。したがって、$HEX*w.*出力形式で値を出力するには 2 倍のカラムが必要です。

IB*w.d* 出力形式: UNIX

整数バイナリ(固定小数点)値を書きます。

カテゴリ: 数値

配置: 左

デフォルト: 4

範囲: 1 から 8, 0–10

UNIX 固有: バイトオーダー

参照項目: "IBw.d Format" (*SAS Formats and Informats: Reference*)

詳細

IB*w.d* 出力形式は整数バイナリ(固定小数点)値を書きます。整数は整数バイナリまたは固定小数点形式にて保存されます。たとえば、数字の 2 は 00000002 と保存されます。出力形式が *d* という値を含む場合、データ値は 10^d 倍されます。

詳細については、"UNIX 環境でのバイナリデータの読み込みと書き込み" (224 ページ)を参照してください。

PD*w.d* 出力形式: UNIX

データをパック 10 進形式で書きます。

カテゴリ: 数値
配置: 左
デフォルト: 1
範囲: 1 から 16, 0–31
UNIX 固有: データ表現
参照項目: “PDw.d Format” (*SAS Formats and Informats: Reference*)

詳細

PD*w.d* 出力形式は値をパック 10 進形式で書きます。パック 10 進データでは、1 バイトに2桁が含まれます。*w* の値は桁の数ではなくバイトの数を表します。値の記号は最初のバイトです。最初のバイト全体が符号として使用されるため、バイト幅は 2 以上を指定してください。

PD*w.d* 出力形式は、欠損数値データを–0 として書き込みます。PD*w.d* 入力形式が–0 の値を読み取ると、結果は 0 の値になります。

詳細については、“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

PIB*w.d* 出力形式: UNIX

正の整数バイナリ(固定小数点)値を書きます。

カテゴリ: 数値
配置: 左
デフォルト: 1
範囲: 1 から 8, 0–10
UNIX 固有: バイトオーダー
参照項目: “PIBw.d Format” (*SAS Formats and Informats: Reference*)

詳細

PIB*w.d* 出力形式は、すべての値を正として処理し、固定小数点バイナリ値を書き込みます。したがって、高位ビットは値の記号ではなく値の一部です。*d* の値が指定された場合、データ値は 10^d 倍されます。

詳細については、“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

RB*w.d* 出力形式: UNIX

バイナリ実数形式で実数バイナリ(浮動小数点)データを書きます。

カテゴリ: 数値
配置: 左
デフォルト: 4
範囲: 2 から 8, 0–10
UNIX 固有: 浮動小数点表現
参照項目: “RBw.d Format” (*SAS Formats and Informats: Reference*)

詳細

RB*w.d* 出力形式は数字データをバイナリ実数(浮動小数点)表記で書きます。SAS はすべての数値を浮動小数点で保存します。

バイナリ実数は数字値を表現するための最も効率のよい形式です。なぜなら SAS はすでに数字をこの方法で表現しており、新たな表現方法を必要としないからです。

詳細については、“RBw.d 入力形式:UNIX” (292 ページ) および“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

ZD*w.d* 出力形式: UNIX

データをゾーン 10 進形式で書きます。

カテゴリ: 数値
配置: 左
デフォルト: 1
範囲: 1 から 32
UNIX 固有: データ表現
参照項目: “ZDw.d Format” (*SAS Formats and Informats: Reference*)

詳細

ZD*w.d* 出力形式はゾーン 10 進データを書きます。この出力形式は重ね打ち末尾数字形式として知られています。UNIX では、フィールドの最後のバイトは符号と最終桁を含みます。次に最後のバイトの変換表を示します。

表 13.1 *変換表*

数字	ASCII 文字	数字	ASCII 文字
0	{	–0	}
1	A	–1	J

2	B	–2	K
3	C	–3	L
4	D	–4	M
5	E	–5	N
6	F	–6	O
7	G	–7	P
8	H	–8	Q
9	I	–9	R

詳細については、“ZDw.d 入力形式: UNIX” (293 ページ) および“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

14 章

UNIX 版に固有の関数と CALL ルーチン

UNIX 版に固有の SAS 関数と CALL ルーチン

このセクションでは、動作が UNIX 環境に固有の SAS 関数と CALL ルーチンについて説明します。各関数と CALL ルーチンの説明には、その関数と CALL ルーチンのどの要素が UNIX 版に固有であるのかを簡単に説明する"UNIX 固有"セクションが含まれています。これらすべての関数と CALL ルーチンについての詳細は、*SAS 関数と CALL ルーチン: リファレンス*を参照してください。

ディクショナリ

BYTE 関数: UNIX

ASCII 照合シーケンスで 1 文字を返します。

カテゴリ: 文字

UNIX 固有: ASCII 照合シーケンスを使用

参照項目: “BYTE Function” (*SAS Functions and CALL Routines: Reference*)

構文

BYTE(*n*)

必須引数

n
特定の ASCII 文字を表す整数を指定します。*n* の値は 0 から 255 に及びます。

詳細

BYTE 関数が、値をまだ長さを割り当てられていない変数に返す場合、デフォルトでは、その変数は 1 の長さを割り当てられます。

CALL MODULE ルーチン: UNIX

実行可能な共有ライブラリに存在する特定のルーチンまたはモジュールを呼び出します。

カテゴリ: 外部ファイル

操作: SAS Server がロックダウン状態にある場合、CALL MODULE ルーチンは実行されません。詳細については、“SAS Processing Restrictions for Servers in a Locked-Down State” (*SAS Language Reference: Concepts* 2 章)を参照してください。

UNIX 固有: すべて

参照項目: “CALL MODULE Routine” (*SAS Functions and CALL Routines: Reference*)

構文

CALL MODULE(<*control*>, *module*, *argument-1*, *argument-2* …, *argument-n*);

number=**MODULEN**(<*control*>, *module*, *argument-1*, *argument-2* …, *argument-n*);

character=**MODULEC**(<*control*>, *module*, *argument-1* …, *argument-2*, *argument-n*);

CALL MODULEI(<*control*>, *module*, *argument-1*, *argument-2* …, *argument-n*);

number=**MODULEIN**(<*control*>, *module*, *argument-1*, *argument-2* …, *argument-n*);

character=**MODULEIC**(<*control*>, *module*, *argument-1*, *argument-2* …, *argument-n*);

必須引数

module

使用する外部モジュールの名前を指定します。*module* は、共有ライブラリとルーチン名またはカンマで区切られた序数の値として指定できます。SASCBTBL 属性テーブルでルーチンに MODULE 属性を指定していたなら、そのルーチンの名前が固有(すなわち、属性ファイル内に同じ名前を持つ他のルーチンがない)のものである限りは、共有ライブラリ名を指定する必要はありません。詳細については、“SASCBTBL 属性テーブル” (106 ページ)を参照してください。

モジュールは共有ライブラリ内に存在する必要があり、外部から呼び出せる必要があります。共有ライブラリ名は大文字と小文字を区別しませんが、ルーチン名は、そのルーチンの実装言語の制限に基づきます。ゆえにルーチン名は大文字と小文字を区別します。

共有ライブラリが序数値の命名をサポートしている場合、'XYZ,30'のように、10 進数が後に続く共有ライブラリ名を提供できます。

module を定数としてではなく、SAS 文字式として指定できます。ほとんどの場合、定数として渡されます。

argument-1, argument-2, ..., argument-n

要求されたルーチンに渡す引数を指定します。引数には適切な属性を使用してください。(すなわち、数値属性には数値引数を、文字属性には文字引数)

注意:

必ず、正しい引数と属性を使用してください。 共有ライブラリ関数に不正な引数や属性を使用すると、SAS のクラッシュが引き起こされたり、予測しない結果がもたらされたりする可能性があります。

オプション引数

control

任意の制御文字列で、最初の文字は必ずアスタリスク(*)になります。次の文字の組み合わせが後に続きます。

I 共有ライブラリルーチンが呼び出される前と後に、すべての引数を 16 進数表記で、MODULE 関数と要求された共有ライブラリルーチンに出力します。このオプションを使うことで、不正な引数または属性テーブルが原因の問題診断に役立ちます。I オプションを指定すると、E オプションがそこに含まれます。

E 詳細なエラーメッセージが出力されます。. E オプション(またはこれより優先される I オプション)がない場合、MODULE 関数が生成するエラーメッセージは"Invalid argument to function,"のみになります。この情報では、通常、エラーの原因を確定するには不十分です。

S*x* 区切り文字として *x* を使い、フィールド定義を分離します。それから、引数のリスト上の *x* を、それ自身の文字引数として指定できます。この引数は引数リストの区切り文字として機能します。このリスト上の引数は単一の構造体として集められます。SASCBTBL 属性テーブルでエントリを供給しない場合にのみ、このオプションを使用してください。SASCBTBL 属性テーブルで、このモジュールにエントリを供給する場合は、このテーブル内の ARG ステートメントの FDSTART オプションを使用し、構造定義を分離します。

H MODULE ルーチン、属性ファイル形式および推奨される SAS 出力/入力形式の構文についての、手短な役立つ情報を提供します。

たとえば、制御文字列'*IS/'では、パラメータリストを出力し、文字列'/'を引数リスト内で区切り文字として扱うことを指定します。

詳細

次の関数では、ベクトル引数と行列引数が許可されます。これらの関数は、IML プロシジャ内でのみ使用できます。

- CALL MODULEI
- MODULEIN
- MODULEIC

詳細については、*SAS/IML Studio: User's Guide* を参照してください。

MODULE 関数では、ルーチン *module* が実行されます。このルーチンは、*argument-1* から *argument-n* までの指定された引数とともに、外部(SAS の外側)共有ライブラリに存在します。

CALL MODULE ルーチンでは、値は返されません。MODULEN と MODULEC 関数では、それぞれ数値または文字値が返されます。どのルーチンを使用するかは、実行する共有ライブラリ関数の予測戻り値によって変わります。

MODULEI、MODULEIC および MODULEIN は、MODULE 関数の特殊なバージョンです。これらにより、ベクトル引数と行列引数が許可されます。これらの戻り値はスカラーのままです。これらの関数は PROC IML からのみ呼び出せます。

名前の違いをを除いて、すべての 6 つのルーチンの構文は同じものです。

MODULE 関数では、パラメータのリストを作成するために、*argument-1* から *argument-n* までの情報と、分割ファイル内で定義されたルーチンの記述および引数属性テーブルが使用されます。MODULE ルーチンを呼び出す前に、SASCBTBL のファイル参照名を定義して、この外部ファイルを示してください。このファイルを作成時に、任意の名前を付けることができます。

その場合、SAS 変数と出力形式を MODULE 関数への引数として使用し、これらの引数が共有ライブラリルーチンへ渡される前に正しく変換されるかを確認できます。

注意:

属性テーブルを定義せずに MODULE 関数を使用すると、SAS がクラッシュしたり、予想外の結果が生じたり、重大なエラーを引き起こす原因になることがあります。 呼び出すすべての外部関数に属性テーブルを使用する必要があります。

関連項目:

関数:

- “PEEKLONG 関数: UNIX” (286 ページ)

その他の参照資料:

- “SASCBTBL 属性テーブル” (106 ページ)

CALL SLEEP ルーチン: UNIX

指定の時間帯に、この CALL ルーチンを呼び出すプログラムの実行を一時停止します。

カテゴリ: 特殊

UNIX 固有: すべて

参照項目: “CALL SLEEP Routine” (*SAS Functions and CALL Routines: Reference*)

構文

CALL SLEEP(*n* <, *unit*>);

必須引数

n
数値の定数であり、プログラムの実行を一時停止させる単位時間の数を指定します。

オプション引数

unit
秒単位で単位時間を指定し、*n* に適用します。たとえば、1 は 1 秒に、001 は 1 ミリ秒に 5 は 5 秒に対応します。

デフォルト .001

詳細

CALL SLEEP は、CPU 時間を使わず、また入力/出力を実行せずに、呼び出した DATA ステップを休止待機状態に設定します。複数の SAS プロセスを実行中の場合は、各プロセスは他のプロセスに作用することなしに、独立して CALL SLEEP を実行できます。

注: スリープ時間を長く設定すると、サイトで設定されたタイムアウト値を基に、自動的にホストセッションは終了することがあります。ホストシステムの管理者に連絡して、サイトで使用するタイムアウトの値を決定します。

CALL SYSTEM ルーチン: UNIX

動作環境のコマンドをサブミットして実行します。

カテゴリ: 特殊

UNIX 固有: *Command* は有効な UNIX コマンドとして評価される必要があります。

参照項目: “CALL SYSTEM Routine” (*SAS Functions and CALL Routines: Reference*)

構文

CALL SYSTEM(*command*);

必須引数

command
次のものを指定します。

- 引用符で囲まれた UNIX コマンド
- 値が UNIX コマンドである式
- 値が UNIX コマンドである文字変数の名前

詳細

CALL SYSTEM ルーチンは、オペレーティングシステムのコマンドを実行します。コマンドの出力結果は、SAS を起動したウィンドウに表示されます。

XSYNC システムオプションの値は、CALL SYSTEM ルーチンの動作に影響を及ぼします。

注: CALL SYSTEM ルーチンは、DATA ステップ内で実行できます。ただし、X ステートメントと、%SYSEXEC マクロプログラムステートメントのどちらも、DATA ステップの実行中の使用には意図されていません。

次の例では、**resp** 変数が **y** の場合に、CALL SYSTEM ルーチンによって、メッセージが **answer.week** 内の各レコードにメールで送られます。

```
data _null_;
set answer.week;
if resp='y' then
do;
call system('mail mgr < $HOME/msg');
end;
run;
```

関連項目:

"SAS セッションからオペレーティングシステムコマンドの実行" (15 ページ)

COLLATE 関数: UNIX

文字列を ASCII 照合順序で返します。

カテゴリ: 文字

UNIX 固有: ASCII 照合順序を使用

参照項目: "COLLATE Function" (*SAS Functions and CALL Routines: Reference*)

構文

COLLATE(*start-position* <, *end-position*>) | (*start-position* <, *length*>)

必須引数

start-position
返された最初の文字の照合順序で数値の位置を指定します。

オプション引数

end-position
返された最後の文字の照合順序で数値の位置を指定します。

length
照合順序で文字の数を指定します。

詳細

COLLATE 関数により、ASCII 文字の文字列が返されます。ASCII 照合順序には、0 から 255 の数で参照される、256 の場所が含まれます。127 以上の文字は、ISO 8859 文字セットで定義されたように、欧州言語で使用される文字に対応しています。

COLLATE 関数の戻り値を、200 未満に定義された長さを持つ変数に割り当てない限り、ASCII 照合順序文字列は、200 の長さまで空白で埋められます。ASCII 照合順序文字列が 201 文字以上の場合は、LENGTH ステートメント内で、戻り文字列の長さを指定してください。さもないと、返された文字列は 200 文字の長さに切り捨てられます。詳細については、次の例を参照してください。

例

例 1: 変数の長さを 200 文字に切り捨てる

次のコードには、LENGTH ステートメントが含まれないため、ADDRESS 変数の長さ属性は 200 文字に切り捨てられます。

```
data sales;
Address=collate(1, 241);
run;
proc contents;
run;
```

アウトプット 14.1 *PROC CONTENTS 出力の一部*

```
Alphabetic List of Variables and Attributes
# Variable Type Len
1 Address Char 200
```

ADDRESS の長さは 200 文字までに制限されているため、COLLATE 関数から返された文字列は、200 文字までに制限されます。

例 2: 201 文字以上の長さを指定する

LENGTH ステートメントを使って、特定の変数に 201 文字以上の長さを指定します。次のコードでは、ADRESS の長さは 240 文字に指定されています。

```
data sales;
length Address $240;
Address=collate(1, 241);
run;
proc contents;
run;
```

アウトプット 14.2 *PROC CONTENTS 出力の一部*

```
Alphabetic List of Variables and Attributes
# Variable Type Len
1 Address Char 240
```

ADDRESS の長さは 240 文字に設定されているため、COLLATE 関数から返された文字列は、240 文字を含みます。

関連項目:

ステートメント:

- “LENGTH ステートメント: UNIX” (343 ページ)

DINFO 関数: UNIX

ディレクトリについての情報を返します。

カテゴリ: 外部ファイル

UNIX 固有: ディレクトリパス名、所有者、グループ、権限、最終変更時間の情報アイテムが使用可能

参照項目: “DINFO Function” (*SAS Functions and CALL Routines: Reference*)

構文

DINFO(*directory-id*, *information-item*)

必須引数

directory-id
ディレクトリが開いたときに割り当てられた識別子を指定します。通常、DOPEN 関数によって開かれます。

information-item
検索される情報アイテムを指定します。*information-item* の値が無効の場合、DINFO は空白を返します。

詳細

DOPEN 関数で開かれるディレクトリは、*directory-id* によって識別されます。使用可能なオペレーティングシステム依存情報アイテムの名前を決定するには、DOPTNAME を使用します。使用可能なディレクトリ情報アイテムの数を決定するには、DOPTNUM を使用します。

UNIX で使用可能な情報アイテムは、ディレクトリパス名(*directory-id* のパス名)、所有者、グループ、権限、最終変更時間です。*directory-id* が連結ディレクトリのリストを指す場合、ディレクトリは連結ディレクトリ名のリストです。

関連項目:

関数:

- “DOPEN 関数: UNIX” (274 ページ)
- “DOPTNAME 関数: UNIX” (275 ページ)
- “DOPTNUM 関数: UNIX” (276 ページ)

DOPEN 関数: UNIX

ディレクトリを開き、ディレクトリ識別子の値を返します。

カテゴリ: 外部ファイル

UNIX 固有: *fileref* を環境変数に割り当てられます。

参照項目: “DOPEN Function” (*SAS Functions and CALL Routines: Reference*)

構文

DOPEN(*fileref*)

必須引数

fileref
ディレクトリへと割り当てられるファイル参照名を指定します。DATA ステップでは、*fileref* は文字式、引用符で囲んだ文字列またはその値にファイル参照名を含む DATA ステップ変数になれます。マクロ内では、*fileref* を任意の式にすることができます。

詳細

DOPEN はディレクトリを開き、ディレクトリ識別子の値(正の数)を返します。この値は、他の SAS 外部ファイルアクセス関数内で開いているディレクトリを識別するのに使われます。そのディレクトリを開けなかった場合、DOPEN は 0 を返します。開かれるディレクトリはファイル参照名によって識別される必要があります。

DOPTNAME 関数: UNIX

ディレクトリ属性情報を返します。

カテゴリ: 外部ファイル

UNIX 固有: ディレクトリパス名、所有者、グループ、権限、最終変更時間の情報アイテムが使用可能

参照項目: “DOPTNAME Function” (*SAS Functions and CALL Routines: Reference*)

構文

DOPTNAME(*directory-id*, *nval*)

必須引数

directory-id
ディレクトリが開いたときに割り当てられた識別子を指定します。通常、DOPEN 関数によって開かれます。

nval
情報アイテムのシーケンス数を指定します。

詳細

UNIX で使用可能な情報アイテムは、ディレクトリパス名(*directory-id* のパス名)、所有者、グループ、権限、最終変更時間です。ディレクトリの *nval* またはシーケンス数は 1 です。*directory-id* が連結ディレクトリのリストを指す場合、ディレクトリは連結ディレクトリ名のリストになります。

DOPTNUM 関数: UNIX

ディレクトリに利用可能な情報アイテムの数を返します。

カテゴリ: 外部ファイル

UNIX 固有: ディレクトリパス名、所有者、グループ、権限、最終変更時間の情報アイテムが使用可能

参照項目: “DOPTNUM Function” (*SAS Functions and CALL Routines: Reference*)

構文

DOPTNUM(*directory-id*)

必須引数

directory-id
ディレクトリが開いたときに割り当てられた識別子を指定します。通常、DOPEN 関数によって開かれます。

詳細

UNIX で使用可能な情報アイテムは、ディレクトリパス名(*directory-id* のパス名)、所有者、グループ、権限、最終変更時間です。情報アイテムの名前は directory です。その値は *directory-id* のパス名またはパス名のリストになり、そのシーケンス数は 1 です。

FDELETE 関数: UNIX

外部ファイルまたは空のディレクトリを削除します。

カテゴリ: 外部ファイル

UNIX 固有: *fileref* を環境変数に割り当てられます。

参照項目: “FDELETE Function” (*SAS Functions and CALL Routines: Reference*)

構文

FDELETE("*fileref*")

必須引数

fileref
外部ファイルまたはディレクトリへと割り当てられるファイル参照名を指定します。ファイル参照名を、連結ファイル名または連結ディレクトリのリストに関連付けることはできません。ファイル参照名がディレクトリに関連付けられる場合、そのディレクトリは空である必要があります。ファイルを削除するための許可が必要です。許可についての詳細は、UNIX の `chmod` の man ページを参照してください。

UNIX では、*fileref* は環境変数にもなれます。その *fileref* は二重引用符で囲む必要があります。

詳細

FDELETE は、その操作が成功した場合は 0 を返し、失敗したときは 0 以外の数を返します。

FEXIST 関数: UNIX

ファイル参照名に関連付けられている外部ファイルの存在を確認します。

カテゴリ: 外部ファイル

UNIX 固有: *fileref* を環境変数に割り当てられます。

参照項目: “FEXIST Function” (*SAS Functions and CALL Routines: Reference*)

構文

FEXIST(*fileref*)

必須引数

fileref
外部ファイルまたはディレクトリへと割り当てられるファイル参照名を指定します。DATA ステップでは、*fileref* は文字式、引用符で囲んだ文字列またはその値にファイル参照名を含む DATA ステップ変数になれます。マクロ内では、*fileref* を任意の式にすることができます。

UNIX では、*fileref* は環境変数にもなれます。その *fileref* または指定する環境変数は、二重引用符で囲む必要があります。

詳細

FEXIST 関数は、*fileref* に関連する外部ファイルが存在するとき 1 の値を返し、存在しないときは 0 を返します。

FILEEXIST 関数: UNIX

外部ファイルの存在を、その物理名によって確認します。

カテゴリ: 外部ファイル

UNIX 固有: *filename* を環境変数に割り当てられます。

参照項目: “FILEEXIST Function” (*SAS Functions and CALL Routines: Reference*)

構文

FILEEXIST(*filename*)

必須引数

filename
外部ファイルに、適した物理名を指定します。DATA ステップ内では、*filename* は文字式、引用符で囲まれた文字列または DATA ステップ変数になれます。マクロ内では、*filename* を任意の式にすることができます。

UNIX では、*filename* は環境変数にもなれます。その *filename* または指定する環境変数は、二重引用符で囲む必要があります。

詳細

FILEEXIST は、外部ファイルが存在するときに 1 を返し、存在しないときは 0 を返します。

ディレクトリの存在をチェックするには、FILEEXIST を使います。

FILENAME 関数: UNIX

外部ファイル、ディレクトリまたは出力デバイスのファイル参照名を割り当て/割り当てを解除します。

カテゴリ: 外部ファイル

UNIX 固有: *fileref* は、環境変数や次の項目の有効な値に割り当てることができます。*device-type*、*host-options*

参照項目: “FILENAME Function” (*SAS Functions and CALL Routines: Reference*)

構文

FILENAME(*fileref*, *filename* <, *device-type* <, *"host-options"* <, *directory-reference*>>>)

必須引数

fileref
外部ファイルに割り当てる *fileref* を指定します。DATA ステップでは、*fileref* は文字式、引用符で囲んだ文字列またはその値にファイル参照名を含む DATA ステップ変数になれます。マクロ(%SYSFUNC 関数など)内では、*fileref* は(アンパサンドなしの)マクロ変数の名前になり、その値には外部ファイルに割り当てるファイル参照名が含まれます。(詳細については、“FILENAME Function” (*SAS Functions and CALL Routines: Reference*)を参照してください。)

UNIX では、その *fileref* は UNIX 環境変数になれます。その *fileref* または指定する環境変数は、二重引用符で囲む必要があります。

filename
外部ファイルを指定します。空白のファイル名(" ")を指定すると、*fileref* の割り当てが解除されます。

UNIX では、ファイル名はデバイスタイプにより異なります。各デバイスにの詳細については、“FILENAME ステートメントのデバイス情報” (336 ページ)を参照してください。UNIX ファイル名は大小文字が区別されるので注意してください。

DATA ステップでは、*filename* は文字式、引用符で囲んだ文字列またはその値にファイル名を含む DATA ステップ変数になれます。マクロ内では、*filename* を任意の式にすることができます。

オプション引数

device-type
物理ファイルではない入力/出力デバイスまたは位置をファイル参照名が示す場合に使われるデバイスまたはアクセス方法の種類を指定します。そのデバイスタイプは、“FILENAME ステートメントのデバイス情報” (336 ページ)内で示されているデバイスのうちのいずれにもなれます。DISK がデフォルトのデバイスタイプです。

host-options
UNIX に特有のオプションです。FILENAME ステートメント内で利用可能なオプションのどれでも使えます。ホストオプションの詳細については、“FILENAME ステートメント: UNIX” (331 ページ) を参照してください。

要件 ホストオプションを引用符で囲んでください。複数のホストオプションがある場合、すべてのホストオプションを 1 組の引用符で囲んでください。次の例は、その構文を示します。

```
rc=filename("try", "MISCHL.FLAT.FILE1", "ftp",
'user="mischl1", host="sdcunx", prompt');
```

directory-reference
外部ファイルが存在するディレクトリへと割り当てられるファイル参照名を指定します。

詳細

FILENAME は、その操作が成功した場合は 0 を返し、失敗したときは 0 以外の数を返します。

FTP アクセス方法を使って、リモートシステムと通信する場合は、次のエラーメッセージが返されることがあります。

```
ERROR: Physical file does not exist.
```

このエラーは、適切なデータセット名が単一引用符内で指定されるときに起きる可能性が高いです。たとえば、次のようになります。

```
FILENAME fileref FTP 'system.dataset.name' USER='username'
PASS='password' HOST='ip_address';
```

デフォルトでは、プロファイル接頭辞がデータセット名の頭に付けられます。データセット名を二重引用符と単一引用符の両方で囲めば、プロファイル接頭辞は追加されません。

```
FILENAME fileref FTP "'external_file'" USER='username'
PASS='password' HOST='ip_address';
```

FILEREF 関数: UNIX

ファイル参照名が現在の SAS セッションに割り当てられているか確認します。

カテゴリ: 外部ファイル

UNIX 固有: *fileref* を環境変数に割り当てられます。

参照項目: “FILEREF Function” (*SAS Functions and CALL Routines: Reference*)

構文

FILEREF(*fileref*)

必須引数

fileref
割り当てられるファイル参照名が有効になるように指定します。DATA ステップでは、*fileref* は文字式、引用符で囲んだ文字列またはその値にファイル参照名を含

む DATA ステップ変数になれます。マクロ内では、*fileref* を任意の式にすることができます。

UNIX では、*fileref* は UNIX 環境変数にもなれます。その *fileref* または指定する環境変数は、二重引用符で囲む必要があります。

詳細

負の数の戻り値コードは、ファイル参照名は存在しても、ファイル参照名に関連する物理ファイルは存在しないことを表します。正の値であれば、ファイル参照名は割り当てられていません。0 の値であれば、ファイル参照名と外部ファイルの両方が存在します。

詳細については、“FILENAME 関数: UNIX” (278 ページ)を参照してください。

FINFO 関数: UNIX

外部ファイルにファイル情報アイテムの値を返します。

カテゴリ: 外部ファイル

UNIX 固有: *information-item* が使用できます。

参照項目: “FINFO Function” (*SAS Functions and CALL Routines: Reference*)

構文

FINFO(*file-id*, *information-item*)

必須引数

file-id
ファイルが開いたときに割り当てられた識別子を指定します。通常、FOPEN 関数によって開かれます。

information-item
検索されるファイル情報アイテムの名前を指定します。この値は、文字の値です。*information-item* は、有効な値を含む変数か、引用符で囲まれた有効な値のどちらかです。

UNIX では、ディスクファイルの *information-item* には、次の値の 1 つが含まれます。

- ファイル名
- 所有者名
- グループ名
- アクセス権限
- ファイルサイズ(バイト単位)

ファイル名を連結する場合、追加の *information-item* (ファイルリスト)を使用できます。

パイプファイルを使用している場合、*information-item* の唯一有効な値は PIPE コマンドです。

詳細

FINFO 関数は、FOPEN 関数によってあらかじめ開かれ *file-id* を割り当てられた外部ファイルに、システム依存情報アイテムの値を返します。*information-item* の値が無効の場合、FINFO は空白を返します。

FINFO 関数の使用例については、“例: パイプデバイスタイプ使用時のファイル属性” (282 ページ)を参照してください。

関連項目:

関数:

- “FOPEN Function” (*SAS Functions and CALL Routines: Reference*)

FOPTNAME 関数: UNIX

外部ファイルについての情報アイテムの名前を返します。

カテゴリ: 外部ファイル

UNIX 固有: 利用可能な情報アイテム

参照項目: “FOPTNAME Function” (*SAS Functions and CALL Routines: Reference*)

構文

FOPTNAME(*file-id*, *nval*)

必須引数

file-id
ファイルが開いたときに割り当てられた識別子を指定します。通常、FOPEN 関数によって開かれます。

nval
検索されるファイル情報アイテムの数を指定します。次の表では、UNIX 環境で *nval* に存在する、単一ファイル、パイプファイル、連結ファイルの値を示します。

ファイル情報アイテム			
nval	**単一ファイル**	**パイプファイル**	**連結ファイル**
1	ファイル名	PIPE コマンド	ファイル名
2	所有者名		ファイルリスト
3	グループ名		所有者名
4	アクセス権限		グループ名
5	ファイルサイズ(バイト単位)		アクセス権限

ファイル情報アイテム			
nval	**単一ファイル**	**パイプファイル**	**連結ファイル**
6			ファイルサイズ(バイト単位)

詳細

FOPTNAME は、エラー発生時に空白を返します。

例: パイプデバイスタイプ使用時のファイル属性

次の例では、パイプ使用時に FOPTNAME 関数により返された NAME 属性と VALUE 属性を含むデータセットを作成します。

```
data fileatt;
length name $ 20 value $ 40;
drop fid j infonum;
filename mypipe pipe 'UNIX-command';
fid=fopen("mypipe", "s");
infonum=foptnum(fid);
do j=1 to infonum;
name=foptname(fid, j);
value=finfo(fid, name);
put 'File attribute' name 'has a value of ' value;
output;
end;
run;
```

次のステートメントは SAS ログ内に表示されます。

ログ 14.1 *SAS ログ*

```
File attribute Pipe Command has a value of UNIX-command
```

Unix-command は、UNIX のコマンドまたはプログラムです。ここで出力をパイプしたり、入力を読み込みます。このコマンドまたはプログラムは PATH 環境変数内で、完全修飾あるいは定義される必要があります。

関連項目:

関数:

- “FINFO 関数: UNIX” (280 ページ)
- “FOPEN Function” (*SAS Functions and CALL Routines: Reference*)
- “FOPTNUM 関数: UNIX” (282 ページ)

FOPTNUM 関数: UNIX

外部ファイルに利用可能な情報アイテムの数を返します。

カテゴリ: 外部ファイル

UNIX 固有: 利用可能な情報アイテム

参照項目: “FOPTNUM Function” (*SAS Functions and CALL Routines: Reference*)

構文

FOPTNUM(*file-id*)

必須引数

file-id
ファイルが開いたときに割り当てられた識別子を指定します。通常、FOPEN 関数によって開かれます。

詳細

UNIX では、すべての種類のファイルに次の 5 つのアイテムが利用できます。

- ファイル名
- 所有者名
- グループ名
- アクセス権限
- ファイルサイズ(バイト単位)

ファイル名を連結する場合、追加の情報アイテム(ファイルリスト)を使用できます。パイプファイルを使用している場合、唯一使用可能な情報アイテムは PIPE コマンドです。

FOPEN 関数で指定された *open-mode* により、FOPTNUM が返す値が決まります。

表 14.1 *オープンモードと FOPTNUM 値*

オープンモード	FOPTNUM 値	利用可能な情報アイテム
追加 入力 更新	連結ファイルに対して 6 単一ファイルに対して 5	利用可能なすべての情報アイテム。
出力	連結ファイルに対して 5 単一ファイルに対して 4	ファイルは、入力に開いているため、ファイルサイズ情報の種類は利用できません。
シーケンシャル (パイプデバイスタイプを使用します。)	1	利用できる唯一の情報アイテムは PIPE コマンドです。

FOPTNUM 関数の使用例については、“例: パイプデバイスタイプ使用時のファイル属性” (282 ページ)を参照してください。

関連項目:

関数:

- “FINFO 関数: UNIX” (280 ページ)
- “FOPEN Function” (*SAS Functions and CALL Routines: Reference*)
- “FOPTNAME 関数: UNIX” (281 ページ)

MODEXIST 関数: UNIX

インストールした SAS のリリースにプロダクトイメージが存在するかどうかを決定します。

カテゴリ: 数値

UNIX 固有: *pathname* が利用可能

参照項目: “MODEXIST Function” (*SAS Functions and CALL Routines: Reference*)

構文

MODEXIST('*product-name*' | '*pathname*')

必須引数

'product-name'
検証中のプロダクトイメージの名前である、文字の定数、変数または式を指定します。

pathname')
検証中のプロダクトイメージのパス名を指定します。

詳細

MODEXIST 関数では、実行可能モジュールの *pathname* 引数にリストアップされるディレクトリが検索されます。実行可能モジュールの名前は MODEXIST へと渡されます。モジュールが見つかれば、MODEXIST は 1 を返し、見つからなければ、0 を返します。

MOPEN 関数: UNIX

ディレクトリ ID とメンバ名によりファイルを開き、ファイル識別子か 0 のどちらかを返します。

カテゴリ: 外部ファイル

UNIX 固有: OPEN モード

参照項目: “MOPEN Function” (*SAS Functions and CALL Routines: Reference*)

構文

MOPEN(*directory-id*, *member-name* <, *open-mode* <, *record-length* <, *record-format*>>>)

必須引数

open-mode
次のように、ファイルへのアクセスタイプを指定します。

A APPEND モードでは、ファイルの現在の終端の後ろに新しいレコードを書き込めます。

I INPUT モードでは、デフォルトで、読み込みのみできます。

O OUTPUT モードでは、FILENAME ステートメントまたは関数内のホストオプションで指定された OPEN モードをデフォルトにします。ホストオプションが指定されない場合、ファイルの先頭に新しいレコードを書き込めます。

S シーケンシャル入力モードは、パイプとハードウェアポートのような他のシーケンシャルデバイスに使用されます。

U UPDATE モードでは、読み込みと書き込みの両方ができます。

W シーケンシャル更新モードは、パイプとポートのような他のシーケンシャルデバイスに使用されます。

詳細

注: このバージョンは、MOPEN 関数構文の簡素化されたバージョンです。構文およびその説明については、“MOPEN Function” (*SAS Functions and CALL Routines: Reference*) を参照してください。

MOPEN はファイルに識別子を返します。ファイルを開けなかったときは、0 を返します。

PATHNAME 関数: UNIX

SAS ライブラリまたは外部ファイルの物理名を返します。または、空白を返します。

カテゴリ: SAS ファイル I/O

UNIX 固有: *fileref* または *libref* 引数では、UNIX 環境変数も指定できます。

参照項目: “PATHNAME Function” (*SAS Functions and CALL Routines: Reference*)

構文

PATHNAME((*fileref* | *libref*) <, *search-reference*>)

必須引数

fileref
外部ファイルへと割り当てられるファイル参照名を指定します。DATA ステップでは、*fileref* は文字式、引用符で囲んだ文字列またはその値にファイル参照名を含む DATA ステップ変数になれます。マクロ内では、*fileref* を任意の式にすることができます。

fileref の値は UNIX 環境変数になれます。

libref
SAS ライブラリに割り当てられるファイル参照名を指定します。DATA ステップでは、*libref* は文字式、引用符で囲んだ文字列またはその値にライブラリ参照名を含

む DATA ステップ変数になれます。マクロ内では、*libref* を任意の式にすることができます。

libref の値は UNIX 環境変数になれます。

オプション引数

search-reference
ファイル参照名またはライブラリ参照名を検索するかどうかを指定します。

F ファイル参照名の検索を指定します。

L ライブラリ参照名の検索を指定します。

詳細

PATHNAME は、外部ファイルまたは SAS ライブラリの物理名を返します。*fileref* または *libref* が無効の場合は、空白を返します。

fileref または *libref* への UNIX 環境変数の使用についての詳細は、“FILENAME 関数: UNIX” (278 ページ)を参照してください。

PEEKLONG 関数: UNIX

32 ビットと 64 ビットのプラットフォーム上で、メモリアドレスの中身を数値変数内に保存します。

カテゴリ: 特殊

操作: SAS Server がロックダウン状態にある場合、PEEKLONG 関数は実行されません。詳細については、“SAS Processing Restrictions for Servers in a Locked-Down State” (*SAS Language Reference: Concepts* 2 章)を参照してください。

UNIX 固有: すべて

参照項目: “PEEKLONG Function” (*SAS Functions and CALL Routines: Reference*)

構文

PEEKCLONG(*address*, *length*)

PEEKLONG(*address*, *length*)

必須引数

address
メモリアドレスである文字列を指定します。

length
データの長さを指定します。

詳細

注意:
MODULE 関数の 1 つに返された情報にアクセスするときのみ、PEEKLONG 関数を使用してください。

PEEKLONG 関数は、メモリ *adress* から始まるデータを含む *length* の値を返します。.

PEEKLONG 関数のバリエーションを示します。

PEEKCLONG
文字列にアクセスします。

PEEKLONG
数値にアクセスします。

通常、PEEKLONG 関数の 1 つを使う必要がある場合は、PEEKCLONG を使って文字列にアクセスすることになります。PEEKLONG 関数は、完了させるために言及されます。

RANK 関数: UNIX

ASCII 照合順序で文字の位置を返します。

カテゴリ: 文字

UNIX 固有: ASCII 照合順序を使用

参照項目: “RANK Function” (*SAS Functions and CALL Routines: Reference*)

構文

RANK(*x*)

必須引数

x
ASCII 照合順序での文字を含む、文字の定数、変数、式を指定します。*x* の長さが 2 以上であれば、文字列内の最初の文字のランクを受け取ります。

詳細

UNIX では、ASCII 文字セットが使われるため、RANK 関数は ASCII 照合順序で文字の位置を表す整数を返します。

SYSGET 関数: UNIX

指定された動作環境の変数の値を返します。

カテゴリ: 特殊

UNIX 固有: *environment-variable* は UNIX 環境変数です

参照項目: “SYSGET Function” (*SAS Functions and CALL Routines: Reference*)

構文

SYSGET('*environment-variable*')

必須引数

environment-variable
UNIX 環境変数の名前です。

詳細

SYSGET 関数は、文字列として環境変数の値を返します。たとえば、このステートメントは、HOME 環境変数の値を返します。

```
here=sysget('HOME');
```

TRANSLATE 関数: UNIX

文字式内の特定の文字を置き換えます。

カテゴリ: 文字

UNIX 固有: *to* と *from* 引数が求められます

参照項目: “TRANSLATE Function” (*SAS Functions and CALL Routines: Reference*)

構文

TRANSLATE(*source*, *to-1*, *from-1* <, …*to-n*, *from-n*>)

必須引数

source
本来の文字の値を含む定数、変数、式を指定します。

to
TRANSLATE が置き換えとして使用する文字を指定します。

from
TRANSLATE が置き換える文字を指定します。

詳細

注: このバージョンは、TRANSLATE 関数構文の簡素化されたバージョンです。構文およびその説明については、“TRANSLATE Function” (*SAS Functions and CALL Routines: Reference*)を参照してください。

UNIX では、*to* 引数と *from* 引数のペアを指定してください。そうすれば、カンマを代替として使用できます。

15 章

UNIX 版に固有の入力形式

UNIX 版に固有の SAS 入力形式

このセクションでは、UNIX 環境に固有の動作または構文を含む SAS 入力形式について説明します。各入力形式の説明内容には、どの入力形式が UNIX 版に固有なのかを説明する簡単な“UNIX specifics”セクションが含まれています。これら入力形式についてはすべて、本書と *SAS 出力形式と入力形式: リファレンス*で説明します。

ディクショナリ

HEX*w.* 入力形式:UNIX

16 進数の正の 2 進値を、固定小数点または浮動小数点の 2 進値のいずれかに変換します。

カテゴリ: 数値

デフォルト: 8

範囲: 1 - 16

UNIX 固有: 浮動小数点表現

参照項目: “HEXw. Informat” (*SAS Formats and Informats: Reference*)

詳細

HEX*w.* 入力形式は、正の 2 進数の 16 進数表現を 2 進浮動小数点実数値に変換します。HEX*w.*入力形式のバイト幅の値は、入力が整数(固定小数点)の 2 進数または実数(浮動小数点)の 2 進数を表しているかを判断する入力形式です。バイト幅の値を 1 - 15 に設定すると、入力形式は入力する 16 進数を整数の 2 進数と解釈します。バイト幅の値を 16 に設定すると、入力形式は入力する 16 進数を浮動小数点値と解釈します。

詳細については、"UNIX 環境でのバイナリデータの読み込みと書き込み" (224 ページ)を参照してください。

$HEX*w.* 入力形式:UNIX

16 進数データを文字データに変換します。

カテゴリ: 文字

デフォルト: 2

範囲: 1 - 32,767

UNIX 固有: 数値は ASCII 値として解釈されます。

参照項目: "$HEXw. Informat" (*SAS Formats and Informats: Reference*)

詳細

$HEX*w.* 入力形式は、2 桁の 16 進数データをそれぞれ 1 バイトの文字データに変換する。$HEX*w.*入力形式を使用して、入力データが印刷可能文字に限定される場合に、16 進数値を文字変数にエンコードします。UNIX では、SAS がこの入力形式で読み取る数値を ASCII 値に変換します。

IB*w.d* 入力形式: UNIX

整数の 2 進(固定小数点)値を読み取ります。

カテゴリ: 数値

デフォルト: 4

範囲: 1 - 8, 0 - 10

UNIX 固有: バイト値

参照項目: "IBw.d Informat" (*SAS Formats and Informats: Reference*)

詳細

IB*w.d* 入力形式は、固定小数点バイナリ値を読み取ります。整数のバイナリデータについて、上位ビットは、正の数値に対する数値記号は 0、負の数値に対する数値記号は 1 です。負の数値は 2 の補数で表現されます。この入力形式が *d* 値を含む場合、データ値を 10^d で割ります。

詳細については、"UNIX 環境でのバイナリデータの読み込みと書き込み" (224 ページ)を参照してください。

PD*w.d* 入力形式: UNIX

パック 10 進数で保存されているデータを読み取ります。

カテゴリ: 数値

デフォルト: 1

範囲: 1 - 16, 0 - 31

UNIX 固有: データ表現

参照項目: “PDw.d Informat” (*SAS Formats and Informats: Reference*)

詳細

PD*w.d* 入力形式は、パック 10 進データを読み取ります。パック 10 進データは、通常、コンソールから直接入力はできませんが、パック 10 進データを書き込むプログラムは多数あります。

パック 10 進データでは、1 バイトに 2 桁が含まれます。第 1 バイトは値の記号です。最初のバイト全体が符号として使用されるため、バイト幅は 2 以上を指定してください。

PD*w.d* 出力形式は、欠損数値データを–0 として書き込みます。PD*w.d* 入力形式が–0 の値を読み取ると、結果は 0 の値になります。

詳細については、“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

PIB*w.d* 入力形式: UNIX

正のバイナリ整数(固定小数点)値を読み込みます。

カテゴリ: 数値

デフォルト: 1

範囲: 1 - 8, 0 - 10

UNIX 固有: バイトオーダー

参照項目: “PIBw.d Informat” (*SAS Formats and Informats: Reference*)

詳細

PIB*w.d* 入力形式では、正の整数のバイナリ(固定小数点)値を読み取ります。正の整数バイナリ値は、整数バイナリと同じです(“IBw.d 入力形式: UNIX” (290 ページ) を参照)。ただし、全ての数値を正の数として扱う場合を除きます。したがって上位ビットは、その数値の記号ではなくその数値の一部です。この入力形式が *d* 値を含む場合、データ値を 10^d で割ります。

詳細については、“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

RB*w.d* 入力形式:UNIX

実数バイナリ(浮動小数点)表現で保存されている数値データを読み取ります。

カテゴリ: 数値

デフォルト: 4

範囲: 2 - 8, 0 - 10

UNIX 固有: 浮動小数点表現、数値データを切り捨てるアプリケーション専用の単精度数をサポート

参照項目: “RBw.d Informat” (*SAS Formats and Informats: Reference*)

詳細

RB*w.d* 入力形式では、実数のバイナリ(浮動小数点)表現で保存されている数値データを読み取ります。SAS はすべての数値を浮動小数点で保存します。

浮動小数点バイナリデータは、通常、コンソールから直接入力はできませんが、浮動小数点バイナリデータを書き込むプログラムは多数あります。RB*w.d* 入力形式は倍精度データの読み取りのみを目的としているため、SAS 以外のプログラムで作成した浮動小数点データの読み取りに RB*w.d* 入力形式を使っている場合は注意が必要です。

現在 SAS でサポートされている UNIX システムはすべて、浮動小数点表現に IEEE 基準を適用しています。この表現法は、浮動小数点について単精度と倍精度の両方をサポートしています。倍精度の表現ではバイトがより正確で、この表現内のデータの解釈はさまざまです。たとえば、16 進数表現の 1 が単精度では **3F800000** になり、倍精度では **3FF0000000000000** になります。

RB*w.d* 入力形式は、倍精度データの読み取りのみを目的としています。スペースを節約するために数値データを切り捨てるアプリケーションを対象とする場合のみ、この入力形式は 8 未満のバイト幅をサポートします。RB4.は単精度の浮動小数点を*予期せず*、4 バイトに切り捨てられた倍精度数を予期します。例として前述の 1 を使用すると、RB4.は **3FF00000** を、1 と解釈されるデータの 4 バイトの 16 進数表現であると予期します。この **3F800000** を単精度値 1 とすると、得られる数値は異なります。

C 言語やフォートラン言語で書かれている外部プログラムなどでは、単精度または倍精度の浮動小数点の数値のみを生成できます。4バイトまたは 8 バイト以外の長さは無効です。RB*w.d* では、保存の必要があるストレージによって、有効な長さは 3 - 8 となります。

FLOAT4.入力形式は、単精度の浮動小数点を読み取るために作成されました。FLOAT4.で 3F800000 を読み込むと、結果は 1 になります。

C プログラムまたはフォートラムプログラムで作成されたデータを読み取るには、使用に適した入力形式を決める必要があります。浮動小数点に必要なバイト幅が 8 の場合は、RB8.入力形式を使用してください。浮動小数点に必要なバイト幅が 4 の場合は、FLOAT4.を使用してください

次の C の例を考えてみましょう。

```
#include <stdio.h>
main() {
FILE *fp;
float x[3];
fp = fopen("test.dat","wb");
x[0] = 1; x[1] = 2; x[2] = 3;
fwrite((char *)x,sizeof(float),3,fp);
```

```
fclose(fp);
}
```

ファイル test.dat は 16 進数表現で `3f8000004000000040400000` を含みます。

次のステートメントは test.dat を正しく読み取ります。

```
 data _null_;
infile 'test.dat';
input (x y z) (float4.);
run;
```

また、IEEE*w.d* 入力形式も使用できます。これは IEEE 浮動小数点データを読み取ります。UNIX システムでは、IEEE8.は RB8.に相当し、IEEE4.は FLOAT4.に相当します。IEEE*w.d* は、IEEE 表現法を利用するプラットフォームで作成される元の IEEE バイナリデータと同じ長さであれば、プラットホームで使用できます。

詳細については、“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

ZD*w.d* 入力形式: UNIX

ゾーン 10 進データを読み取ります。

カテゴリ: 数値

デフォルト: 1

範囲: 1 - 32

UNIX 固有: 最終バイトは符号を含みます。データ表現

参照項目: “ZDw.d Informat” (*SAS Formats and Informats: Reference*)

詳細

ZD*w.d* 入力形式は、ゾーン 10 進データを読み取ります。また、重ね打ち末尾数値形式としても知られています。UNIX では、フィールドの最後のバイトは符号と最終桁を含みます。次に最後のバイトの変換表を示します。

数字	ASCII 文字	数字	ASCII 文字
0	{	–0	}
1	A	–1	J
2	B	–2	K
3	C	–3	L
4	D	-4	M
5	E	–5	N
6	F	-6	O
7	G	–7	P

8	H	–8	Q
9	I	–9	R

詳細については、“ZDw.d 出力形式: UNIX” (264 ページ) および“UNIX 環境でのバイナリデータの読み込みと書き込み” (224 ページ)を参照してください。

16 章

UNIX 版に固有のマクロ機能

UNIX 版に固有のマクロ機能について

SAS のマクロ機能の大部分は、どのような動作環境でも有効です。本書では、UNIX 環境によるマクロ機能のこのような要素のみに焦点を当てて説明します。詳細については、次のドキュメントをご参照ください。

- *SAS マクロ言語: リファレンス*
- *SAS マクロ機能のヒントとテクニック*
- マクロ機能のオンラインヘルプ

UNIX 環境の自動マクロ変数

次の自動マクロ変数はすべての動作環境で有効ですが、動作環境によって決定されます。

SYSCC
現在の SAS 条件コードを含みます。SAS は終了時に、この条件コードを動作環境に対して意味のある値を含むリターンコードに変換します。

注: SYSCC の値は、オペレーティングシステムから返されるリターンコードとは一致しない可能性があります。

UNIX では返されるコードは次のとおりです。

0
正常終了

1
SAS からの警告

2
SAS からのエラー表示

3
ABORT;

4
ABORT RETURN *n*;

5
ABORT ABEND *n*;

6
内部エラー

注: ERRORCHECK=NORMAL の場合、LIBNAME ステートメントや FILENAME ステートメント、または SAS/SHARE ソフトウェアの LOCK ステートメントにエラーがあったとしても、戻りコードは 0 になります。ファイルが存在しないために%INCLUDE ステートメントが失敗した場合も、SAS ジョブや SAS セッションは中止されません。詳細については、“ERRORCHECK= System Option” (*SAS System Options: Reference*)を参照してください。

SYSDEVIC
現在のグラフデバイス名を含みます。現在のグラフデバイスは、DEVICE システムオプションにより決定されます。オンサイトの SAS サポート担当者と連絡を取り、サイトで利用できるグラフデバイスを決定してください。詳細については、“DEVICE システムオプション: UNIX” (373 ページ) および“DEVICE= System Option” (*SAS/GRAPH: Reference*)を参照してください。

SYSENV
SAS が対話形式で実行されているかを報告します。SYSENV の値は、TERMINAL システムオプションが有効な場合は **FORE** で、NOTERMINAL システムオプションが有効な場合は **BACK** になります。

SYSJOBID
SAS を実行している(たとえば、00024 など) PID (プロセス ID 番号)を一覧表示します。

SYSMAXLONG
UNIX での長整数最大値である 9,007,199,254,740,992 を返します。32 ビットのシステム上では最大値 2,147,483,647 です。

SYSRC
SAS セッションから実行される最後の UNIX コマンドにより返される終了ステータスコードの 10 進値を保持します。次の出力は、2 つの SYRC 値のサンプルを示す対話型ラインモード SAS セッションを示しています。

アウトプット 16.1 サンプル SYSRC 値

```
1? x 'data';
/bin/ksh: data: not found
2? %put UNIX exit status code is &sysrc;
UNIX exit status code is 256
3? x 'date';
Tue Mar 15 09:41:27 CST 2011
4? %put UNIX exit status code is now &sysrc;
UNIX exit status code is now 0
```

SYSSCP
HP IPF、**SUN 64** または **AIX 64** など、プロセッサアーキテクチャの略語を返します。

SYSSCPL
HP-UX、**SunOS** または **AIX** など、使用している特定の UNIX 環境の名前を返します。この変数は、UNIX コマンド **uname** により返される同一の値を返します。

UNIX 環境のマクロステートメント

次のステートメントで入力できる引数は動作環境に影響されます。

%SYSEXEC
UNIX コマンドを実行します。“SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)で説明されている X ステートメントに類似しています。%SYSEXEC ステートメントにより、動作環境コマンドを即座に実行することができます。また必要に応じて、自動マクロ変数 SYSRC の値を調べて動作環境コマンドの実行が成功したかどうかを決定します。マクロ内またはオープンコード内の%SYSEXEC ステートメントを使用できます。%SYSEXEC ステートメントのフォームは次のとおりです。*command* は UNIX のコマンドになります。

%SYSEXEC <*command*>;

たとえば、次のコードは UNIX シェルに対するデフォルトプリンタのステータスを書き込みます。

```
%sysexec lpstat;
```

UNIX コマンドなしで%SYSEXEC を入力すると、新しいシェルが開始されます。ただし SAS の X インターフェースにおける入力は除きます。詳細については、“SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)を参照してください。

UNIX 環境のマクロ関数

次の関数には動作環境への依存性があります。

%SCAN
指定する単語を、文字列での位置により検索します。次に示すのは%SCAN 関数のフォームです。

%SCAN(*argument*,*n*,<*delimiters*>);

ASCII システム上ではデフォルトの区切り文字は次のようになります。

```
blank . < ( + & ! $ * ) ; ^ - / , % |
```

%SYSGET
引数で渡した環境変数の値である文字列を返します。UNIX 環境変数も SAS 環境変数も、%SYSGET 関数を使用して変換できます。グローバル変数が存在しない場合は、警告メッセージが書き込まれます。次に示すのは、%SYSGET 関数のフォームです。

%SYSGET(*environment-variable*);

たとえば、次のコードは HOME 環境変数の値を SAS ログに書き込みます。

```
%let var1=%sysget(HOME); %put &var1;
```

UNIX 環境でマクロ機能が使用する SAS システムオプション

次のシステムオプションには動作環境への依存性があります。

MSYMTABMAX
あらゆる記号テーブル(グローバルとローカルを組み合わせたもの)に使用可能なメモリの最大容量を指定します。UNIX では、このオプションのデフォルト値は 4M になります。

MVARSIZE
メモリに保存されているマクロ変数に対する最大バイト数を指定します。UNIX では、このオプションのデフォルト値は 32K になります。

SASAUTOS
AUTOCALL ライブラリを指定します。詳細については、“SASAUTOS システムオプション” (299 ページ)を参照してください。

UNIX 環境で自動呼び出しライブラリを使用する

自動呼び出しライブラリについて

自動呼び出しライブラリには、SAS マクロを定義するファイルが含まれます。次のセクションでは、動作環境に依存している自動呼び出しライブラリの側面について説明します。詳細については、*SAS マクロ言語: リファレンス*を参照してください。

使用可能な自動呼び出しマクロ

自動呼び出しマクロには、SAS が提供するマクロとユーザーが定義するマクロの 2 種類があります。自動呼び出し機能を使用するには、MAUTOSOURCE システムオプションをセットするが必要です。

SASAUTOS システムオプションは、SAS のインストール時に構成ファイル内に定義され、SAS が提供するデフォルトマクロの位置を示します。サイトでライセンスされたプロダクトは、ユーザー所有の自動呼び出しマクロで使用可能なものを決定します。また、独自の自動呼び出しマクロを定義して 1 つ以上のディレクトリに保存できます。自動呼び出しマクロのファイル名が大文字または大文字と小文字の両方を用いて書かれて

いる場合、そのマクロは SAS に認識されません。ファイル名には小文字のみを使用してください。

マクロファイルの命名ガイドライン

SAS 内のマクロ名では大文字と小文字を区別しませんが、これらマクロ名はすべて、小文字のファイル名に対応付けられます。UNIX ディレクトリに自動呼び出しマクロを保存する場合、ファイルの拡張子 **.sas** とファイル名は全体を小文字にしてください。UNIX 環境では、UNIX ディレクトリの 各マクロは、ファイル名と一致するマクロ名がついているマクロ定義を含む必要があります。たとえば、ファイル名が **prtdata.sas** であるファイルは、マクロ名が **prtdata** であるマクロを定義します。

SASAUTOS システムオプション

SAS プログラム内の独自の自動呼び出しマクロを使用するには、SASAUTOS システムオプションで対象マクロのディレクトリを指定します。詳細については、“SASAUTOS システムオプション: UNIX” (413 ページ)を参照してください。

注: UNIX では、ASAUTOS システムオプションは全体または部分的に大文字を含むファイル名を認識しません。

SASAUTOS システムオプションは SAS 起動時に設定するか、あるいは、SAS セッション中に OPTIONS ステートメント内の SASAUTOS システムオプションを使用することも可能です。ただし、OPTIONS ステートメントで指定する自動呼び出しライブラリは、以前の設定をオーバーライドします。

CONFIG システムオプションを使用して構成ファイルを指定する場合は、SAS が提供するライブラリ連結に自動呼び出しライブラリを追加してください。デフォルトの構成ファイル(sasv9.cfg)を使用する場合は、この自動呼び出しライブラリを指定してください。

自動呼び出しライブラリは、指定される順に検索されます。

例: 自動呼びダリライブラリ内のマクロの設定とテスト

この例は、自動呼び出しライブラリ内のマクロの設定方法とテスト方法を示しています。

次の出力は、2 つの UNIX コマンド(**cat**)を実行して 2 つのファイルのコンテンツを表示した結果と、1 つの SAS コマンドを実行して autocall.sas プログラムを起動した結果を示しています。

アウトプット 16.2 *AUTOCALL ライブラリの例*

```
$ cat maclib/testauto.sas
%macro testauto;
x echo 'Autocall library is working.';
%mend testauto;
$ cat source/autocall.sas
filename sysautos ('!SASROOT/sasautos' '$HOME/test/sasautos');
options mautosource sasautos=(sysautos '$HOME/macros/maclib');
%testauto
%TestAuto
%TESTAUTO
$ sas source/autocall.sas
Autocall library is working.
Autocall library is working.
Autocall library is working.
```

17 章

UNIX 版に固有のプロシジャ

UNIX 版に固有の SAS プロシジャ

このセクションでは、UNIX 環境に固有の動作または構文をもつ SAS プロシジャについて説明します。各プロシジャの説明には、そのプロシジャのどの要素が UNIX 版に固有であるのかを簡単に説明する“UNIX 固有”セクションが含まれています。各プロシジャの説明は、このドキュメントと *Base SAS プロシジャガイド* に記載されています。

ディクショナリ

CATALOG プロシジャ: UNIX

SAS カタログ内のエントリを管理します。

UNIX 固有: CONTENTS ステートメント内の FILE=オプション

参照項目: 9 章: “CATALOG プロシジャ” (*Base SAS Procedures Guide*)

構文

PROC CATALOG CATALOG=<*libref.*> *catalog* <ENTRYTYPE=*etype*> <KILL>;
CONTENTS <OUT=*SAS-data-set*> <FILE=*fileref*>;

オプション引数

fileref
UNIX 動作環境に固有のファイル仕様に命名します。

詳細

注: このバージョンは、CATALOG プロシジャ構文の簡易バージョンです。構文およびその説明については、9 章: “CATALOG プロシジャ” (*Base SAS Procedures Guide*)を参照してください。

CATALOG プロシジャの CONTENTS ステートメント内の FILE=オプションは、ファイル参照名を受け入れます。指定された名前がファイル参照名に対応しない場合、その名前のファイルと拡張子 **.lst** が、カレントディレクトリに作成されます。たとえば、**myfile** がファイル参照名でない場合、次のコードにより、カレントディレクトリ内にファイル **myfile.lst** が作成されます。

```
proc catalog catalog=sasuser.profile;
contents file=myfile;
run;
```

SAS は、次の出力をログに書き込みます。

```
NOTE: 6 entries have been written to the output file /users/userid/MYFILE.lst.
```

注: 作成されるファイル名は、常に小文字で保存されます。大文字で指定した場合でも小文字で保存されます。ただし、SAS ログ内では、ファイル名は大文字で表示されます。

CIMPORT プロシジャ: UNIX

CPORT プロシジャで作成された移送ファイルを復元します。

UNIX 固有: 移送ファイルの名前と場所

参照項目: 11 章: “CIMPORT プロシジャ” (*Base SAS Procedures Guide*)

構文

PROC CIMPORT *destination=libref* | <*libref.*> *member-name* <*option(s)*> ;

必須引数

destination
移送ファイル内の各ファイルを単一の SAS データセット、単一の SAS カタログ、または SAS ライブラリの複数メンバとして識別します。

libref* | *<libref.>member-name
SAS データセット、カタログ、または移送ファイルから作成するライブラリの名前を指定します。

このバージョンは、CIMPORT プロシジャ構文の簡易バージョンです。構文およびその説明については、11 章: “CIMPORT プロシジャ” (*Base SAS Procedures Guide*)を参照してください。

詳細

注: SAS 9.1 以降、MIGRATE プロシジャを使用して、以前のリリースから SAS ライブラリを移行できます。詳細については、"UNIX 環境での 32 ビット版 SAS ファイルの 64 ビット版への移行" (45 ページ)、MIGRATE プロシジャおよび Cross-Release Compatibility を、**テクニカルサポート Web サイト**で参照してください。

CIMPORT プロシジャでは、CPORT プロシジャで作成された移送ファイルを*インポート*します(作成: *エクスポート*)。この移送ファイルには、SAS データセット、SAS カタログ、または SAS ライブラリ全体が含まれています。

通常、INFILE= オプションは、移送ファイルのソースを指定するのに使用されます。このオプションが省略されていると、CIMPORT は、カレントディレクトリ内のデフォルトファイル Sascat.dat を移送ファイルとして使用します。

注: CIMPORT では、CPORT プロシジャにより作成された移送ファイルだけが使用できます。移送ファイルが、XPORT エンジンを使用して COPY プロシジャにより作成された場合、その移送ファイルを復元するには、他の PROC COPY を使用する必要があります。詳細については、14 章: "COPY プロシジャ" (*Base SAS Procedures Guide*)を参照してください。

例: データセットの移動

この例では、複数のデータセットを含む SAS ライブラリが、外部ホスト上で CPORT プロシジャを使ってファイル(**transport-file** と呼ばれる)にエクスポートされました。この移送ファイルは、次に、バイナリ転送により受信側ホストに移動されます。

次のコードは、移送ファイルの中に保存されているすべての SAS データセットとカタログを抽出し、**SAS-library** 新規ライブラリの中でその元の状態に復元します。

```
libname newlib 'SAS-library';
filename tranfile 'transport-file';

proc cimport lib=newlib infile=tranfile;
run;
```

関連項目:

プロシジャ:

- "CPORT プロシジャ: UNIX" (309 ページ)

その他の参照資料:

- "UNIX 環境での 32 ビット版 SAS ファイルの 64 ビット版への移行" (45 ページ)
- 1 章: "Moving and Accessing SAS Files between Operating Environments" (*Moving and Accessing SAS Files*)

CONTENTS プロシジャ: UNIX

SAS ライブラリの 1 つ以上のファイルのコンテンツの説明を印刷します。

UNIX 固有: SAS 出力内に表示される情報

参照項目: 13 章: "CONTENTS プロシジャ" (*Base SAS Procedures Guide*)

構文

PROC CONTENTS <*option(s)*>;

比較

CONTENTS プロシジャでは、DATASETS プロシジャ内の CONTENTS ステートメントと同じ情報が作成されます。(“DATASETS プロシジャ: UNIX” (310 ページ) を参照して比較してください。)

例: PROC CONTENTS の実行

次の SAS コードでは、2 つのデータセット、**classes.grades** および **classes.majors** が作成され、**classes.majors** データセットを説明するために PROC CONTENTS が実行されます。

```
libname classes '.';

data classes.grades (label='First Data Set');
input student year state $ grade1 grade2;
label year='Year of Birth';
format grade1 4.1;
datalines;
1000 1980 NC 85 87
1042 1981 MD 92 92
1095 1979 PA 78 72
1187 1980 MA 87 94
;

data classes.majors(label='Second Data Set');
input student $ year state $ grade1 grade2 major $;
label state='Home State';
format grade1 5.2;
datalines;
1000 1980 NC 84 87 Math
1042 1981 MD 92 92 History
1095 1979 PA 79 73 Physics
1187 1980 MA 87 74 Dance
1204 1981 NC 82 96 French
;

proc contents data=classes.majors;
run;
```

画面 17.1 CONTENTS プロシジャからの出力

The SAS System

The CONTENTS Procedure

Data Set Name	CLASSES.MAJORS	**Observations**	5
Member Type	DATA	**Variables**	6
Engine	V9	**Indexes**	0
Created	Thu, Feb 24, 2011 10:26:34 AM	**Observation Length**	48
Last Modified	Thu, Feb 24, 2011 10:26:34 AM	**Deleted Observations**	0
Protection		**Compressed**	NO
Data Set Type		**Sorted**	NO
Label	Second Data Set		
Data Representation	HP_UX_64, RS_6000_AIX_64, SOLARIS_64, HP_IA64		
Encoding	latin1 Western (ISO)		

Engine/Host Dependent Information	
Data Set Page Size	8192
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	169
Obs in First Data Page	5
Number of Data Set Repairs	0
Filename	/r/sanyo.unx.sas.com/vol/vol810/u81/*user-id*/majors.sas7bdat
Release Created	9.0301B0
Host Created	HP-UX
Inode Number	25962216
Access Permission	rw-r--r--
Owner Name	*user-id*
File Size (bytes)	16384

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Label
4	grade1	Num	8	5.2	
5	grade2	Num	8		
6	major	Char	8		
3	state	Char	8		Home State
1	student	Char	8		
2	year	Num	8		

CONVERT プロシジャ: UNIX

BMDP と OSIRIS システムファイルおよび SPSS エクスポートファイルを SAS データセットに変換します。

UNIX 固有: すべて

構文

PROC CONVERT *product-specification* <*option-list*>;

必須引数

product-specification

Product-specification は、次のいずれかです。

BMDP=*fileref* <(CODE=*code* CONTENT=*content-type*)>

UNIX (AIX)環境で作成された BMDP 保存ファイルの第 1 メンバを SAS データセットに変換します。次は、その例です。

```
filename save '/usr/mydir/bmdp.dat';
proc convert bmdp=save;
run;
```

fileref 引数で参照される BMDP ファイル内に複数の保存ファイルがある場合は、*fileref* の後のかっこ内に 2 つのオプションを使用できます。CODE=オプションは、必要とする保存ファイルのコードを指定し、CONTENT=オプションは、その保存ファイルの内容を指定します。たとえば、**code=judges** のファイルのコンテンツが DATA の場合、次のステートメントを使用できます。

```
filename save '/usr/mydir/bmdp.dat';
proc convert bmdp=save(code=judges
content=data);
run;
```

OSIRIS=*fileref|libref*

SAS データセットへの変換対象の OSIRIS ファイルのファイル参照名またはライブラリ参照名を指定します。また、DICT=オプションを含められます。

SPSS=*fileref|libref*

SAS データセットへの変換対象となる SPSS エクスポートファイルのファイル参照名またはライブラリ参照名を指定します。SPSS ファイルは SPSS EXPORT コマンドを使用して作成する必要がありますが、どのオペレーティングシステムでも使用できます。

オプション引数

option-list

Option-list は、次のいずれかです。

DICT=*fileref|libref*

OSIRIS ファイルの辞書ファイルのファイル参照名またはライブラリ参照名を指定します。DICT=は、OSIRIS 製品仕様で使用される場合のみ有効です。

FIRSTOBS=*n*

BMDP、OSIRIS、または SPSS ファイルの開始点でオブザベーションをスキップできるよう、変換を開始するオブザベーションの番号を提供します。

OBS=*n*

変換対象の最後のオブザベーションの番号を指定します。このオプションを使用すると、ファイル末尾のオブザベーションを除外できます。

OUT=*SAS-data-set*

変換したデータを保持する SAS データセットに名前を付けます。OUT=が指定されてない場合、SAS はそれまでと同じように Work データセットを作成し、DATA ステートメント内でデータセット名を省略した場合とまるで同様に、

DATA*n* という名前を自動的に付けます。詳細については、"UNIX 環境の SAS ファイル、ライブラリ、エンジンについて" (35 ページ)を参照してください。

詳細

システムファイルの変換

この CONVERT プロシジャにより、BMDP と OSIRIS システムファイルおよび SPSS エクスポートファイルが SAS データセットに変換されます。互換性確保のため、このプロシジャが提供されます。このプロシジャで、ファイルを変換するための適切なエンジンが呼び出されます。

PROC CONVERT では 1 つの出力データセットが作成されますが、印刷出力はありません。新規データセットには、入力システムファイルと同じ情報が含まれます。例外は、"欠損値の取り扱い方法" (307 ページ)に記載されています。

このプロシジャでは、これらの製品のシステムファイルが変換されます。

- BMDP では、最新リリースまでのファイルが保存されます(AIX、HP-UX および Solaris のみに利用可能)。
- OSIRIS では、OSIRIS IV までのファイルが保存されます。(階層ファイル構造はサポートされていません。)

BMDP、OSIRIS および SPSS の各製品は SAS 以外の組織が保守しているため、現在のバージョンの PROC CONVERT と互換性のないファイルを新たに作成する変更が加えられる場合があります。SAS では、SAS の新バージョンのリリース時のみ、これらの製品への変更内容に対応するよう、PROC CONVERT がアップグレードされます。

欠損値の取り扱い方法

出力データセット内の数値に値がないか、またはシステム欠損値がある場合、PROC CONVERT はそれに欠損値を割り当てます。

変数名の割り当て法

次のセクションでは、CONVERT プロシジャで作成された SAS 変数への名前の割り当て方法について説明します。

注意:

変換される名前が固有であることを確認してください。 変数は、次のセクションに示される通りに変換されます。

BMDP 出力内の変数名

BMDP 保存ファイルの変数名は SAS データセット内で使用されますが、連続していない空白およびすべての特殊文字は、SAS 変数名の中ではアンダースコアに変換されます。x(1)などの BMDP 変数名の中のサブスクリプトは、かっこを省略した SAS 変数名(つまり、X1)の一部になります。英文字の BMDP 変数は、対応する長さの SAS 文字変数となります。BMDP からのカテゴリレコードは、受け入れられません。

OSIRIS 出力内の変数名

単一応答変数の場合、V1 - V9999 の名前は SAS 変数名となります。複数応答変数では、変数に接頭辞 R*n* が追加されます。ただし、*n* は応答です。たとえば、V25R1 は、複数応答 V25 の最初の応答となります。V1000 より後の変数が 100 以上の応答を持っている場合、99 を超える応答は削除されます。OSIRIS が文字、固定小数点バイナリ、または浮動小数点バイナリモードで保存する数値変数は SAS 数値変数になります。英文字変数は SAS 文字変数になります。長さが 200 を超える英文字変数

は、200 までに切り詰められます。OSIRIS 変数説明は SAS 変数ラベルになり、そし OSIRIS プリント形式情報は SAS 出力形式になります。

SPSS 出力内の変数名

SPSS 変数名および変数ラベルは、変数名および変数ラベルとなり、変更は加えられません。SPSS 英文字変数は同一長さの SAS 文字変数となります。SPSS 空白値は、SAS 欠損値に変換されます。SPSS プリント形式は SAS 出力形式になり、小数点以下の桁がない SPSS デフォルト精度は変数の出力形式の一部になります。SPSS DOCUMENT データは、CONTENTS プロシジャがそれを表示できるようコピーされます。SPSS 値ラベルはコピーされません。

インターフェイスライブラリエンジンとの比較

CONVERT プロシジャは、インタフェースライブラリエンジン BMDP、OSIRIS および SPSS に密接に関連しています。(実際、CONVERT プロシジャではこれらのエンジンを使用します。) たとえば、コードの次の 2 つのセクションでは、得られる結果は同じです。

```
filename myfile 'mybmdp.dat';
proc convert bmdp=myfile out=temp;
run;

libname myfile bmdp 'mybmdp.dat';
data temp;
set myfile._first_;
run;
```

ただし、BMDP、OSIRIS および SPSS エンジンは、PROC CONVERT よりも広範な機能を提供します。たとえば、PROC CONVERT は、保存ファイル内の 1 番目の BMDP メンバのみを変換します。BMDP エンジンは、COPY プロシジャとともに、すべてのメンバをコピーします。

例

例 1: BMDP 保存ファイルの変換

次のステートメントでは、BMDP 保存ファイルが変換され、その変換データを含む一時 SAS データセット **temp** が作成されます。

```
filename bmdpfile 'bmdp.savefile';
proc convert bmdp=bmdpfile out=temp;
run;
```

例 2: OSIRIS ファイルの変換

次のステートメントでは、OSIRIS ファイルが変換され、その変換データを含む一時 SAS データセット **temp** が作成されます。

```
filename osirfile 'osirdata';
filename dictfile 'osirdict';
proc convert osiris=osirfile dict=dictfile
out=temp;
run;
```

例 3: SPSS ファイルの変換

次のステートメントでは、SPSS ファイルが変換され、その変換データを含む一時 SAS データセット **temp** が作成されます。

```
filename spssfile 'spssfile.num1';
proc convert spss=spssfile out=temp;
run;
```

関連項目:

"UNIX 環境で BMDP、OSIRIS、SPSS ファイルにアクセスする" (61 ページ)

CPORT プロシジャ: UNIX

SAS データセットおよびカタログを移送ファイルに書き込みます。

UNIX 固有: 移送ファイルの名前と場所

参照項目: 15 章: "CPORT プロシジャ" (*Base SAS Procedures Guide*)

構文

PROC CPORT *source-type=libref* | <*libref.*> *member-name* <*option(s)*> ;

必須引数

source-type
単一 SAS データセット、単一 SAS カタログ、または SAS ライブラリの複数メンバのうちのいずれかでエクスポートするファイルを識別します。

libref* | <*libref.*> *member-name
エクスポートする SAS データセット、カタログ、またはライブラリの名前を指定します。

詳細

注: このバージョンは、CPORT プロシジャ構文の簡易バージョンです。構文およびその説明については、15 章: "CPORT プロシジャ" (*Base SAS Procedures Guide*)を参照してください。

SAS 9.1 以降、MIGRATE プロシジャを使用して、以前のリリースから SAS ライブラリを移行できます。詳細については、"UNIX 環境での 32 ビット版 SAS ファイルの 64 ビット版への移行" (45 ページ)および **テクニカルサポート Web サイト**を参照してください。

CPORT プロシジャでは、後で CIMPORT プロシジャにより復元される(*インポートされる*)移送ファイルが作成されます。この移送ファイルには、SAS データセット、SAS カタログ、または SAS ライブラリ全体を含められます。

通常、FILE=オプションは、移送ファイルのパスを指定するのに使用されます。FILE=オプションの値としては、FILENAME ステートメントまたは環境変数の中に定義されるファイル参照名を使用できます。このオプションが省略されていると、CPORT は、カレントディレクトリ内のデフォルトファイル Sascat.dat を移送ファイルとして使用します。

例: ファイルのエクスポート

この例では、**oldlib** と呼ばれる SAS ライブラリには、複数のデータセットが含まれており、**transport-file** と呼ばれるファイルにエクスポートされているところです。

```
libname oldlib 'SAS-data-library';
filename tranfile 'transport-file';
```

```
proc cport lib=oldlib file=tranfile;
run;
```

そして、通常この移送ファイルは別のホストにバイナリ転送で移動され、CIMPORT プロシジャを使用して SAS ライブラリが復元されます。

関連項目:

プロシジャ:

- “CIMPORT プロシジャ: UNIX” (302 ページ)

その他の参照資料:

- “UNIX 環境での 32 ビット版 SAS ファイルの 64 ビット版への移行” (45 ページ)
- 1 章: “Moving and Accessing SAS Files between Operating Environments” (*Moving and Accessing SAS Files*)

DATASETS プロシジャ: UNIX

SAS ファイルを管理し、SAS データセットのインデックスおよび整合性制約の作成/削除を行います。

UNIX 固有: ディレクトリ情報、CONTENTS ステートメント出力

参照項目: 16 章: “DATASETS プロシジャ” (*Base SAS Procedures Guide*)

構文

PROC DATASETS <*option(s)*>;
CONTENTS <*option(s)*>;

オプション引数

CONTENTS *option*
option の値には、次を使用できます。

DIRECTORY
UNIX 動作環境に固有な情報のリストを書き込みます。

詳細

注: このバージョンは、DATASETS プロシジャ構文の簡易バージョンです。構文およびその説明については、16 章: “DATASETS プロシジャ” (*Base SAS Procedures Guide*)を参照してください。

DATASETS プロシジャの出力では、ライブラリ参照名、エンジンおよびそのライブラリに関連付けられている物理名、さらにそのライブラリに含まれている SAS ファイルの名前とその他プロパティが表示されます。DATASETS プロシジャにより表示される、ファイル名やアクセス権限などの SAS ライブラリ情報の中には、動作環境およびエンジンに依存するものもあります。CONTENTS ステートメントにより生成される情報は、また、データセットに関連付けられたデバイス種別またはアクセス方法によって変わります。

CONTENTS ステートメントの中に DIRECTORY オプションを指定すると、ディレクトリ情報がログウィンドウと出力ウィンドウの両方に表示されます。

DATASETS プロシジャ内の CONTENTS ステートメントは、CONTENTS プロシジャと同じエンジンおよびホスト依存情報を生成します。

例

次の SAS コードの場合、2 つのデータセット、**classes.grades** および **classes.majors** が作成され、また **classes.majors** を入力データセットとして使用して PROC DATASETS が実行されます。

この例での出力のページ 1 は、CONTENTS ステートメント内の DIRECTORY オプションにより作成されます。この情報は、SAS ログの中にも表示されます。この出力の中のページ 2 は、データセット **classes.majors** について説明しており、SAS 出力の中にのみ表示されます。

```
libname classes '.';

data classes.grades (label='First Data Set');
input student year state $ grade1 grade2;
label year='Year of Birth';
format grade1 4.1;
datalines;
1000 1980 NC 85 87
1042 1981 MD 92 92
1095 1979 PA 78 72
1187 1980 MA 87 94
;

data classes.majors(label='Second Data Set');
input student $ year state $ grade1 grade2 major $;
label state='Home State';
format grade1 5.2;
datalines;
1000 1980 NC 84 87 Math
1042 1981 MD 92 92 History
1095 1979 PA 79 73 Physics
1187 1980 MA 87 74 Dance
1204 1981 NC 82 96 French
;

proc datasets library=classes;
contents data=majors directory;
run;
```

画面 17.2 DATASETS プロシジャからの出力

The SAS System

Directory	
Libref	CLASSES
Engine	V9
Physical Name	/r/sanyo.unx.sas.com/vol/vol810/u81/*user-id*
Filename	/r/sanyo.unx.sas.com/vol/vol810/u81/*user-id*
Inode Number	22543450
Access Permission	rwxr-xr-x
Owner Name	*user-id*
File Size (bytes)	32768

#	Name	Member Type	File Size	Last Modified
1	GRADES	DATA	16384	24Feb11:09:34:06
2	MAJORS	DATA	16384	24Feb11:09:34:06

The SAS System

The DATASETS Procedure

Directory	
Libref	CLASSES
Engine	V9
Physical Name	/r/sanyo.unx.sas.com/vol/vol810/u81/*user-id*
Filename	/r/sanyo.unx.sas.com/vol/vol810/u81/*user-id*
Inode Number	22543450
Access Permission	rwxr-xr-x
Owner Name	*user-id*
File Size (bytes)	32768

#	Name	Member Type	File Size	Last Modified
1	GRADES	DATA	16384	24Feb11:09:34:06
2	MAJORS	DATA	16384	24Feb11:09:34:06

The SAS System

The DATASETS Procedure

Data Set Name	CLASSES.MAJORS	**Observations**	5
Member Type	DATA	**Variables**	6
Engine	V9	**Indexes**	0
Created	Thu, Feb 24, 2011 09:34:06 AM	**Observation Length**	48
Last Modified	Thu, Feb 24, 2011 09:34:06 AM	**Deleted Observations**	0
Protection		**Compressed**	NO
Data Set Type		**Sorted**	NO
Label	Second Data Set		
Data Representation	HP_UX_64, RS_6000_AIX_64, SOLARIS_64, HP_IA64		
Encoding	latin1 Western (ISO)		

Engine/Host Dependent Information	
Data Set Page Size	8192
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	169
Obs in First Data Page	5
Number of Data Set Repairs	0
Filename	/r/sanyo.unx.sas.com/vol/vol810/u81/*user-id*/majors.sas7bdat
Release Created	9.0301B0
Host Created	HP-UX
Inode Number	25962217
Access Permission	rw-r--r--
Owner Name	*user-id*
File Size (bytes)	16384

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Label
4	grade1	Num	8	5.2	
5	grade2	Num	8		
6	major	Char	8		
3	state	Char	8		Home State
1	student	Char	8		
2	year	Num	8		

関連項目:

プロシジャ:

- 13 章: “CONTENTS プロシジャ” (*Base SAS Procedures Guide*)

OPTIONS プロシジャ: UNIX

SAS システムオプションの現在の設定を表示します。

UNIX 固有: UNIX 環境でのみ利用可能なオプション

参照項目: 5 章: “OPTIONS プロシジャ” (*SAS System Options: Reference*)

構文

PROC OPTIONS <*option(s)*>;

オプション引数

option

HOST | NOHOST
displays ホストオプション(HOST)のみを表示するか、またはポータブルオプション(NOHOST)のみを表示します。PORTABLE は、NOHOST の別名です。

詳細

基本

注:

このバージョンは、OPTIONS プロシジャ構文の簡易バージョンです。構文およびその説明については、5 章: “OPTIONS プロシジャ” (*SAS System Options: Reference*)を参照してください。

PROC OPTIONS は、すべての動作環境で利用可能なシステムオプション、また UNIX 環境で利用可能なシステムオプションの現在の設定を表示します。PROC OPTIONS ステートメント内に HOST オプションを指定すると、UNIX 環境でのみ利用可能なオプション(ホストオプション)が表示されます。PROC OPTIONS で表示されるオプション値は、SAS のデフォルト値、あなたのサイト管理者により指定されたデフォルト値、ご使用の構成ファイル内のデフォルト値、**システムオプション**ウィンドウまたは OPTIONS ステートメントを通じて現在のセッションの中で加えられた変更内容、そして場合によっては、SAS を実行中のデバイスによって変わります。

特定のオプションについての詳細は、“UNIX 版に固有の SAS システムオプション” (360 ページ)を参照してください。

オプションが設定された場所の識別

PROC OPTIONS ステートメント内で VALUE オプションを指定すると、そのオプションの範囲およびそのオプションの設定に使用した方法を確認できます。オプションの値が複数の方法で(たとえば、OPTIONS ステートメントの中で、または複数の構成ファイルの中で)設定されている場合、VALUE オプションを指定すると、どの方法で有効な値が決定されるかを確認できます。その方法が構成ファイルである場合には、VALUE オプションによりファイルパスが確認されます。

VALUE オプションを指定するには、次の PROC OPTIONS ステートメントを使用してください。

```
proc options option=option-name value;
```

詳細については、5 章: “OPTIONS プロシジャ” (*SAS System Options: Reference*)を参照してください。

関連項目:

"SAS 構成ファイルの処理の優先順位" (24 ページ)

PMENU プロシジャ: UNIX

SAS ソフトウェアで作成されるウィンドウ用のメニューファシリティを定義します。

UNIX 固有: TEXT ステートメント内の ATTR= および COLOR=オプションは影響しません。ITEM ステートメント内の ACCELERATE= および MNEMONIC=オプションは無視されます。

参照項目: 40 章: "PMENU プロシジャ" (*Base SAS Procedures Guide*)

構文

PROC PMENU <CATALOG=<*libref.*> *catalog*> <DESC '*entry-description*'>;

オプション引数

CATALOG=<*libref.*>*catalog*
PMENU エントリを保存するカタログを指定します。*libref* を省略する場合、PMENU エントリが、Sasuser ライブラリ内のカタログに保存されます。CATALOG= を省略すると、これらのエントリは、Sasuser.Profile カタログ内に保存されます。

DESC '*entry-description*'
このステップで作成された PMENU カタログエントリの説明を提供します。

詳細

注: このバージョンは、PMENU プロシジャ構文の簡易バージョンです。構文およびその説明については、40 章: "PMENU プロシジャ" (*Base SAS Procedures Guide*)を参照してください。

PMENU プロシジャは、Base SAS ソフトウェア内の WINDOW ステートメント、%WINDOW マクロステートメント、SAS/AF ソフトウェアの BUILD プロシジャ、または SAS/AF および SAS/FSP ソフトウェアの SAS Component Language (SCL) PMENU 機能を使用して作成されるウィンドウの PMENU 機能を定義します。

UNIX 環境では、次のオプションが無視されます。

- TEXT ステートメント内の ATTR=および COLOR=オプション。テキストおよび入力フィールドの色と属性は、**SASCOLOR** ウィンドウで指定した CPARMS 色で制御されます。詳細については、"UNIX 環境で色をカスタマイズする" (202 ページ)を参照してください。
- "TITLE Statement" (*SAS Statements: Reference*)内の ACCELERATE= および MNEMONIC=オプション。

PRINTTO プロシジャ: UNIX

SAS プロシジャ出力と SAS ログの割り当て先を定義します。

UNIX 固有: 次の項目の有効な値: *file-specification*

参照項目: 43 章: "PRINTTO プロシジャ" (*Base SAS Procedures Guide*)

構文

PROC PRINTTO <*option(s)*>;

オプション引数

option
PRINTTO プロシジャで使用するオプションを指定します。次のオプションを使用できます。

LOG=*file-specification*
絶対パス名(引用符内)、環境変数、ファイル参照名、またはカレントディレクトリ内のファイル(拡張子なし)を指定します。

PRINT=*file-specification*
絶対パス名(引用符内)、環境変数、ファイル参照名、またはカレントディレクトリ内のファイル(拡張子なし)を指定します。PRINTER デバイスタイプキーワードで定義されるファイル参照名を指定すると、出力が直接プリンタへ送信されます。

詳細

注: このバージョンは、PRINTTO プロシジャ構文の簡易バージョンです。構文およびその説明については、43 章: "PRINTTO プロシジャ" (*Base SAS Procedures Guide*)を参照してください。

次のステートメントでは、RUN ステートメントの後に生成されるすべての SAS ログエントリが、ファイル参照名 **myfile**:

```
filename myfile '/users/myid/mydir/mylog';
proc printto log=myfile;
run;
```

に関連付けられた外部ファイルへ送信されます。

myfile がファイル参照名として定義されていない場合には、PROC PRINTTO により、カレントディレクトリ内にファイル **myfile.log** が作成されます。

次のステートメントにより、RUN ステートメントの後に生成されるすべてのプロシジャ出力がこのファイルに送信されます。**/users/myid/mydir/myout**:

```
proc printto print='/users/myid/mydir/myout';
run;
```

次のステートメントでは、CONTENTS プロシジャからのプロシジャ出力が直接システムプリンタに送信されます。

```
filename myfile printer;
proc printto print=myfile;
run;
proc contents data=oranges;
run;
```

SAS ログおよびプロシジャ出力をその元のデフォルト割り当て先にリダイレクトするには、オプションなしで PROC PRINTTO を実行します。

```
proc printto;
run;
```

ファイル参照名 **myprint** および **mylog** が定義されていない場合には、次のステートメントが SAS プロシジャ出力を **myprint.lst** に、そしてすべてのログ出力をカレントディレクトリ内の **mylog.log** に送信します。

```
proc printto print=myprint log=mylog;
run;
```

ファイル参照名 **myprint** および **mylog** が定義されていた場合には、その出力は、これらのファイル参照名と関連付けられたファイルに送信されます。

関連項目:

"UNIX 環境における出力印刷の概要" (90 ページ)

SORT プロシジャ: UNIX

1 つ以上の変数により設定された SAS データ内のオブザベーションをソートし、次にソート済みのオブザベーションを新規 SAS データセット内に保存するか、または元のデータセットを置き換えます。

UNIX 固有: 利用可能なソートユーティリティ

参照項目: 55 章: "SORT プロシジャ" (*Base SAS Procedures Guide*)

構文

PROC SORT <*option(s)*> <*collating-sequence-option*>;

オプション引数

option

SORTSIZE=*memory-specification*
SORT プロシジャに利用可能な最大量のメモリを指定します。SORTSIZE=オプションについての詳細は、"SORTSIZE=オプション" (318 ページ)を参照してください。

TAGSORT
BY 変数およびオブザベーション番号を一時ファイルに保存します。TAGSORT オプションは、SyncSort を使用する UNIX ホストには影響しません。

TAGSORT オプションについての詳細は、"TAGSORT オプション" (319 ページ)を参照してください。

DETAILS
メモリ内でソートが実行されたかどうかの詳細メッセージを PROC SORT が SAS ログに書き込むことを指定します。(このオプションは、ステートメントオプションです。)

ソートがメモリ内で実行されなかった場合、SAS ログに書き込まれる詳細には、使用されたユーティリティファイルの数およびそのサイズが含まれます。

ヒント DETAILS オプションを使用すると、理想的な SORTSIZE 値を簡単に決定できます。

詳細

基本

注: このバージョンは、SORT プロシジャ構文の簡易バージョンです。構文およびその説明については、55 章: "SORT プロシジャ" (*Base SAS Procedures Guide*)を参照してください。

SORT プロシジャは、1 つ以上の文字変数または数値変数により設定された SAS データセット内のオブザベーションをソートし、元のデータセットを置き換えるか、または新規のソート済みデータセットを作成します。UNIX 環境では、デフォルトにより、SORT プロシジャは ASCII 照合順序を使用します。

SORT プロシジャは、SORTPGM システムオプションで指定されたソートユーティリティを使用します。ソートは、SAS、または **syncsort** ユーティリティによって実行できます。SORTSEQ および NODUPKEY オプションなど、SAS ソートユーティリティに利用可能なすべてのオプションを使用できます。状況により、NOEQUALS オプションを使用することでパフォーマンスを向上できる場合があります。ホストソートがサポートしていないオプションを指定すると、SAS ソートがかわりに使用されます。利用可能なすべてのオプションについての詳細は、55 章: "SORT プロシジャ" (*Base SAS Procedures Guide*)を参照してください。

SORTSIZE=オプション

PROC SORT に利用可能なメモリ量の制限

SORT プロシジャに利用可能なメモリ量を制限するには、PROC SORT ステートメント内に SORTSIZE=を使用することが可能です。このオプションを使用すると、SAS がデータセットをソートするのに必要な切替の量を減らせます。

注: SORTSIZE=オプションを指定しない場合、PROC SORT は、SORTSIZE システムオプションの値を使用します。SORTSIZE システムオプションは、コマンド行の中、または SAS 構成ファイルの中で定義できます。

SORTSIZE=オプションの構文

SORTSIZE=システムオプションの構文は、次の通りです。

SORTSIZE=*memory-specification*

memory-specification には、次のいずれかを使用できます。

n は、メモリの量をバイト数で指定します。

*n*K は、1KB の倍数でメモリの量を指定します。

*n*M は、メモリの量を 1MB の倍数で指定します。

*n*G メモリの量を 1GB の倍数で指定します。

SORTSIZE=オプションのデフォルト値

デフォルトの SAS 構成ファイルでは、SORTSIZE システムオプションの値に基づいてこのオプションが設定されます。あなたの動作環境のデフォルト値を参照するには、次のコードを実行してください。

```
proc options option=sortsize;
run;
```

SORTSIZE システムオプションのデフォルト値は、次のいずれかの方法でオーバーライドできます。

- PROC SORT ステートメント内で異なる SORTSIZE= 値を指定します。
- SORTSIZE システムオプションを新しい値に設定する OPTIONS ステートメントを送信します。
- SAS 起動中にコマンド行に SORTSIZE システムオプションを設定します。

SORTSIZE=オプションによるパフォーマンスの向上

SORTSIZE システムオプションでは、PROC SORT で利用可能なメモリの量が制限されます。通常、MEMSIZE オプションにより、SORTSIZE=オプションを、SAS プロセスで利用可能なメモリの量以下に設定する必要があります。

SORTSIZE=値が、メモリ内に設定されたデータセット全体を収めるのに十分な大きさの場合、ご使用のコンピュータシステムの利用可能な物理 RAM と同じ SORTSIZE=値になっていれば、最適なソートパフォーマンスを実現できます。物理 RAM が不足する場合、コンピュータは、過剰なメモリページをディスクへスワップし始め、メモリ使用によるパフォーマンス向上の効果を無効にします。

ソート対象のデータセット全体が、SORTSIZE で割り当てられたメモリ領域に収まらない場合、SAS はそのデータを格納するために一時ユーティリティファイルを作成します。この場合、SAS は、メモリではなくディスクスペースを使用してソートするよう調整されたソートアルゴリズムを使用します。これらの一時ユーティリティファイルは、SAS WORK の場所に配置されますが、これらのファイルは異なるファイルシステムを参照するよう設定できるので、UTILLOC システムオプションを使用するとき、I/O は阻害されません。

ソートする SAS データファイルをマシンの物理メモリに置くと、SAS でのソートが非常に効率的になります。SORTSIZE は、データファイルのサイズより大きくなるように設定します。データファイルが物理メモリに収まらない場合は、SORTSIZE を 1G 以下に設定します。さらに、SORTSIZE は、常に、MEMSIZE より 8M 以上小さい値に設定するようにしてください。

注: SORTSIZE システムオプションも使用できます。これは、PROC SORT ステートメント内に SORTSIZE=オプションを使用した場合とその効果は同じです。

TAGSORT オプション

大きな SAS データセットをソートするだけのディスク領域がない可能性がある場合は、PROC SORT ステートメント内に TAGSORT オプションを使用するのが便利です。TAGSORT オプションを指定すると、ソートキーのみ(つまり、BY ステートメントで指定された変数)および各オブザベーション用のオブザベーション番号が一時ユーティリティファイルに保存されます。ソートキーは、オブザベーション番号とともに、タグと呼ばれます。ソートプロセス完了時に、ソート順の入力データからレコードを抽出するのにこれらのタグが使用されます。このため、それらのソートキーの総バイト数がレコード長と比較して小さい場合、一時ディスク使用が大幅に低下します。

データセット(出力データセット)の追加コピーおよびタグを含むユーティリティファイルを保持できるだけのディスク領域を確保する必要があります。デフォルトでは、このユーティリティファイルは、Work ライブラリに保存されます。このディレクトリが非常に小さい場合、WORK システムオプションを使用して変更できます。詳細については、“WORK システムオプション: UNIX” (434 ページ)を参照してください。

注: TAGSORT オプションを使用すると一時ディスク使用が低下する可能性はあり、処理時間が長くなる可能性があります。ただし、利用可能なディスク領域が制限されたシステムでは、TAGSORT オプションを使用することで、それ以外ではソート不能な状況で、データセットのソートが可能になる場合があります。

PROC SORT のディスクスペースに関する注意点

PROC SORT を実行するのに必要なディスク領域の量を調べるには、次の情報を検討する必要があります。

入力 SAS データセット
: PROC SORT は、DATA=オプションで指定した SAS 入力データセットを使用します。

出力 SAS データセット
PROC SORT では、出力された SAS データセットは、OUT=オプションで指定した場所に保存されます。SAS シングルスレッドソートを使用する場合、OUT=オプションが指定されていないと、PROC SORT は、出力 SAS データセットを Work ライブラリに保存します。

ユーティリティファイル
UTILLOC システムオプションは、SAS マルチスレッドソートの使用時のみ、ユーティリティファイルの格納場所に影響します。SAS シングルスレッドソートでは、そのユーティリティファイルは、従来どおり **Work** ディレクトリに格納されます。通常、シングルスレッドソートの場合、追加のソートキーデータが各レコードと一緒に含まれるため、ユーティリティファイルは通常、無圧縮の入力 SAS データセットよりも少しだけ大きくなります。BY 変数があるオブザベーションの大部分からなる場合、または SORTSEQ=LINGUISTIC オプションを文字 BY 変数と一緒に使用する場合は、ユーティリティファイルが無圧縮の入力 SAS データセットよりもかなり大きくなることがあります。入力データセットが非常に大きいのに、ソートにはごくわずかなメモリのみが使用可能な場合、またはユーティリティファイルのページサイズが大きい場合などは、特別な状況によっては、ユーティリティファイルのサイズが倍になる可能性もあります。

SORT プロシジャがマルチスレッドソートを呼び出す場合、複数のユーティリティーファイルを異なる場所に配布することができます。ユーティリティファイルは、無圧縮の入力 SAS データセットとほぼ同じサイズです。通常、このサイズの単一ユーティリティファイルのみが必須です。ただし、特別な状況によっては、このサイズのユーティリティファイルが最大 2 つ使用される可能性があります。PROC SORT は、その 2 つのユーティリティファイルを、次の 2 つの最も最近に使用した場所へ配布します。

注: 各アプリケーションでユーティリティファイルの保存先として使用できる場所を指定するには、UTILLOC システムオプションを使用してください。

一時出力 SAS データセット
ソート時、PROC SORT は、OUT=オプションで指定されたディレクトリ(OUT=オプションが指定されていないときの入力 SAS データセットのディレクトリ)にその出力を作成します。この一時データセットのファイル名は、拡張子が.lck であることを除いて、元のデータセットと同じです。ソートが正常に完了すると、元のデータセットが削除され、その後一時データセットの名前が元のデータセットと一致するように変更されます。そのため、データセットの 2 つのコピーを保持できるだけの利用可能なディスク領域がターゲットディレクトリ内に必要です。

PROC SORT ステートメントに OVERWRITE オプションを指定することで、必要とされるディスク領域の量を減らせます。OVERWRITE が指定されていると、SORT では、可能な場合は、入力データセットを削除してから、置き換え出力データセットの書き込みを試行します。入力データセットを削除すると、最初にディスク領域が解放されます。このオプションは、必ず、バックアップされるデータセットまたは再構築可能なデータセットと一緒に使用してください。詳細については、55 章: “SORT プロシジャ” (*Base SAS Procedures Guide*)を参照してください。

PROC SORT のパフォーマンス調整

SAS による使用メモリ量の決定
MEMSIZE システムオプションにより、SAS プロセスで利用可能なメモリの量が制限されます。SORTSIZE システムオプションでは、PROC SORT で利用可能なメモリの量が制限されます。REALMEMSIZE システムオプションでは、SAS が利用できるようになる(仮想ではない)実メモリの量を指定します。

メモリ設定が MEMSIZE および SORTSIZE のデフォルト値を下回ることでソートおよび SAS のパフォーマンスに好ましくない影響を与える可能性はありますが、多量のメ

モリを使用可能にすることにメリットが得られない場合もあります。メモリの追加によりパフォーマンスが向上するかどうかを決定するキーは、そのソートがメモリに収まるかどうかです。ソートされたファイルが割り当てよりも多くのメモリを必要とする場合には、SORTSIZE 値を 64 - 512M の範囲で使用するのが通常に最適な結果につながります。SORTSIZE は、常に、MEMSIZE より 8M 以上小さい値に設定するようにしてください。

REALMEMSIZE システムオプションについての詳細は、“REALMEMSIZE システムオプション: UNIX” (409 ページ)を参照してください。

注: メモリ不足エラーが発生した場合は、MEMSIZE の値を増やしてください。詳細については、“MEMSIZE システムオプション: UNIX” (398 ページ)を参照してください。

REALMEMSIZE システムオプションの設定ガイドライン

使用するメモリ量を判断するには、REALMEMSIZE システムオプションを PROC SORT とともに使用してください。REALMEMSIZE 値は、ご使用のシステム上で利用可能なメモリ量を反映することに留意してください。最適なパフォーマンスを得るためには、すべてのアプリケーションのメモリ設定(ファイルキャッシュを含む)が、コンピュータ上の物理 RAM の量を一切超えないようにすることが必要です。REALMEMSIZE のデフォルト値は、MEMSIZE 設定の 80%です。REALMEMSIZE の設定値を非常に高くすると、PROC SORT が実際に利用可能なメモリを超えるメモリを使用する可能性があります。メモリの過大な使用は、過剰なページングを引き起こし、システムのパフォーマンスに悪影響を与えます。

通常に、REALMEMSIZE は、実行時に SAS に利用可能と予想される物理メモリの量(スワップスペースを含まない)に設定してください。コンピュータに実装された物理メモリの量から、各アプリケーションおよびオペレーティングシステムの実行により使用される量を差し引いた量を開始値として使用することをお勧めします。現在の環境で最適なパフォーマンスが得られるまで、REALMEMSIZE 値を試してください。場合によっては、REALMEMSIZE 値を非常に低くすることで、最適なパフォーマンスが得られる場合があります。値を低くことで SAS が使用するメモリが少なくなり、オペレーティングシステムが I/O キャッシュの実行により多くのメモリを確保できるようになる場合があります。

詳細については、“REALMEMSIZE システムオプション: UNIX” (409 ページ)を参照してください。

パフォーマンスに影響する他のオプションの使用

THREADS システムオプションは、スレッド対応プロシジャがスレッドを使用するかどうかを制御します。これは、PROC SORT の中で、システムオプションとしても、またプロシジャのオーバーライドとしても利用できます。

CPUCOUNT オプションは、THREADS オプションに直接関連しており、コンピュータ上の CPU 数にデフォルトで設定されます。ご使用のファイルシステムおよび同時ユーザー数によって、多くの CPU をもつマシン上では CPUCOUNT を低くすることでメリットが得られる場合があります。CPUCOUNT の値が ACTUAL に等しい場合は、SAS は、SAS が実行中の動作環境に関連付けられた物理 CPU 数を返します。

UTILLOC システムオプションでは、ユーティリティファイルの拡散が可能であり、これは I/O のバランスさせるのに良好なオプションです。

PROC SORT ステートメント内に指定された DETAILS オプションを使用すると、ソートがメモリ内で実行されたかどうかを示す SAS ログに対して PROC SORT がメッセージを書き込むようにします。ソートがメモリ内で実行されなかった場合、書き込まれる詳細には、ユーティリティファイルの数およびそのサイズが含まれます。

THREADS、CPUCOUNT および UTILLOC システムオプションについての詳細は、*SAS システムオプション: リファレンス*を参照してください。

独自の照合順序の作成
各自の照合順序を指定したい場合、または提供された照合順序を変更したい場合は、TRANTAB プロシジャを使用して変換テーブルを作成または変更してください。詳細については、19 章: “TRANTAB プロシジャ” (*SAS National Language Support (NLS): Reference Guide*)を参照してください。各自用の変換テーブルを作成すると、それらは、Sasuser.Profile カタログに保存され、またそれらは Host カタログ内に保存されたのと同じ名前のすべての変換テーブルをオーバーライドします。

注: システムマネージャは、新規に作成された各テーブルを Profile カタログから Host カタログへコピーすることにより、Host カタログを変更できます。そうすることで、新規の、または変更された変換テーブルに、すべてのユーザーがアクセスできるようになります。

SAS ウィンドウ環境を使用しており、また Host カタログ内に保存された照合順序の名前を確認したい場合は、任意のウィンドウから次のコマンドを発行してください。

```
catalog sashelp.host
```

SAS ウィンドウ環境を使用していない場合には、次のステートメントを発行して、Host カタログ内のコンテンツのリストを生成してください。

```
proc catalog catalog=sashelp.host;
contents;
run;
```

種別 TRANTAB のエントリは、その照合順序です。

特定の変換テーブルのコンテンツを確認するには、次のステートメントを使用します。

```
proc trantab table=table-name;
list;
run;
```

照合順序のコンテンツは、SAS ログ内に表示されます。

ホストソートユーティリティの指定

ホストソートの使用について
SAS では、UNIX 環境において、**syncsort** と呼ばれる 1 つのホストソートユーティリティがサポートされています。このソートアプリケーションは、SAS ソートの代替ソートアルゴリズムとして使用できます。SAS は、SORTNAME、SORTPGM、SORTCUT および SORTCUTP システムオプション用に設定された値によってどのソートを使用するかを判断します。

ホストソートユーティリティのソートアルゴリズムとしての設定
ホストソートユーティリティをソートアルゴリズムとして指定するには、次のプロシジャを実行します。

1. ホストユーティリティ(**syncsort**)の名前を SORTNAME システムオプションに指定します。
2. SAS に対してホストソートユーティリティをいつ使用するかを指定するには、SORTPGM システムオプションを使用します。
   - SORTPGM=HOST を指定すると、SAS は常にホストソートユーティリティを使用することを優先します。
   - SORTPGM=BEST を指定すると、SAS は、目下の状況下で最善のソート方法を選択します(SAS ソートまたはホストソートのいずれか)。

サイズまたはオブザベーションに基づくソート

SAS が使用するソートルーチンは、データセット内のオブザベーション数またはデータセットのサイズから作成できます。SORTPGM システムオプションが BEST に設定されていると、SAS は次の優先順位に基づいて最初に利用可能で関連のあるソーティングアルゴリズムを使用します。

- ホストソートユーティリティ
- SAS ソートユーティリティ

SORTCUT システムオプションは、データセット内のオブザベーション数に基づいて決められます。SORTCUTP システムオプションは、そのデータセットのサイズに基づいて決められます。SAS は SORTCUT および SORTCUTP システムオプションの値をみて、どのソートルーチンを使用するかを判断します。オブザベーション数が SORTCUT の値以上であれば、SAS はホストソートユーティリティを使用します。データセット内のバイト数が SORTCUTP の値より大きい場合は、SAS はホストソートユーティリティを使用します。

SORTCUT および SORTCUTP がゼロに設定されていると、SAS は SAS ソートユーティリティを使用します。両方のシステムオプションを指定する場合で、両方の条件に適合する場合には、SAS はホストソートユーティリティを使用します。

次の OPTIONS ステートメントが有効な場合、ホストソートユーティリティ(**syncsort**)は、オブザベーション数が 500 以上の場合に使用されます。

```
options sortpgm=best sortcut=500;
```

この例では、データセットのサイズが 40M を超える場合に、ホストソートユーティリティが使用されます。

```
options sortpgm=best sortcutp=40M;
```

これらのソートオプションについての詳細は、“SORTCUT システムオプション: UNIX” (419 ページ)、“SORTCUTP システムオプション: UNIX” (420 ページ) および “SORTPGM システムオプション: UNIX” (422 ページ)を参照してください。

ホストソートユーティリティにより使用される一時ファイルの場所の変更

デフォルトでは、ホストソートユーティリティは、一時ファイル用の-WORK オプションの中に指定された場所を使用します。これらの一時ファイルの場所を変更するには、SORTDEV システムオプションを使用することによって 1 つの場所を指定してください。ここに例を示します。

```
options sortdev="/tmp/host";
```

詳細については、“SORTDEV システムオプション: UNIX” (421 ページ)を参照してください。

ホストソートユーティリティにオプションを渡す

ソートユーティリティのオプションを指定するには、SORTANOM システムオプションを使用してください。有効なオプションのリストについては、“SORTANOM システムオプション: UNIX” (418 ページ)を参照してください。

ホストソートユーティリティにパラメータを渡す

ソートユーティリティに各パラメータを渡すには、SORTPARM システムオプションを使用してください。指定できるパラメータは、ホストソートユーティリティに依存します。詳細については、“SORTPARM システムオプション: UNIX” (422 ページ)を参照してください。

ホストソートユーティリティを使用して SORTSEQ=オプションを指定する

SORTSEQ=オプションを使用すると、ソートの照合順序を指定できます。有効な値のリストについては、55 章: “SORT プロシジャ” (*Base SAS Procedures Guide*)を参照してください。

注意:
データをソートするのにホストソートユーティリティを使用している場合、SORTSEQ=オプションを指定すると、ソート順序変換テーブルおよびその逆が 1 対 1 のマッピングになっていなければ、文字 BY 変数を破損する可能性があります。 つまり、ソートが起動するには、変換テーブルは、各文字を一意の重みにマッピングする必要があり、また逆テーブルは各重みを一意の文字にマッピングする必要があります。

ご使用の変換テーブルが 1 対 1 のマッピングを行っていない場合には、次のいずれかの方法によりソートを実行できます。

- 1 対 1 のマッピングを行う変換テーブルを作成します。1 対 1 のマッピングを行う変換テーブルを作成した後は、TRANTAB プロシジャを使用して対応する逆テーブルを簡単に作成できます。変換テーブルが 1 対 1 のマッピングでない場合は、逆テーブルを作成しようとすると、SAS ログに次の NOTE が書き込まれます。

  ```
  NOTE: This table cannot be mapped one to one.
  ```

  詳細については、19 章: “TRANTAB プロシジャ” (*SAS National Language Support (NLS): Reference Guide*)を参照してください。
- SAS ソートを使用します。SORTPGM システムオプションを使用して SAS ソートを指定できます。詳細については、“SORTPGM システムオプション: UNIX” (422 ページ)を参照してください。
- ホストソートユーティリティの照合順序オプションを指定します。詳細については、ご使用のホストソートユーティリティのドキュメントを参照してください。
- 単一 BY 変数でビューを作成します。例については、“例: 単一の BY 変数を使用してビューを作成する” (324 ページ)を参照してください。

注: このいずれかの方法を使用した後、NOTSORTED オプションまたは NOBYSORTED システムオプションのいずれかを使用して後続 BY 処理を実行する必要がある場合があります。NOTSORTED オプションについての詳細は、“BY Statement” (*SAS Statements: Reference*)を参照してください。NOBYSORTED システムオプションの詳細については、“BYSORTED System Option” (*SAS System Options: Reference*)を参照してください。

例: 単一の BY 変数を使用してビューを作成する

次の例は、単一 BY 変数を使用したビューの作成方法を示しています。SAS は、SORTPGM システムオプションに BEST 引数を使用してデータをソートします。BEST を使用することで、SAS は、ホストソートまたは SAS ソートのいずれかを選択します。(SAS/ACCESS エンジンを使用する場合、DBMS でもソートを実行できます。)

```
options sortpgm=best msglevel=i;

data one;
input name $ age;
datalines;
Anne 35
ALBERT 10
JUAN 90
Janet 5
Bridget 23
BRIAN 45
```

```
;

data oneview / view=oneview;
set one;
name1=upcase(name);
run;

proc sort data=oneview out=final(drop=name1);
by name1;
run;

proc print data=final;
run;
```

ログ 17.1 *ログ出力*

注：SAS スレッドソートが使用されました。

画面 17.3 *単一 BY 変数でビューを作成した場合の出力*

The SAS System

Obs	name	age
1	ALBERT	10
2	Anne	35
3	BRIAN	45
4	Bridget	23
5	Janet	5
6	JUAN	90

関連項目:

プロシジャ:

- 19 章: “TRANTAB プロシジャ” (*SAS National Language Support (NLS): Reference Guide*)

システムオプション:

- “MEMSIZE システムオプション: UNIX” (398 ページ)
- “REALMEMSIZE システムオプション: UNIX” (409 ページ)
- “SORTANOM システムオプション: UNIX” (418 ページ)
- “SORTCUT システムオプション: UNIX” (419 ページ)
- “SORTCUTP システムオプション: UNIX” (420 ページ)
- “SORTDEV システムオプション: UNIX” (421 ページ)
- “SORTNAME システムオプション: UNIX” (421 ページ)
- “SORTPARM システムオプション: UNIX” (422 ページ)
- “SORTPGM システムオプション: UNIX” (422 ページ)

- “SORTSIZE システムオプション: UNIX” (423 ページ)
- “UTILLOC= System Option” (*SAS System Options: Reference*)

18 章

UNIX 版に固有のステートメント

UNIX 版に固有の SAS ステートメント

このセクションでは、UNIX 環境に固有の動作や構文を表す SAS ステートメントについて説明します。各ステートメントの説明には、そのステートメントのどの要素が UNIX 版に固有であるのかを簡単に説明する"UNIX 固有"セクションが含まれています。"UNIX 固有"の情報が"すべて"の場合、そのステートメントはこのドキュメントでのみ説明されています。それ以外の場合、ステートメントはこのドキュメントの他に *SAS ステートメント: リファレンス*でも説明されています。

ディクショナリ

ABORT ステートメント: UNIX

現在の DATA ステップ、SAS ジョブまたは SAS セッションの実行を中止します。

該当要素: DATA ステップ

UNIX 固有: 次の項目の値: *n*

参照項目: “ABORT Statement” (*SAS Statements: Reference*)

構文

ABORT <ABEND | RETURN> <*n*>;

詳細

n オプションを使用して、SAS の実行中止時にシェルに戻される終了ステータスコードの値を指定できます。*n* の値は 0 - 255 の範囲で指定できます。通常は、プログラムがエラーの発生なしに実行したことを示す、0 のリターンコードが使用されます。0 より大きいリターンコードは、順序立ててより深刻になっていくエラー状態を示します。0 - 6 のリターンコードと 977 より大きいコードは、SAS での使用のために予約されます。

関連項目:

“UNIX 環境で SAS ジョブの完了ステータスの特定” (25 ページ)

ATTRIB ステートメント: UNIX

出力形式と入力形式、ラベルまたは長さを、1 つ以上の変数に関連付けます。

該当要素: DATA ステップ

UNIX 固有: 長さの仕様

参照項目: “ATTRIB Statement” (*SAS Statements: Reference*)

構文

ATTRIB *variable-list*-1 *attribute-list*-1 <*...variable-list-n attribute-list-n*> ;

必須引数

attribute-list

LENGTH=<$>*length*
variable-list で変数の長さを指定します。数値変数に指定できる長さの最小値は、ユーザーのシステムが使用する浮動小数点形式によって異なります。ほとんどのシステムでは IEEE 浮動小数点形式が使用され、最小値は 3 バイトです。

詳細

注: ATTRIB ステートメント構文は単純化された形です。完全な構文とその説明については、“ATTRIB Statement” (*SAS Statements: Reference*)を参照してください。

関連項目:

“UNIX 環境の数値変数の長さと精度” (223 ページ)

FILE ステートメント: UNIX

PUT ステートメントの現在の出力ファイルを指定します。

該当要素: DATA ステップ

UNIX 固有: 次の項目の有効な値: *file-specification*、*host-options* および *encoding-value*

参照項目: “FILE Statement” (*SAS Statements: Reference*)

構文

FILE *file-specification* <PERMISSION='*permission-value*'> <ENCODING='*encoding-value*'> <*options*> <*host-options*>;

必須引数

file-specification

“UNIX 環境で外部ファイルまたはデバイスにアクセスする” (69 ページ)で説明されているファイル仕様形のうち、どれでも使用できます。

オプション引数

PERMISSION='*permission-value*'

指定ファイル参照名に対して設定する権限を指定します。2 つ以上の権限値セットを指定するには、引用符内でカンマを使用して区切ります。

次の形式で *permission-value* を指定します。

`A::<trustee_type>::<permissions>`

‘A’ は、これらがアクセス権限であることを示します。現在サポートされている値は他にありません。

trustee_type が取り得る値を次に示します。

u ユーザー

g グループ(ファイルのグループ所有者)

o その他(他のユーザーすべて)

権限値は、英文字 *r* (読み取り)、*w* (書き込み)、および *x* (実行)の値をこの順序で取ります。これらの権限の 1 つを付与しない場合は、その位置に ‘-’ を入力します (たとえば、*r-x* や *rw-*など)。

ファイル参照名に対して読み取り、書き込み、および実行権限を得たいとします。また、ファイルのグループ所有者に対して読み取りおよび実行権限を指定するとします。最後に、他のユーザーすべてがファイルに対して読み取り権限のみを得ることを許可します。これらのオプションは、次のように指定できます。

```
permission='A::u::rwx,A::g::r-x,A::o::r--'
```

3 つのトラスティの種類すべてに対して権限値を指定します。権限値のリストで省略したトラスティの種類はいずれも、指定ファイル参照名へのアクセスをすべて拒否されます。たとえば、次の権限値を使用したとします。

```
permission='A::u::rwx,A::g::r-x'
```

この場合、指定ファイルにアクセスできるのは所有者およびグループ所有者のみになります。所有者およびグループ所有者以外のユーザーはいずれもファイルへのアクセスをすべて拒否されます。

ENCODING='*encoding-value*'

出力ファイルへの書き込み時に使用するエンコーディングを指定します。ENCODING=の値は、出力ファイルのエンコーディングが、現在のセッションのエンコーディングとは異なることを示します。

出力ファイルにデータを書き込むときは、セッションのエンコーディングから指定されたエンコーディングへ、SAS によってデータがトランスコードされます。

有効なエンコーディング値については、“Overview to SAS Language Elements That Use Encoding Values” (*SAS National Language Support (NLS): Reference Guide* 22 章)を参照してください。

options
すべての動作環境で使用可能な FILE ステートメントの任意のオプションです。これらのオプションの説明については、“FILE Statement” (*SAS Statements: Reference*)を参照してください。

host-options
UNIX 環境に固有です。これらのオプションでは次のいずれもが使用できます。

BLKSIZE=
BLK=
1 回の I/O 操作で物理的に書き込まれるバイト数を指定します。デフォルト値は 8K です。最大値は 1G–1 です。

TERMSTR=
UNIX でフォーマットされたファイルでの行末またはレコード区切り文字を制御します。TERMSTR=オプションで有効な値は次のとおりです。

CRLF キャリッジリターン、ラインフィード。

LF ラインフィード。このパラメータは、UNIX でフォーマットされたファイルの読み込みに使用されます。LF がデフォルトです。

LRECL=
論理レコード長を指定します。この値は、有効なレコード形式によって異なります(RECFM)。SAS 9.4 では、LRECL=のデフォルト値は 32,767 です。固定長レコード(RECFM=F)を使用する場合、LRECL=のデフォルト値は 256 になります。最大レコード長は 1G です。

- RECFM=F の場合、LRECL=オプションの値により各出力レコードの長さが決定されます。出力レコードは、切り捨てられるか、指定されたサイズに合うように空白が追加されます。

  注: RECFM=F の場合、SAS 9.4 が SAS の以前のバージョンと通信するには、LRECL=は 256 に設定する必要があります。

- RECFM=N の場合、LRECL=オプションの値は 256 以上にする必要があります。
- RECFM=V の場合、LRECL=オプションの値により最大レコード長が決定されます。指定された長さを超えるレコードは、複数のレコードに分割されます。

MOD
ファイルに書き込まれたデータをそのファイルに追加する必要があることを示します。

NEW | OLD
新規ファイルまたは既存ファイルのどちらを出力に使用するかを指定します。NEW を指定すると、出力用に新規のファイルが開きます。ファイルがすでに存在する場合は、削除された後でファイルが再作成されます。OLD を指定すると、以前のファイルの内容が置き換えられます。NEW がデフォルト値です。

RECFM=
レコード形式を指定します。RECFM=オプションの値は次のとおりです。

D デフォルトのレコード形式(可変長形式)。

F 固定長形式。つまり、各レコードの長さは同じです。キャリッジコントロール文字が含まれる外部ファイルには、RECFM=F を使用しないでください。

N バイナリ形式。ファイルはバイトストリームで構成され、レコード境界はありません。

P プリント形式。SAS で、キャリッジコントロール文字が書き込まれます。

V 可変長形式。各レコードは改行文字で終わります。

S370V 可変長 S370 レコード形式(V)。

S370VB 可変長ブロック S370 レコード形式(VB)。

S370VBS スパンレコード可変長ブロック S370 レコード形式(VBS)。

UNBUF
その後の FILE ステートメントでは、バッファされた書き込みをファイルに対して行わないよう SAS に指示します。このオプションは特に、データ収集デバイスに書き込んでいるときに適用されます。

詳細

ENCODING=オプションは、予約済みのファイル参照名ではないファイル仕様が FILE ステートメントに含まれている場合のみ有効です。FILE ステートメントに、ENCODING=引数および予約済みのファイル参照名 LOG または PRINT が *file-specification* として含まれている場合は、エラーメッセージが表示されます。FILE ステートメントの ENCODING=値は、ENCODING=システムオプションの値をオーバーライドします。

SAS セッション内から **umask** コマンドを発行することによって、出力ファイルの権限を設定できます。詳細については、“SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)を参照してください。

関連項目:

“外部ファイルとデバイスの使用” (68 ページ)

FILENAME ステートメント: UNIX

SAS ファイル参照名を外部ファイルまたは出力デバイスに関連付けます。ファイル参照名と外部ファイルの関連付けを解除します。外部ファイルの属性を一覧表示します。

該当要素: 任意の場所

制限事項: SAS がロックダウン状態の場合、UNIX では、FILENAME ステートメントアクセス方法の EMAIL、FTP、HADOOP、SOCKET、URL がアクセス不能(無効)になります。SAS Server 管理者は、これらのアクセス方法の 1 つ以上を再有効化して、SAS がロックダウン状態の場合もアクセス可能にできます。詳細については、“SAS Processing Restrictions for Servers in a Locked-Down State” (*SAS Language Reference: Concepts* 2 章)を参照してください。

UNIX 固有: *device-type*、*external-file*、*host-options*、*encoding-value*

参照項目: “FILENAME Statement” (*SAS Statements: Reference*)

構文

FILENAME *fileref* <*device-type*> '*external-file*' <PERMISSION='*permission-value*'> <ENCODING='*encoding-value*'> <'*host-options*'> <LOCKINTERNAL= AUTO | SHARED>;

FILENAME *fileref device-type*<'*external-file*'> <ENCODING='*encoding-value*'> <'*host-options*'> <LOCKINTERNAL= AUTO | SHARED>;

FILENAME *fileref* ('*pathname-1*' … '*pathname-n*') <ENCODING='*encoding-value*'> <'*host-options*'> <LOCKINTERNAL= AUTO | SHARED>;

FILENAME *fileref directory-name* <ENCODING='*encoding-value*'> <LOCKINTERNAL= AUTO | SHARED>;

FILENAME *fileref* <*access-method*> '*external-file*' *access-information*;

FILENAME *fileref* CLEAR | _ALL_ CLEAR;

FILENAME *fileref* LIST | _ALL_ LIST;

必須引数

fileref
ファイルを参照するために使用する名前です。UNIX 版では、*fileref* の値として環境変数を使用できます。

注: 環境変数によって定義されるファイル参照名を削除することはできません。環境変数によって定義されるファイル参照名は、SAS セッション全体に割り当てられます。

詳細については、"UNIX 環境で環境変数を使用してファイル参照名を割り当てる" (77 ページ)を参照してください。

'external-file'
デバイスタイプによって異なります。"FILENAME ステートメントのデバイス情報" (336 ページ) 各デバイスに対応する情報を表示します。UNIX ファイル名は大小文字が区別されるので注意してください。詳細については、"UNIX 環境でのパス名の指定" (70 ページ)を参照してください。

注: ファイル名の先頭に空白があれば、その空白は削除されます。

オプション引数

device-type
出力用デバイス(ディスク、ターミナル、プリンタ、パイプなど)を指定します。device-type キーワードは、*fileref* の後ろで、かつ *pathname* の手前に記述する必要があります。"FILENAME ステートメントのデバイス情報" (336 ページ) は有効なデバイスタイプを説明します。DISK がデフォルトのデバイスタイプです。ファイル参照名と DISK ファイルを関連付ける場合は、デバイスタイプを指定する必要はありません。

PERMISSION='*permission-value*'
指定ファイル参照名に対して設定する権限を指定します。2 つ以上の権限値セットを指定するには、引用符内でカンマを使用して区切ります。

次の形式で *permission-value* を指定します。

```
A::<trustee_type>::<permissions>
```

'A' は、これらがアクセス権限であることを示します。現在サポートされている値は他にありません。

trustee_type が取り得る値を次に示します。

u ユーザー

g グループ(ファイルのグループ所有者)

o その他(他のユーザーすべて)

権限値は、英文字 *r* (読み取り)、*w* (書き込み)、および *x* (実行)の値をこの順序で取ります。これらの権限の 1 つを付与しない場合は、その位置に '-' を入力します(たとえば、*r-x* や *rw-*など)。

ファイル参照名に対して読み取り、書き込み、および実行権限を得たいとします。また、ファイルのグループ所有者に対して読み取りおよび実行権限を指定するとします。最後に、他のユーザーすべてがファイルに対して読み取り権限のみを得ることを許可します。これらのオプションは、次のように指定できます。

```
permission='A::u::rwx,A::g::r-x,A::o::r--'
```

3 つのトラスティの種類すべてに対して権限値を指定します。権限値のリストで省略したトラスティの種類はいずれも、指定ファイル参照名へのアクセスをすべて拒否されます。たとえば、次の権限値を使用したとします。

```
permission='A::u::rwx,A::g::r-x'
```

この場合、指定ファイルにアクセスできるのは所有者およびグループ所有者のみになります。所有者およびグループ所有者以外のユーザーはいずれもファイルへのアクセスをすべて拒否されます。

ENCODING='*encoding-value*'

外部ファイルに対して書き込みまたは読み込みを行うときに使用するエンコーディングを指定します。ENCODING=の値は、外部ファイルのエンコーディングが、現在のセッションのエンコーディングとは異なることを示します。

外部ファイルのデータを読み取るときは、指定されたエンコーディングからセッションのエンコーディングへ、SAS によってデータがトランスコードされます。外部ファイルにデータを書き込むときは、セッションエンコーディングから指定されたエンコーディングへ、SAS によってデータがトランスコードされます。

注: UPRINTER デバイスタイプは ENCODING オプションをサポートしません。

有効なエンコーディング値については、"Overview to SAS Language Elements That Use Encoding Values" (*SAS National Language Support (NLS): Reference Guide* 22 章)を参照してください。

'*host-options*'

UNIX 環境に固有です。これらのオプションでは次のいずれもが使用できます。

BLKSIZE=
BLK=
1 回の I/O 操作で物理的に書き込まれるまたは読み込まれるバイト数を指定します。デフォルト値は 64KB です。最大値は 1G–1 です。RECFM=S370VBS を指定する場合は、255 文字を超えるレコードに発生するエラーを回避するために BLKSIZE=32,760 を指定する必要があります。

TERMSTR=
UNIX でフォーマットされたファイルでの行末またはレコード区切り文字を制御します。TERMSTR=オプションで有効な値は次のとおりです。

CR キャリッジリターン。

CRLF キャリッジリターン、ラインフィード。

LF ラインフィード。このパラメータは、UNIX でフォーマットされたファイルの読み込みに使用されます。LF がデフォルトです。

UNIX で読み込むファイルに書き込む場合は、TERMSTR=LF を指定します。

LRECL=
論理レコード長を指定します。この値は、有効なレコード形式によって異なります(RECFM)。SAS 9.4 では、LRECL=のデフォルト値は 32,767 です。固定長レコード(RECFM=F)を使用する場合、LRECL=のデフォルト値は 256 になります。最大長は 1G です。

- RECFM=F の場合、LRECL=オプションの値は、1 件のレコードとして読み込まれるバイト数または各出力レコードの長さを決定します。出力レコードは、切り捨てられるか、指定されたサイズに合うように空白が追加されます。

  注: RECFM=F の場合、SAS の以前のバージョンでのデフォルト値を使用して作成された固定長レコードを読み込むときには、LRECL=を 256 に設定する必要があります。

- RECFM=N の場合、LRECL=オプションの値は 256 以上にする必要があります。
- RECFM=V の場合、LRECL=オプションの値により最大レコード長が決定されます。指定された長さを超えるレコードは、出力上では複数のレコードに分割され、入力上では切り捨てられます。
- RECFM=S370VBS を指定する場合は、255 文字を超えるレコードに発生するエラーを回避するために LRECL=32,760 を指定する必要があります。

MOD
ファイルに書き込まれたデータをそのファイルに追加する必要があることを示します。

NEW | OLD
新規ファイルまたは既存ファイルのどちらを出力に使用するかを指定します。NEW を指定すると、出力用に新規のファイルが開きます。ファイルがすでに存在する場合は、削除された後でファイルが再作成されます。OLD を指定すると、以前のファイルの内容が置き換えられます。NEW がデフォルト値です。

RECFM=
レコード形式を指定します。RECFM=オプションの値は次のとおりです。

D	デフォルトのレコード形式(可変長形式)。
F	固定長形式。つまり、各レコードの長さは同じです。キャリッジコントロール文字が含まれる外部ファイルには、RECFM=F を使用しないでください。
N	バイナリ形式。ファイルはバイトストリームで構成され、レコード境界はありません。N は、PIPE デバイスタイプでは無効です。LRECL オプションを指定しない場合、デフォルトではファイルから 256 バイトが読み込まれます。
P	プリント形式。On 出力では、キャリッジコントロール文字が書き込まれます。
V	可変長形式。各レコードは改行文字で終わります。
S370V	可変長 S370 レコード形式(V)。
S370VB	可変長ブロック S370 レコード形式(VB)。
S370VBS	スパンレコード可変長ブロック S370 レコード形式(VBS)。RECFM=S370VBS を指定する場合は、255 文字を超えるレコードに発生するエラーを回避するために BLKSIZE=32,760 および LRECL=32,760 を指定する必要があります。

RECFM=オプションは、入力および出力の両方に使用されます。

LOCKINTERNAL=AUTO | SHARED
FILENAME ステートメントにリスト表示されているファイルに使用する、SAS System のロックを指定します。LOCKINTERNAL は次のいずれかの値を持ちます。

AUTO
SAS セッションで、あるユーザーがあるファイルへの書き込みアクセスを持っている場合、他のユーザーはそのファイルへの読み込みまたは書き込みアクセスを持てないように、ファイルをロックします。あるユーザーがあるファイルへの読み込みアクセスを持っている場合、他のユーザーにはそのファイルへの書き込みアクセスを持つことはできませんが、読み込みアクセスは複数のユーザーが持つことができます。

SHARED
SAS セッションで、2 人のユーザーが同時に同じファイルへのアクセスを持てないように、ファイルをロックします。その場合ファイルは、書き込みアクセスを持っている 1 人のユーザーと読み込みアクセスを持っている複数のユーザーが、同時に共有できます。

デフォルト AUTO

UNBUF
その後の FILE ステートメントでは、バッファされた書き込みをファイルに対して行わないよう SAS に指示します。このオプションは、特にユーザーがデータ収集デバイスに対して読み込みまたは書き込みを行っているときに適用されます。*SAS ステートメント: リファレンス*で説明されているとおり、INFILE ステートメントでのバッファされた読み込みも行わないようにします。

'pathname-1' ... 'pathname-n'
ユーザーが同一のファイル参照名を使用してアクセスしようとしているファイルのパス名です。ファイル名を連結したいときは、この形の FILENAME ステートメントを使用します。ファイル名の連結は DISK ファイルの場合のみ可能です。したがって、*device-type* を指定する必要はありません。カンマまたは空白スペースを使ってパス名を区切ります。それぞれのパス名を引用符で囲みます。表 2.3 (52 ページ) パス名を指定するときに使用できる文字の置換を表示します。定義の対象となるファイル参照が入力で使用される場合は、ワイルドカードが使用できます(“ワイルドカードのパス名への使用(入力のみ)” (71 ページ)を参照)。UNIX ファイル名は大小文字が区別されるので注意してください。

directory-name
アクセスするファイルが含まれるディレクトリを指定します。詳細については、“(集計構文を使用して)ディレクトリにファイル参照名を割り当てる” (76 ページ)を参照してください。

access-method
物理ファイルではない入出力デバイスまたは位置をファイル参照名が示す場合に使われるアクセス方法やデバイスタイプを指定します。“FILENAME ステートメントのデバイス情報” (336 ページ) では、これらのアクセス方法に必要となる情報を記述しています。

access-information
アクセス方法によって異なります。“FILENAME ステートメントのデバイス情報” (336 ページ) 各アクセス方法に対応する情報を表示します。

CLEAR
指定されたファイル参照名をクリアします。_ALL_を指定した場合は、現在定義されているファイル参照名をすべてクリアします。

注: 環境変数によって定義されるファイル参照名を削除することはできません。環境変数によって定義されるファイル参照名は、SAS セッション全体に割り当てられます。

ALL
現在定義されているすべてのファイル参照名を参照します。このキーワードは、ファイル参照名をリストまたはクリアするときに使用します。

LIST
SAS ログに指定されたファイル参照名のパス名を書き込みます。また、_ALL_を指定した場合は、現在定義されているすべてのファイル参照名の定義をリスト表示します。環境変数として定義されているファイル参照名は、それらがすでに SAS ステートメントで使用されている場合のみ、表示されます。Bourne シェルまたは Korn シェルを使用している場合、SAS ではすでに開かれたファイルの名前が判断できないため、ファイル名のかわりに次の文字列が表示されます。

```
<File Descriptor number>
```

詳細については、"UNIX 環境で環境変数を使用してファイル参照名を割り当てる" (77 ページ)を参照してください。

詳細

ファイルロック

外部ファイルのファイルロックは、LOCKINTERNAL オプションによって FILENAME ステートメントレベルで制御されます。LOCKINTERNAL に AUTO 値(デフォルト)を使用する場合、SAS では、書き込みアクセスを持っている 1 人のユーザーのために排他的にファイルがロックされます。あるいは、読み込みアクセスを持っている複数のユーザーのために、非排他的にファイルがロックされます。たとえば、UPDATE または OUTPUT モードでファイルが開かれた場合、内部プロセスからのその他のアクセスすべてがブロックされます。ファイルが INPUT モードで開かれた場合、複数のユーザーがそのファイルを読み込むことができますが、UPDATE 機能および OUTPUT 機能はブロックされます。

LOCKINTERNAL に SHARED 値を使用する場合は、1 人のユーザーがファイルの書き込みアクセスを許可され、複数のユーザーがそのファイルの読み込みを許可されます。

FILENAME ステートメントのデバイス情報

次の表の、デバイスタイプまたはアクセス方法と、関連する外部ファイルとの関係を示します。

表 18.1 *FILENAME ステートメントのデバイス情報*

デバイスまたはアクセス方法	機能	外部ファイル
ACTIVEMQ	SAS プログラムで、HTTP プロトコルによる ActiveMQ メッセージブローカとのメッセージの送受信を可能にします。	ActiveMQ RESTful API を使用した URL を介してアクセスされます。ActiveMQ MessageServlet が、HTTP 要求と ActiveMQ メッセージディスパッチャ間を統合しています。詳細については、*Application Messaging with SAS* を参照してください。

デバイスまたはアクセス方法	機能	外部ファイル
CATALOG	SAS カタログを外部ファイルとして参照します。	有効な 2、3 または 4 パートの SAS カタログ名です。(必要に応じて)後ろにカタログオプションが付きます。詳細については、*SAS 言語リファレンス: 解説編*を参照してください。
DATAURL	DATAURL アクセス方法を使用して、ユーザー指定のテキストからデータを読み込みできます。	ネットワークの場所ではなく、データ URL の指定から直接アクセスされます。詳細については、"FILENAME Statement, DATAURL Access Method" (*SAS Statements: Reference*)を参照してください。
DISK	ファイル参照名と DISK ファイルを関連付けます。	単独ファイルのパス名です。ファイル名を連結する場合は、カンマまたはスペースで区切られカッコで囲まれたパス名のリストです。仕様のレベルは、ユーザーのファイルシステム内の場所によって異なります。表 2.3 (52 ページ)UNIX パス名を指定するときに使用できる文字の置換を表示します。
DUMMY	ファイル参照名をヌルデバイスと関連付けます。	なし。DUMMY を使用すると、デバイスでの読み込み/書き込みを行うことなくアプリケーションをデバッグできます。このデバイスへの出力は破棄されます。
EMAIL*	アドレスあてに電子メールを送信します。	アドレスおよび電子メールオプションです。詳細については、"FILENAME ステートメント(EMAIL)を使用し、電子メールを送信する" (82 ページ)を参照してください。
FTP*	FTP サーバーを実行しているネットワーク上のコンピュータのファイルに対して、読み込みまたは書き込みを行います。	リモートコンピュータ上の外部ファイルのパス名です。後ろに FTP オプションが付きます。詳細については、*SAS 言語リファレンス: 解説編* および "他のシステムのファイルへのファイル参照名の割り当て(FTP、SFTP、SOCKET アクセスタイプ)" (75 ページ)を参照してください。 ファイルを z/OS 動作環境から UNIX に転送し、そのファイルのアクセスに S370V 形式または S370VB 形式のいずれかを使用する必要がある場合には、そのファイルを転送する前に、ファイルを RECFM=U および BLKSIZE=32760 にしておく必要があります。z/OS コンピュータに対して FTP 送信を行う場合は、一度で書き込むことができる z/OS PDS のメンバは 1 件のみです。複数のメンバに対して同時に書き込む必要がある場合は z/OS PDSE または UNIX システムサービスディレクトリを使用する必要があります。
Hadoop*	構成ファイルに場所が指定されている Hadoop 分散ファイルシステム(HDFS)上のファイルにアクセスできます。	HDFS にある外部ファイルのパス名を指定し、Hadoop オプションはその後に続けます。詳細については、"FILENAME Statement, Hadoop Access Method" (*SAS Statements: Reference*)を参照してください。

デバイスまたはアクセス方法	機能	外部ファイル
JMS	SAS プログラムで、JMS API 準拠メッセージサービスとのメッセージの送受信を可能にします。	JMS API 仕様を実装している、サードパーティのメッセージ指向ミドルウェアベンダによる JMS プロバイダを介してアクセスされます。classpath に指定が必要です。詳細については、*Application Messaging with SAS* を参照してください。
PIPE	UNIX コマンドの入力を読み込んだり、それに対して出力を書き込んだりします。	UNIX コマンドです。詳細については、4 章, “出力の印刷と出力先指定” (89 ページ)を参照してください。
PLOTTER	出力をプロッタに送信します。	デバイス名およびプロッタオプションです。詳細については、“ウィンドウのコンテンツの印刷” (97 ページ) および “UNIX 環境で PRINTTO プロシジャを使用する” (98 ページ)を参照してください。
PRINTER	出力をプリンタに送信します。	デバイス名およびプリンタオプションです。詳細については、“ウィンドウのコンテンツの印刷” (97 ページ)および“UNIX 環境で PRINTTO プロシジャを使用する” (98 ページ)を参照してください。
SFTP	SSHD サーバーを実行しているネットワーク上で接続が可能なホストコンピュータのファイルに対して、読み込みまたは書き込みを行います。	リモートコンピュータ上の外部ファイルのパス名で、後ろに SFTP オプションが付きます。詳細については、“FILENAME Statement, SFTP Access Method” (*SAS Statements: Reference*)、および“他のシステムのファイルへのファイル参照名の割り当て(FTP、SFTP、SOCKET アクセスタイプ)” (75 ページ)を参照してください。
SOCKET*	TCP/IP ソケットによって情報の読み込みと書き込みを行います。	SAS アプリケーションがサーバーアプリケーションかクライアントアプリケーションかに依存します。クライアントアプリケーションでは、*external-file* はホストの名前または IP アドレスと接続先の TCP/IP ポート番号です。後ろに TCP/IP オプションが付きます。クライアントアプリケーションでは、*external-file* はホストの名前または IP アドレスと接続先の TCP/IP ポート番号です。後ろに TCP/IP オプションが付きます。詳細については、*SAS ステートメント: リファレンス*を参照してください。
TEMP	ファイル参照名を、Work ライブラリに保存されている外部ファイルと関連付けます。	なし
TERMINAL	ファイル参照名をターミナルと関連付けます。	ターミナルのパス名です。

デバイスまたはアクセス方法	機能	外部ファイル
UPRINTER	**Printer Setup** ダイアログボックスを使ってセットアップされたデフォルトのプリンタに、出力を送信します。	なし

URL*	ファイルの URL を使用してファイルにリモートアクセスできるようにします。	URL サーバー上で読み込みまたは書き込みを行いたいファイルの名前です。URL は次のいずれかの形になっている必要があります。
		`http://hostname/file`
		`http://hostname:portno/file`

WebDAV	WebDAV(Web Distributed Authoring and Versioning)を使用して WebDAV サーバーが稼動しているネットワーク上で接続が可能なホストマシンのファイルに対して、読み込みまたは書き込みが行えるようにします。	WebDAV サーバー上で読み込みまたは書き込み対象ファイルの名前です。外部ファイルは次のいずれかの形になっている必要があります。
		`http://hostname/path-to-the-file`
		`https://hostname/path-to-the-file`
		`http://hostname:port/path-to-the-file`
		`https://hostname:port/path-to-the-file`

ZIP	ZIP ファイルにアクセスできます。	zlib をサポートする UNIX マシンとの読み取り/書き込みが可能な ZIP zlib サービスを使用する ZIP ファイルのパス名を指定します。詳細については、“FILENAME Statement, ZIP Access Method” (*SAS Statements: Reference*)を参照してください。

* SAS がロックダウン状態の場合、アクセス方法は、システム管理者が再有効化しない限りアクセス不能(無効)です。詳細については、“SAS Processing Restrictions for Servers in a Locked-Down State” (*SAS Language Reference: Concepts* 2 章)を参照してください。

関連項目:

- 4 章, “出力の印刷と出力先指定” (89 ページ)
- “外部ファイルとデバイスの使用” (68 ページ)

FOOTNOTE ステートメント: UNIX

プロシジャまたは DATA ステップ出力の下部に、最大で 10 行のテキストを書き込みます。

該当要素: 任意の場所

UNIX 固有: フットノートの最大長

参照項目: “FOOTNOTE Statement” (*SAS Statements: Reference*)

構文

FOOTNOTE <*n*> <'*text*' | "*text*">;

詳細

フットノートの最大長は 255 文字です。指定されたフットノートの長さが LINESIZE オプションの値よりも大きい場合、フットノートは行のサイズに収まるように切り捨てられます。

%INCLUDE ステートメント: UNIX

SAS プログラムステートメント、データ行またはその両方を SAS プログラムに取り込みます。

該当要素: 任意の場所

UNIX 固有: *source*(ファイル仕様が使用される場合)。次の値の有効な値: *encoding-value*

参照項目: "%INCLUDE Statement" (*SAS Statements: Reference*)

構文

%INCLUDE *source-1* <... *source-n*> </<SOURCE2> <S2=*length*> <ENCODING='*encoding-value*'> <*host-options*>>;

必須引数

source
%INCLUDE ステートメントを使用してアクセスしたい場所を説明します。ファイル仕様、内部行またはキーボードエントリの 3 つのソースが可能です。ファイル仕様は、"UNIX 環境で外部ファイルまたはデバイスにアクセスする" (69 ページ)で説明されているファイル仕様形式のいずれにもできます。

注 集計構文を使用するとき、メンバ名に先頭の桁が含まれる場合は、そのメンバ名を引用符で囲みます。メンバ名にマクロ変数参照が含まれる場合は、二重引用符を使用します。

オプション引数

ENCODING='*encoding-value*'
指定されたソースから読み込むときに使用するエンコーディングを指定します。ENCODING=の値は、指定されたソースのエンコーディングが、現在のセッションのエンコーディングとは異なることを示します。

指定されたソースのデータを読み込むときは、指定されたエンコーディングからセッションのエンコーディングへ、SAS によってデータがトランスコードされます。

有効なエンコーディング値については、"Overview to SAS Language Elements That Use Encoding Values" (*SAS National Language Support (NLS): Reference Guide* 22 章)を参照してください。

host-options
UNIX 版で有効なステートメントオプションで構成されます。次のオプションを使用できます。

BLKSIZE=*block-size*
BLK=*block-size*
1 回の I/O 操作で物理的に書き込まれたり読み込まれたりするバイト数を指定します。デフォルト値は 8K です。最大で 1KB です。

LRECL=*record-length*
論理レコード長(バイト単位)を指定します。SAS 9.4 では、LRECL=のデフォルト値は 32,767 です。固定長レコード(RECFM=F)を使用する場合、LRECL=のデフォルト値は 256 になります。*record-length* 値の可能な範囲は、1 から 1,048,576 (1 MB)です。

RECFM=*record-format*
レコード形式を指定します。UNIX 版では次の値が使用できます。

D	デフォルトのレコード形式(可変長形式)。
F	固定長形式。つまり、各レコードの長さは同じです。
N	バイナリ形式。ファイルはバイトストリームで構成され、レコード境界はありません。
P	プリント形式。
V	可変長形式。各レコードは改行文字で終わります。
S370V	可変長 S370 レコード形式(V)。
S370VB	可変長ブロック S370 レコード形式(VB)。
S370VBS	スパンレコード可変長ブロック S370 レコード形式(VBS)。

S370 値は、z/OS ファイルとして配置されたファイルの場合のみ有効です。つまり、バイナリで、variable-length レコードを持ち、EBCDIC 形式であるファイルです。固定長形式の z/OS ファイルを使用する場合は、まずそれを variable-length のバイナリの z/OS ファイルにコピーします。

詳細

%INCLUDE ステートメントでオプションを指定する場合は、オプションリストの前にスラッシュ(/)を付けることを忘れないでください。

関連項目:

“UNIX 環境の外部ファイルとデバイスについて” (68 ページ)

INFILE ステートメント: UNIX

INPUT ステートメントを使用して読み込む外部ファイルを特定します。

該当要素: DATA ステップ
UNIX 固有: 次の項目の有効な値: *encoding-value*、*file-specification* および *host-options*
参照項目: “INFILE Statement” (*SAS Statements: Reference*)

構文

INFILE *file-specification* <ENCODING='*encoding-value*'> <*option(s)*> <*host-option(s)*>;

必須引数

file-specification
“UNIX 環境で外部ファイルまたはデバイスにアクセスする” (69 ページ)で説明されているファイル仕様形のうち、どれでも使用できます。

オプション引数

ENCODING='*encoding-value*'
外部ファイルからの読み取り時に使用するエンコーディングを指定します。ENCODING=の値は、外部ファイルのエンコーディングが、現在のセッションのエンコーディングとは異なることを示します。

外部ファイルのデータを読み取るときは、指定されたエンコーディングからセッションのエンコーディングへ、SAS によってデータがトランスコードされます。

有効なエンコーディング値については、“Overview to SAS Language Elements That Use Encoding Values” (*SAS National Language Support (NLS): Reference Guide* 22 章)を参照してください。

host-options
UNIX 環境に固有です。これらのオプションでは次のいずれもが使用できます。

BLKSIZE=
BLK=
1 回の I/O 操作で物理的に読み込まれるバイト数を指定します。デフォルト値は 8K です。最大値は 1G–1 です。

TERMSTR=
UNIX でフォーマットされたファイルでの行末またはレコード区切り文字を制御します。TERMSTR=オプションで有効な値は次のとおりです。

CR キャリッジリターン。

CRLF キャリッジリターン、ラインフィード。

LF ラインフィード。このパラメータは、UNIX でフォーマットされたファイルの読み込みに使用されます。LF がデフォルトです。

LRECL=
論理レコード長を指定します。この値は、有効なレコード形式によって異なります(RECFM)。SAS 9.4 では、LRECL=のデフォルト値は 32,767 です。固定長レコード(RECFM=F)を使用する場合、LRECL=のデフォルト値は 256 になります。最大長は 1G です。

- RECFM=F の場合、LRECL=オプションの値により、1 レコードとして読み取られるバイト数が決定されます。

  注: RECFM=F の場合、SAS 9.4 が SAS の以前のバージョンと通信するには、LRECL=は 256 に設定する必要があります。

- RECFM=N の場合、LRECL=オプションの値は 256 以上にする必要があります。
- RECFM=V の場合、LRECL=オプションの値により最大レコード長が決定されます。指定された長さを超えるレコードは切り捨てられます。

RECFM=
レコード形式を指定します。UNIX 版では次の値が使用できます。

D デフォルトのレコード形式(可変長形式)。

F 固定長形式。つまり、各レコードの長さは同じです。

N	バイナリ形式。ファイルはバイトストリームで構成され、レコード境界はありません。LRECL オプションを指定しない場合、デフォルトではファイルから 256 バイトが読み込まれます。
P	プリント形式。
V	可変長形式。各レコードは改行文字で終わります。
S370V	可変長 S370 レコード形式(V)。
S370VB	可変長ブロック S370 レコード形式(VB)。
S370VBS	スパンレコード可変長ブロック S370 レコード形式(VBS)。

詳細

ENCODING=オプションは、予約済みのファイル参照名ではないファイル仕様が INFILE ステートメントに含まれている場合のみ有効です。INFILE ステートメントに、ENCODING=引数および予約済みのファイル参照名 DATALINES または DATALINES4 が *file-specification* として含まれている場合は、エラーメッセージが表示されます。INFILE ステートメントの ENCODING=値は、ENCODING=システムオプションの値をオーバーライドします。

関連項目:

“外部ファイルとデバイスの使用” (68 ページ)

LENGTH ステートメント: UNIX

変数の保存用のバイト数を指定します。

該当要素: DATA ステップ

UNIX 固有: 有効な数値変数の長さ

参照項目: “LENGTH Statement” (*SAS Statements: Reference*)

構文

LENGTH <*variable-1*> <*...variable-n*> <$> *length* <DEFAULT=*n*>;

必須引数

length
UNIX 版の数値変数の範囲は 3 - 8 です。数値変数に指定できる長さの最小値は、ユーザーのシステムが使用する浮動小数点形式によって異なります。ほとんどのシステムでは IEEE 浮動小数点形式が使用され、最小値は 3 バイトです。

DEFAULT=*n*
新規に作成された数値変数の値を保存するのに使用されるデフォルトのバイト数を、8 バイトから *n* の値に変更します。UNIX では、*n* の範囲は 3 - 8 です。

関連項目:

“データ表現” (223 ページ)

LIBNAME ステートメント: UNIX

SAS ライブラリとライブラリ参照名(ショートカット名)を関連付けたり、関連付けを解除したりします。1 つまたはすべてのライブラリ参照名をクリアします。SAS ライブラリの特徴をリストします。SAS ライブラリを連結します。SAS カタログを暗示的に連結します。ファイルロックを無効化します。

該当要素: 任意の場所

UNIX 固有: *engine*、*library* および *engine/host-options*

参照項目: “LIBNAME Statement” (*SAS Statements: Reference*)

構文

LIBNAME *libref* <*engine*> '*SAS-library*' <*options*> <*engine/host-options*>;

LIBNAME *libref* <*engine*> ('*library-1*' <, ... '*library-n*'>) <*options*>;

LIBNAME *libref* ('*library-1*' | *libref-1*, ..., '*library-n*' | *libref-n*);

LIBNAME *libref* CLEAR | _ALL _ CLEAR;

LIBNAME *libref* LIST | _ALL _ LIST;

必須引数

libref
有効なライブラリ参照名(“LIBNAME Statement” (*SAS Statements: Reference*)を参照)。SAS では、特殊なシステムライブラリ用に一部のライブラリ参照名が予約されます。予約済み一部のライブラリ参照名の詳細については、“UNIX 環境で SAS によって割り当てられるライブラリ参照名” (55 ページ)を参照してください。

オプション引数

engine
UNIX 版でサポートされるライブラリエンジンのうちの 1 つです。エンジンに関する説明は、“詳細” (345 ページ)を参照してください。エンジン名が指定されない場合は、使用するエンジンが自動的に決定されます(“LIBNAME ステートメントからのエンジン名の省略” (347 ページ)を参照)。

'SAS-library'
指定するエンジンおよび現在使用している作業ディレクトリに依存します。表 18.2 (346 ページ) では、各エンジンがこの引数に対して期待していることを説明しています。ディレクトリのパス名を指定します(“UNIX 環境でのパス名の指定” (51 ページ)を参照)。LIBNAME ステートメントを使用してディレクトリを作成することはできません。指定するディレクトリはすでに存在している必要があり、ユーザーがそのディレクトリへのアクセス権を持っている必要があります。ライブラリ名を引用符で囲みます。UNIX のパス名は大小文字が区別されるので注意してください。

'library-n' | libref-n
ユーザーが 1 つのライブラリ参照名でアクセスしようとしているライブラリの(割り当て済みの)パス名またはライブラリ参照名です。ライブラリを連結したいときは、LIBNAME ステートメントのこれらの形を使用します。カンマまたは空白スペースを使ってパス名を区切ります。ライブラリのパス名を引用符で囲みます。ライブラリ参照名は引用符で囲みません。ライブラリ連結の詳細については、“複数のディレクトリへの 1 つのライブラリ参照名の割り当て(ディレクトリの連結)” (52 ページ)を参照してください。

options
すべての動作環境で使用可能な LIBNAME ステートメントオプションです。これらのオプションについては、“LIBNAME Statement” (*SAS Statements: Reference*)を参照してください。

engine/host-options
“エンジン/ホストオプション” (348 ページ)で説明されているオプションのうちいずれかが使用できます。

ALL
現在定義されているすべてのライブラリ参照名を参照します。キーワードは、ライブラリ参照名をリストまたはクリアするときに使用します。

CLEAR
指定されたライブラリ参照名をクリアします。_ALL_を指定した場合は、現在定義されているライブラリ参照名をすべてクリアします。Sasuser、Sashelp および Work が割り当てられたままになります。

注: 環境変数によって定義されているライブラリ参照名をクリアすると、変数は、定義されたままになりますがライブラリ参照名としては認識されません。その変数を、ライブラリ参照名またはファイル参照名としてくりかえし使用できます。詳細については、“UNIX 環境におけるライブラリ参照名としての環境変数の使用” (54 ページ)を参照してください。

ライブラリ参照名とそれに対応するライブラリとの関連付けは、ジョブまたはセッションの終了時に自動的にクリアされます。現在のセッション中に既存のライブラリ参照名を別の SAS ライブラリと関連付ける場合、セッションを終了したりライブラリ参照名をクリアしたりする必要はありません。新規の SAS ライブラリの LIBNAME ステートメントを発行するときに、ライブラリ参照名が自動的に再割り当てされます。

LIST
指定されたライブラリ参照名に関連付けられているエンジン、パス名、ファイル形式、アクセス権などを、SAS ログに書き込みます。_ALL_を指定した場合は、現在定義されているすべてのライブラリ参照名のそれらの情報を印刷します。環境変数として定義されているライブラリ参照名は、それらがすでに SAS ステートメントで使用されている場合のみ、表示されます。

NOSETPERM
ライブラリメンバが同一のライブラリ参照名で開かれるとき、あるライブラリメンバから別のライブラリメンバへ権限の設定が引き継がれないように指定します。パスに対して 2 つの割り当て(NOSETPERM オプションのあるものとないもの)がある場合、それらの割り当てはパスが一致しないものとして処理されます。
NOSETPERM オプションがある LIBNAME ステートメントでは、権限の設定は引き継がれません。

NOSETPERM オプションを使用してライブラリ参照名の権限の設定を無効化すると、そのオプションは、ユーザーがライブラリ参照名を使用するとき常に有効になります。NOSETPERM オプションを無効化するオプションはありません。
NOSETPERM オプションを無効化するには、次のステートメントを送信してください。

```
libname libref clear;
```

詳細

エンジンの種類
エンジンは、主に 2 つの種類があります。

ビューエンジン
SAS/ACCESS ソフトウェア、SQL プロシジャおよび DATA ステップビューによって記述される SAS ビューを、SAS で読み込めるようにします。SAS ビューの名前はデータセットのディスクリプタ部分の一部として保存されるため、SAS ビューエンジンの使用は自動です。

ライブラリエンジン
SAS ライブラリレベルでアクセスを制御します。すべての SAS ライブラリはライブラリエンジンが関連付けられており、そのライブラリ内のファイルは、そのエンジンを介してのみアクセスできます。ライブラリエンジンには 2 つの種類があります。

ネイティブエンジン
SAS によって作成され保持される SAS ファイルにアクセスします。これらのエンジンに関する説明は、次の表を参照してください。

インタフェースエンジン
他のベンダのファイルを SAS ファイルと同様に処理します。詳細については、次の表と"UNIX 環境で BMDP、OSIRIS、SPSS ファイルにアクセスする" (61 ページ)を参照してください。

表 18.2 *エンジン名と説明*

エンジンの種類	名前(エイリアス)	説明	SAS ライブラリ
デフォルト	V9 (BASE) V8	SAS データファイルの新規作成と、バージョン 8 または SAS 9 で作成された既存の SAS データファイルへのアクセスを可能にします。V8 と V9 のエンジンは同一です。このエンジンは、SAS の以前のリリースで作成されたデータファイルへの読み込みアクセスを可能にします。ただしこのエンジンは、SAS 9 のカタログをサポートしている唯一のエンジンです。このエンジンはデータセットのインデックス作成と圧縮を可能にします(*SAS 言語リファレンス: 解説編*にも説明があります)。	ライブラリが含まれているディレクトリのパス名です。
互換性	V6	6.09 - 6.12 のリリースで作成されたデータファイルにアクセスします。このエンジンは読み取り専用です。	ライブラリが含まれているディレクトリのパス名です。
サーバー	SPD サーバー	クライアントセッションとデータサーバーとの通信を可能にします。ご使用のクライアントコンピュータでこのエンジンを使用するには、SAS Scalable Performance Data Server のライセンス契約が必要です。詳細については、*SAS Scalable Performance Data Server ユーザーガイド*を参照してください。	サーバー上の SAS Scalable Performance Data Server (SPD Server)のライブラリの論理的な LIBNAME ドメイン名です。ネームサーバーは、そのドメイン名をライブラリの物理的なパスに分解します。

エンジンの種類	名前(エイリアス)	説明	SAS ライブラリ
	MDDB	クライアントセッションと MDDB サーバーとの通信を可能にします。ご使用のサーバーでこのエンジンを使用するには、SAS/MDDB Server のライセンス契約が必要です。	
transport	XPORT	移送データセットにアクセスします。このエンジンは、リリース 6.06 以降の SAS を実行しているすべてのホストのもとで使用できる、コンピュータに依存しない SAS 移送ファイルを作成します。このエンジンについては、*SAS ファイルの移動とアクセス* で説明されています。	シーケンシャルデバイスまたはディスクファイルのパス名です。
XML	XML	アプリケーションおよびコンピュータに依存しないファイルである XML ドキュメントを、生成(書き込み)したり処理(読み込み)したりします。	XML ドキュメントのパス名です。
インターフェイス	BMDP	BMDP ファイルへの読み込み専用アクセスを提供します。このエンジンは、AIX、HP-UX および Solaris でのみ使用できます。	データファイルのパス名です。
	OSIRIS	OSIRIS ファイルへの読み込み専用アクセスを提供します。	データファイルのパス名です。
	SPSS	SPSS ファイルへの読み込み専用アクセスを提供します。	データファイルのパス名です。

LIBNAME ステートメントからのエンジン名の省略

SAS に適正なエンジンを判断させるよりも、エンジン名を指定するほうが通常はより効率的です。ただし、ユーザーが LIBNAME ステートメントでエンジン名を省略する場合、または環境変数をライブラリ参照名として定義する場合には、SAS が正しいエンジンを判断します。

ENGINE=システムオプションが指定してある場合、SAS はユーザーが指定したエンジン名を使用します。ENGINE=システムオプションの説明については、“ENGINE=システムオプション: UNIX” (376 ページ)を参照してください。

注: ENGINE=システムオプションは、ディスク上にあるライブラリのデフォルトエンジンのみを指定します。

ENGINE=システムオプションを指定しなかった場合、SAS は所定のディレクトリにあるファイルの拡張子を判断し、次のルールを用いてエンジンを決定します。

- ライブラリ内のすべての SAS データセットが同一のエンジンによって作成された場合、ライブラリ参照名はそのエンジンを使用して割り当てられます。

  注: データセットの作成に使用されたエンジンがデフォルトのエンジンとは異なる場合、ビューまたはストアドプログラムを作成することはできません。詳細については、“UNIX 環境における 1 つのライブラリへの複数エンジンの使用” (54 ページ)を参照してください。

- 所定のディレクトリに SAS データセットが存在しない場合、ライブラリ参照名はデフォルトのエンジンを使用して割り当てられます。
- 複数のエンジンの SAS データセットが存在する場合は、混合した種類のエンジンが検出されたことを通知するメッセージが発行され、ライブラリ参照はデフォルトのエンジンで割り当てられます。

エンジン/ホストオプション

LIBNAME ステートメントでは次のオプションを使用できます。

ENABLEDIRECTIO
LIBNAME ステートメントで特定されたライブラリ内で開かれるすべてのファイルに対して、ダイレクト I/O を使用可能にする指定です。

注: ENABLEDIRECTIO は、**Work** ディレクトリでは使用できません。

あるディレクトリに ENABLEDIRECTIO オプション付きで割り当てられたライブラリ参照名は、同じディレクトリに ENABLEDIRECTIO オプションなしで割り当てられたライブラリ参照名とは一致しません。2 つのライブラリ参照名は同一のディレクトリを指定しますが、前者のライブラリ参照名を使用して開かれたファイルの読み込みと書き込みには、ダイレクト I/O が使用されます。別のライブラリ参照名を使用して開かれたファイルの読み込みと書き込みには、通常のディスク I/O 呼び出しが使用されます。

ライブラリ参照名が LIBNAME ステートメントにリストされているファイル(複数ある場合も含む)のダイレクト I/O を有効化するには、ENABLEDIRECTIO オプションと USEDIRECTIO=オプションを併用する必要があります。次の例では、ENABLEDIRECTIO オプションと USEDIRECTIO= LIBNAME オプションが併用されています。この例では、ライブラリ参照名テストで参照されるすべてのファイルがダイレクト I/O 用に開きます。

```
LIBNAME test '.' ENABLEDIRECTIO USEDIRECTIO=yes;
```

ヒント 次の例では、ENABLEDIRECTIO LIBNAME オプションを使用し、ライブラリ参照名 test に関連付けられたファイルをダイレクト I/O のために開けるようにします。USEDIRECTIO=データセットオプションは、ダイレクト I/O で test.file1 を開きます。test.file2 はダイレクト I/O が可能であっても、ダイレクト I/O では開かれません。

```
LIBNAME test'.'ENABLEDIRECTIO;
data test.file1(USEDIRECTIO=yes);
... more SAS statements ...
run;
data test.file2;
... more SAS statements ...
run;
```

FILELOCKS=NONE | FAIL | CONTINUE
LIBNAME ステートメントのライブラリ参照名の下で開いているファイルについて、ファイルのロックを有効化するか無効化するかを指定します。FILELOCKS ステートメントオプションは、ライブラリ参照名に関連付けられたファイルに対してのみ適用されるという点を除けば、FILELOCKS システムオプションと同様に動作します。FILELOCKS ステートメントオプションでは次の値が使用できます。

NONE
ファイルのロックをオフにします。NONE は、SAS がファイルでの既存のロックを確認せずにファイルを開く指定です。NONE では、ファイルにオペレーティングシステムロックがかかりません。このようなファイルは、共有更新アクセスから保護されません。

FAIL
ファイルのロックをオンにします。FAIL は、SAS がファイルにオペレーティングシステムロックをかけようとする指定です。ファイルへのアクセスは、ファイルがすでにロックされている場合、またはファイルをロックできない場合に拒否されます。

CONTINUE
ファイルのロックをオンにします。CONTINUE は、SAS がファイルにオペレーティングシステムロックをかけようとする指定です。ファイルがすでに他のユーザーにロックされている場合は、ファイルを開くことができません。別の何らかの理由によってファイルをロックできない場合(ファイルシステムがロック機能をサポートしていない場合など)、ファイルは開かれ、警告メッセージがログに送られます。

LIBNAME ステートメントの FILELOCKS オプションは、LIBNAME ステートメント内でリストされているライブラリ参照名の下で開いているほとんどの(一部は除く)SAS の I/O ファイル(データセット、カタログなど)に適用されます。

FILELOCKS システムオプションを使用するばあい、RESET は有効な値になりますが、FILELOCKS ステートメントオプションの場合、RESET は有効な値にはなりません。

FILELOCKS ステートメントオプションのかわりに FILELOCKS システムオプションを使用して、ユーザーのファイルにロック動作を設定します。(FILELOCKS ステートメントオプションは、SAS の今後のリリースでは重要度が低くなります。) LIBNAME ステートメントの FILELOCK オプションは LIBNAME システムオプションをオーバーライドします。詳細については、UNIX 動作環境の FILELOCKS システムオプションを参照してください。 Scalable Performance Data Server でサポートされているオプションを指定することもできます。詳細については、次のサイトにある *Scalable Performance Data Server: User's Guide* を参照してください。**テクニカルサポート Web サイト**を参照してください。

FILELOCKWAIT=*n*
ロックされたファイルが別のプロセスで使用可能になるまで SAS が待機する秒数を設定します。

ロックされたファイルが *n* で指定した秒数が経過する前にロック解除された場合、そのファイルは現在のプロセスのためにロックされ、処理が続行します。指定した秒数に達してもファイルがロックされたままの場合、SAS はロックされたファイルのエラーをログに書き込み、DATA ステップは失敗します。

Interaction FILELOCKWAIT=オプションを指定すると、1 つ以上の SAS/SHARE Server と、別のプロセスにロックされた SAS ファイルのロック解除を待機しているクライアントセッションに対して、悪影響が生じるおそれがあります。1 つ以上の待機条件により、SAS/SHARE サーバーおよびクライアントプロセスの失敗を招くおそれがあります。

Interaction SAS/SHARE プロセスが失敗する可能性を防ぐためには、FILELOCKWAIT=0 を設定します。これにより、SAS/SHARE Server およびクライアントが、ロックされたファイルのロック解除を待機する時間がキャンセルされます。待機時間のキャンセルが、プロセス失敗を防ぐ場合があります。詳細については、FILELOCKWAITMAX システムオプションを参照してください。また、*SAS/SHARE: User's Guide* の LIBNAME ステートメントの使用によるサーバーライブラリの事前定義に関するセクションで、FILELOCKWAITMAX システムオプションを参照してください。

Range 0–600

Default 0

TRANSFERSIZE=*n*K | *n*M
開いているファイルから読み込まれるデータの大きなブロックのサイズを指定します。

n
整数を指定します。

K
ブロックのサイズをキロバイト単位で指定します。

M
ブロックのサイズをメガバイト単位で指定します。

TRANSFERSIZE オプションを使用するには、ファイルをダイレクト I/O 用に開く必要があります。つまり、ENABLEDIRECTIO オプションと USEDIRECTIO=オプションを両方とも有効にする必要があります。NABLEDIRECTIO オプションおよび USEDIRECTIO=オプションなしで TRANSFERSIZE オプションを使うと、そのオプションは受け付けられますが機能しません。

次の例では、test.file1 がダイレクト I/O 用に開かれるため、128k ブロックのデータがこのファイルから読み込まれます。est.file2 はダイレクト I/O 用に開かれず、TRANSFERSIZE オプションはこのファイルには影響しません。

```
LIBNAME test'.'ENABLEDIRECTIO TRANSFERSIZE=128k;
data test.file1(USEDIRECTIO=yes);
... more SAS statements ...
run;
data test.file2;
... more SAS statements ...
run;
```

次の例では、DATA ステートメントにリストされているすべてのファイルが 128k ブロックのデータを読み込みます。これは、すべてのファイルが ENABLEDIRECTIO、USEDIRECTIO= および TRANSFERSIZE の各オプションの影響を受けるからです。

```
LIBNAME test'.'ENABLEDIRECTIO USEDIRECTIO=yes TRANSFERSIZE=128k;
data test.file1;
... more SAS statements ...
run;
data test.file2;
... more SAS statements ...
run;
data test.file3;
... more SAS statements ...
run;
```

USEDIRECTIO= YES | NO
ENABLEDIRECTIO ステートメントオプションと併用することにより、LIBNAME ステートメントにリストされているライブラリ参照名に関連付けられたすべてのファイルに対してダイレクトファイル I/O を有効化または無効化にします。USEDIRECTIO を使用する場合は、Work ライブラリではなく、永久ライブラリを使用する必要があります。

注: USEDIRECTIO は、**Work** ディレクトリでは使用できません。

詳細については、“エンジン/ホストオプション” (348 ページ)を参照してください。

要件
USEDIRECTIO=を ENABLEDIRECTIO ステートメントオプションと併用して、ダイレクトファイル I/O を有効化します。

関連項目:

システムオプション:

- “FILELOCKS システムオプション: UNIX” (377 ページ)

その他の参照資料:

- “UNIX 環境の SAS ファイル、ライブラリ、エンジンについて” (35 ページ)

SYSTASK ステートメント: UNIX

非同期タスクを実行します。

該当要素: 任意の場所

UNIX 固有: すべて

構文

SYSTASK COMMAND "*operating-environment-command*" <WAIT | NOWAIT> <TASKNAME=*taskname*> <MNAME=*name-variable*> <STATUS=*status-variable*> <SHELL<="*shell-command*"> > <CLEANUP>;

SYSTASK LIST <_ALL_ | *taskname*> <STATE> <STATVAR> ;

SYSTASK KILL *taskname* <*taskname*...>;

必須引数

コマンド
operating–environment-command を実行します。

LIST
特定のアクティブなタスク、またはシステム内のすべてのアクティブなタスクをリストします。タスクは、それが実行中の場合、またはそれが完了しており、WAITFOR ステートメントを使用してそれが待機されていない場合は *アクティブ*です。

KILL
指定されたタスクを強制的に終了します。

operating-environment-command
UNIX コマンド(コマンド固有のオプションを含む)の名前、または X Window System アプリケーションの名前を指定します。一重または二重の引用符でコマンドを囲みます。コマンド固有のオプションに二重引用符が必要な場合は、各オプションでその記号をくりかえします。たとえば、次のようになります。

```
SYSTASK COMMAND "xdialog -m ""There was an error."" -t ""Error"" -o";
```

注: コマンド名がシェルのエイリアスである場合、またはコマンドのパス名にシェルの特殊記号チルダ(~)およびアスタリスク(*)を使用する場合は、シェルがエイリアスや特殊記号を処理できるように SHELL オプションを指定する必要があります。

```
SYSTASK COMMAND "mv ~usr/file.txt /tmp/file.txt" shell;
```

次の例では、SHELL オプション、~usr パスが実行時に拡張されますが、直接には実行されません。

注: ユーザーが指定した *operating-environment-command* は、キーボードからの入力を必須にできません。

ヒント SHELL オプションが使用される場合であってもエラー内でシェルのエイリアスの結果を使用する場合は、そのシェルはシェル初期化ファイルを処理しません。SHELL エイリアスのかわりに実際の SHELL コマンドを使用します。

オプション引数

WAIT | NOWAIT
SYSTASK COMMAND コマンドが現在の SAS セッションの実行をタスクが完了するまで中断するかどうかを決定します。NOWAIT は初期値です。NOWAIT オプションで始まるタスクの場合、SAS タスクが完了するまでセッションの実行を中断する必要があるときは、WAITFOR ステートメントを使用できます。詳細については、“WAITFOR ステートメント: UNIX” (355 ページ)を参照してください。

TASKNAME=*taskname*
タスクを特定する名前を指定します。タスク名は、すべてのアクティブなタスクの中で一意である必要があります。タスクは、それが実行中の場合、またはそれが完了しており、WAITFOR ステートメントを使用してそれが待機されていない場合はアクティブです。タスク名が重複すると、SAS ログ内でエラーが発生します。タスク名を指定しない場合は、SYSTASK が名前を自動生成します。タスク名に空白文字が含まれる場合は、それを引用符で囲みます。

タスクが完了しても、そのタスクに WAITFOR ステートメントを発行するか CLEANUP オプションを指定しない限り、タスク名をくりかえし使用することはできません。

MNAME=*name-variable*
タスクに対して自動的に生成されたタスク名を SYSTASK を使って保存するための、マクロ変数を指定します。TASKNAME オプションと MNAME オプションを両方指定する場合、SYSTASK は、ユーザーが TASKNAME で指定した名前を、MNAME で指定した変数にコピーします。

STATUS=*status-variable*
タスクのステータスを SYSTASK を使って保存するための、マクロ変数を指定します。ステータス変数は、すべてのアクティブなタスクの中で一意である必要があります。

SHELL<="*shell-command*"
operating-environment-command がオペレーティングシステムシェルコマンドを使って実行されるように指定します。このシェルは、*operating-environment-command* に含まれるシェル特殊文字を拡張します。*shell-command* を指定する場合、SYSTASK はユーザーがシェルの呼び出し用に指定したシェルコマンドを使用します。それ以外の場合は SYSTASK デフォルトのシェルを使用します。シェルコマンドを引用符で囲みます。

注: SHELL オプションは、ユーザーが指定した SHELL コマンドが-i オプションを使用してステートメントを通すことを前提としています。ユーザーのシェルコマンドは、通常 **sh**、**csh**、**ksh**、**bash** のいずれかです。

CLEANUP
タスクが完了したら LISTTASK 出力からタスクが削除されるよう指定します。タスクが削除されれば、WAITFOR ステートメントを発行しないでもタスク名を再使用できます。

SYSTASK コマンドを複数回使用する長期のジョブがある場合は、SYSTASK コマンドで WAITFOR ステートメントまたは CLEANUP オプションを使用してメモリをクリアします。WAITFOR ステートメントは、SYSTASK コマンドによって開始されたすべての完了プロセスの情報を削除することによってメモリを解放します。CLEANUP オプションは、特定のジョブが完了したときにメモリをクリアし、今後の使用に備えてメモリを開放します。ジョブが完了した後に WAITFOR ステートメントを使用する場合、そのジョブはすでに CLEANUP オプションによってクリーンアップされているので、このステートメントは無効になります。

詳細

SYSTASK ステートメントはユーザーの SAS セッションまたはアプリケーション内から、ホスト固有のコマンドを実行できるようにします。X ステートメントとは異なり、SYSTASK ステートメントはこれらのコマンドを非同期タスクとして実行します。つまり、これらのタスクは現在実行中の他のすべてのタスクから独立して実行します。非同期タスクはバックグラウンドで実行するため、ユーザーは非同期タスクがまだ実行中であっても追加のタスクを実行できます。

たとえば、新規のシェルを起動して、そのシェルで UNIX **cp** コマンドを実行する場合は、次のステートメントを使用できます。

```
systask command "cp /tmp/sas* ~/archive/" taskname="copyjob1"
status=copysts1 shell;
```

cp コマンドのリターンコードはマクロ変数 COPYSTS1 に保存されます。

コマンドの出力は SAS ログに表示されます。

UNIX と PC とでは構文が異なる場合があるので、PC の SAS ジョブを UNIX 上で実行できるように変換すると変換プロセスでエラーが起きることがあります。たとえば、

```
systask command "md directory-name" taskname="mytask";
```

のようなコマンドを入力するとエラーになります。**md** は PC で“ディレクトリを作成”するコマンドであり、UNIX では意味がないので、エラーになります。変換プロセスで **md** は **mkdir** になります。SYSTASK ステートメントで SHELL オプションを使用する必要があります。**mkdir** は UNIX シェルに組み込まれているものであり、PC の場合のような独立したコマンドではないからです。

ログにはエラーメッセージが書き込まれます。

注: SAS アプリケーションの SYSTASK ステートメントに続くプログラムステップは、通常は SYSTASK ステートメントの実行が成功するかどうかに依存します。したがって、SYSTASK ステートメントで構文エラーが発生すると、使用中の SAS アプリケーションは中止されます。

SAS から開始できる非同期プロセスは 2 つのタイプがあります。

Task
: SYSTASK コマンドで開始されるタスクはすべて Task タイプです。これらのタスクについて STATVAR または STATE を指定しない場合、SYSTASK LIST は、タスクの名前、タイプ、状態およびステータスのマクロ変数の名前を表示します。SYSTASK KILL を使用すれば、Task タイプのタスクのみを終了できます。

SAS/CONNECT プロセス
: SIGNON ステートメントまたはコマンドと RSUBMIT ステートメントを使用して SAS/CONNECT から開始されるタスクは、タイプが SAS/CONNECT プロセスです。SAS/CONNECT プロセスを表示するには、LISTTASK ステートメントを使用してタスク名、タイプおよび状態を表示します。SAS/CONNECT プロセスを終了するには、KILLTASK ステートメントを使用します。SAS/CONNECT プロセスについては、*SAS/CONNECT User's Guide* を参照してください。

注: タスク(SAS/CONNECT プロセスに限らない)を表示する望ましい方法は、SYSTASK LIST のかわりに LISTTASK ステートメントを使用することです。タスクを編集する望ましい方法は、SYSTASK KILL のかわりに KILLTASK ステートメントを使用することです。

SYSRC マクロ変数には、SYSTASK ステートメントのリターンコードが含まれています。STATUS オプションで指定したステータス変数には、SYSTASK COMMAND で開始されたプロセスのコードが含まれています。タスクが正しく実行されるようにするには、SYSTASK ステートメントのステータスと SYSTASK ステートメントによって開始されるプロセスのステータスの、両方を監視する必要があります。

SYSTASK ステートメントの実行が失敗する場合、SYSRC マクロ変数に非ゼロ値が含まれていることがあります。たとえば、タスクを完了するためのリソースが十分にないか、SYSTAS ステートメントに構文エラーが含まれている可能性があります。SYSTASK KILL ステートメントを使用する場合、1 つ以上のプロセスが終了できなくても、SYSRC が非ゼロ値に設定されます。

タスクが開始されると、そのステータス変数が NULL に設定されます。各タスクのステータス変数を使用して、どのタスクが完了できなかったかを判断できます。ステータス変数が NULL であるタスクは実行が完了していません。タスクが異常終了すると、そのタスクのステータス変数は -1 に設定されます。詳細については、“WAITFOR ステートメント: UNIX” (355 ページ)を参照してください。

X ステートメントとは異なり、SYSTASK ステートメントを使用して対話セッションを新規に開始することはできません。

関連項目:

ステートメント:

- “WAITFOR ステートメント: UNIX” (355 ページ)
- “X ステートメント: UNIX” (356 ページ)

その他の参照資料:

- “SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)

TITLE ステートメント: UNIX

SAS 出力のタイトル行を指定します。

該当要素: 任意の場所

UNIX 固有: タイトルの最大長

参照項目: “TITLE Statement” (*SAS Statements: Reference*)

構文

TITLE <*n*> <'*text*' | "*text*">;

詳細

対話モードの場合、タイトルの最大長は 254 文字です。その他の場合、最大長は 200 文字です。指定されたタイトルの長さが LINESIZE オプションの値よりも大きい場合、そのタイトルは行のサイズに収まるように切り捨てられます。

WAITFOR ステートメント: UNIX

タスクの実行が終了するまで、現在の SAS セッションの実行を中断します。

該当要素: 任意の場所

UNIX 固有: すべて

構文

WAITFOR <_ANY_ | _ALL_> *taskname* <*taskname*...> <TIMEOUT=*seconds*> ;

必須引数

taskname
ユーザーが待機するタスクの名前を指定します。タスク名については、“SYSTASK ステートメント: UNIX” (351 ページ)を参照してください。ユーザーが指定するタスク名は、SYSTASK COMMAND ステートメントを使用して割り当てられるタスク名と完全に一致している必要があります。ワイルドカードを使用してタスク名を指定することはできません。

オプション引数

ANY | _ALL_
現在の SAS セッションの実行を、指定された 1 つまたは全部のタスクの実行が終了するまで中断します。デフォルトの設定は _ANY_ です。これは、指定されたタスクのうちの 1 つが実行を完了すると同時に、WAITFOR ステートメントが実行を終了することを意味しています。

TIMEOUT=*seconds*
WAITFOR が現在の SAS セッションを中断する最大秒数を指定します。TIMEOUT オプションを指定しない場合、WAITFOR は SAS セッションの実行を無期限に中断します。

詳細

WAITFOR ステートメントは、指定されたタスクの実行が終了するか TIMEOUT=の間隔時間が経過するまで、現在の SAS セッションを中断します。指定されたタスクが WAIT オプションで開始された場合、WAITFOR ステートメントはそのタスクを無視します。WAIT オプションについては、“SYSTASK ステートメント: UNIX” (351 ページ)を参照してください。

たとえば次のステートメントは、3 つの異なる X クライアントプログラムを開始し、それらが完了するのを待機します。

```
systask command "xv" taskname=pgm1;
systask command "xterm" taskname=pgm2;
systask command "xcalc" taskname=pgm3;
waitfor _all_ pgm1 pgm2 pgm3;
```

WAITFOR ステートメントを使用して、同時に複数の SAS セッションを実行できます。次のステートメントは 3 つの異なる SAS ジョブを開始し、それら 3 つのジョブの実行が終了するまで現在の SAS セッションの実行を中断します。

```
systask command "sas myprog1.sas" taskname=sas1;
systask command "sas myprog2.sas" taskname=sas2;
```

```
systask command "sas myprog3.sas" taskname=sas3;
waitfor _all_ sas1 sas2 sas3;
```

注: この方法では、SAS は各コマンドの後に終了するため、パフォーマンスが低下します。SAS/CONNECT を使用して、並行する SAS セッションを実行することもできます。詳細については、*SAS/CONNECT User's Guide* を参照してください。.

SYSTASK コマンドを複数回使用する長期のジョブがある場合は、SYSTASK コマンドで WAITFOR ステートメントまたは CLEANUP オプションを使用してメモリをクリアします。WAITFOR ステートメントは、SYSTASK コマンドによって開始されたすべての完了プロセスの情報を削除することによってメモリを解放します。CLEANUP オプションは、特定のジョブが完了したときにメモリをクリアし、今後の使用に備えてメモリを開放します。ジョブが完了した後に WAITFOR ステートメントを使用する場合、そのジョブはすでに CLEANUP オプションによってクリーンアップされているので、このステートメントは無効になります。

SYSRC マクロ変数には、WAITFOR ステートメントのリターンコードが含まれています。WAITFOR ステートメントの実行が失敗する場合、SYSRC マクロ変数に非ゼロ値が含まれていることがあります。たとえば、WAITFOR ステートメントに構文エラーが含まれている場合があります。TIMEOUT オプションで指定された秒数が経過した場合は WAITFOR ステートメントの実行が終了し、次のいずれかが発生した場合は SYSRC が非ゼロ値に設定されます。

- 実行が終了しないタスクを 1 つ指定します。
- 複数のタスクと_ANY_オプション(デフォルトの設定)を指定しますが、それらのタスクの実行はいずれも終了しません。
- 複数のタスクと_ALL_オプションを指定しますが、それらのタスクのうち 1 つの実行が終了しません。

WAITFOR ステートメントが実行された後もステータス変数が依然として NULL であるタスクは、実行が完了していません。個別のタスクのステータス変数に関する説明は、“SYSTASK ステートメント: UNIX” (351 ページ)を参照してください。

関連項目:

ステートメント:

- “SYSTASK ステートメント: UNIX” (351 ページ)
- “X ステートメント: UNIX” (356 ページ)

その他の参照資料:

- *SAS/CONNECT User's Guide*
- “SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)

X ステートメント: UNIX

SAS セッション内から動作環境コマンドを発行します。

該当要素: 任意の場所

UNIX 固有: 有効なオペレーティングシステムコマンド

参照項目: “X Statement” (*SAS Statements: Reference*)

構文

X <'*operating-system-command*'>;

オプション引数

operating-system-command
UNIX コマンドを指定します。UNIX コマンドを 1 つ指定する場合は、それを引用符で囲む必要はありません。また、Korn シェルから SAS を実行する場合は、エイリアスが使用できません。

詳細

X ステートメントは SAS セッション内から UNIX コマンドを発行します。SAS は X ステートメントを即座に実行します。

X ステートメントも%SYSEXEC マクロプログラムステートメントも、DATA ステップ実行中での使用は想定されていません。ただし CALL SYSTEM ルーチンは、DATA ステップ内からの実行が可能です。例については、“CALL SYSTEM ルーチン: UNIX” (271 ページ)を参照してください。

注: X Window System の場合、引数のない X ステートメントはサポートされません。

関連項目:

“SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)

19 章

UNIX 版に固有のシステムオプション

UNIX 版に固有の SAS システムオプション

UNIX 環境に固有の動作や構文

このセクションでは、UNIX 環境に固有の動作または構文を含む SAS システムオプションについて説明します。各システムオプションの説明には、そのシステムオプションのどの要素が UNIX 版に固有であるのかを簡単に説明する"UNIX 固有"セクションが含まれています。"UNIX 固有"の情報が"すべて"の場合、そのシステムオプション

はこのドキュメントでのみ説明されています。それ以外の場合のシステムオプションは *SAS システムオプション: リファレンス*で説明されています。

コマンド行にかっこを使用する場合

コマンド行で引数を引用符で囲む場合、UNIX で引数が正しく解釈されるように開始かっこと終了かっこの前にバックスラッシュを使用する必要があります。

システムオプションの値にスペースが含まれる場合

システムオプションの値にスペースが含まれる場合は、コマンド行や構成ファイル内ではその値を引用符で囲む必要があります。次に、正しい構文例を示します。

```
-bufsize='3 k';

-bottommargin='2 in';
```

システムオプションの値にスペースが含まれない場合は、その値を引用符で囲む必要はありません。

```
-bufsize=3k;

-bottommargin=2in;
```

SAS システムオプションの設定法を指定する

SAS オプション間の相互通信によって、予期しない結果が生じる可能性があります。たとえば、FULLSTIMER システムオプションと NONOTES システムオプションの両方が設定されている場合、結果的に FULLSTIMER 情報は SAS ログに書き込まれません。構成ファイルに 1 つのオプションを設定し、他のオプションは OPTIONS ステートメントに設定できるため、このような問題は簡単には明らかになりません。

VALUE オプションと共に PROC OPTIONS ステートメントを実行してオプションの値をクエリする際、オプションの値は、そのオプションの設定に使用されたメソッド(場所)と共に SAS ログに表示されます。オプションを構成ファイルに設定した場合、**Config file name** フィールドにファイル名がリストで表示されます。たとえば、MEMSIZE システムオプションの値をクエリする際には、次の出力が SAS ログに表示されます。

```
proc options option=memsize value;
run;

Option Value Information for SAS Option MEMSIZE
Value: 100663296
Scope: SAS Session
How option value set: Config file
Config file name: /usr/local/SAS/SASFoundation/9.4/sasv9_local.cfg
```

WORK システムオプションの値をクエリするには、PROC OPTIONS ステートメントを実行します。WORK 値は、サーバー構成ファイル、環境設定またはコマンド行設定から設定できます。WORK パスは、最初にサーバーによって指定される WORK パスとホスト指定値および実行接尾辞を組み合わることによって生成されます。次の例は、SAS ログに書き込まれる情報を示しています。

```
proc options option=WORK value;
run;

Option Value Information for SAS Option WORK
```

```
Value: /sastemp/SAS_workA1234567_bcd89
Scope: SAS Session
How option value set: Config file
Config file name: /usr/local/SAS/SASFoundation/9.4/sasv9_local.cfg
```

制限オプション

制限オプションとは、サイト管理者によってその値が決定されるシステムオプションです。制限オプションをオーバーライドすることはできません。サイト管理者は、SAS の起動時に制限されるオプション値を指定する制限オプションテーブルを作成できます。制限オプションテーブルにリストで表示されるシステムオプションを変更しようと試みると、SAS ログにメッセージが表示されます。このメッセージでは、システムオプションがサイト管理者によって制限され、更新できることが示されます。詳細については、“Restricted Options” (*SAS System Options: Reference* 1 章)を参照してください。

ディクショナリ

ALIGNSASIOFILES システムオプション: UNIX

SAS データセットでデータのページを配置します。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 入力制御: データ処理

PROC OPTIONS GROUP= SASFILES

デフォルト: NOALIGNSASIOFILES

UNIX 固有: すべて

構文

–ALIGNSASIOFILES | –NOALIGNSASIOFILES

引数なし
必須の引数はありません。

詳細

SAS データセットは、1 ページ以上のデータが後に続く 1 つのヘッダーで構成されます。通常、ヘッダーサイズは、UNIX 動作環境では 8K、Windows 環境では 1K です。ALIGNSASIOFILES は、より効果的な I/O を可能にする境界へページが配置できるように、強制的にヘッダーをデータページと同じサイズにします。

ALTLOG システムオプション: UNIX

SAS ログの出力先を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES、LOGCONTROL

デフォルト: NOALTLOG

UNIX 固有: すべて

構文

–ALTLOG *file-specification* | **–NOALTLOG**

必須引数

-ALTLOG *file-specification*
代替 SAS ログの書き込み先の場所を指定します。*file-specification* 引数は、ディレクトリ、ファイル名、またはパスに関連付けられた環境変数への任意の有効な UNIX パスのいずれかとなります。ディレクトリへのパスのみを指定する場合、SAS ログは指定ディレクトリのファイル内に配置されます。ファイル名は *filename*.log となり、*filename* は SAS ジョブ名を表します。SAS を対話型で実行し、ディレクトリへのパスのみを指定する場合、ログの書き込み先は該当パスの sas.log になります。

-NOALTLOG
SAS ログがコピーされないよう指定します。

詳細

ALTLOG の基本

ALTLOG システムオプションは、SAS ログのコピーの書き込み先を指定します。SAS ログに書き込まれるすべてのメッセージは、*file-specification* で指定される場所にも書き込まれます。印刷用ログ出力を取り込むには、このオプションを使用します。

注: ログの任意の部分を外部ファイルにリダイレクトするには、PRINTTO プロシジャで LOG オプションを使用します。PROC PRINTTO のコードは現在のセッションの SAS ログには表示されませんが、ALTLOG システムオプションを指定して作成した SAS ログには表示されます。

注: OBJECTSERVER および NOTERMINAL システムオプションを指定して SAS を起動し、ログを指定しない場合、SAS によってすべてのログと代替ログメッセージが廃棄されます。

ALTLOG でのディレクティブの使用

時間、月および曜日などのリアルタイムイベントに基づき、ログコピーを開く/閉じるタイミングやその命名法を調整するには、ALTLOG システムオプションでディレクティブを使用します。ディレクティブのリストについては、“LOGPARM= System Option” (*SAS System Options: Reference*)を参照してください。

関連項目:

オプション:

- “ALTPRINT システムオプション: UNIX” (364 ページ)
- “PRINTTO プロシジャ: UNIX” (315 ページ)

その他の参照資料:

- “The SAS Log” (*SAS Language Reference: Concepts* 9 章)
- “SAS システムオプションを使用し、出力先を指定する” (100 ページ)

ALTPRINT システムオプション: UNIX

SAS プロシジャから出力ファイルの出力先を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: NOALTPRINT

UNIX 固有: すべて

構文

-ALTPRINT *file-specification* | **-NOALTPRINT**

必須引数

-ALTPRINT *file-specification*
書き込み対象のプロシジャ出力のコピーを指定します。*file-specification* 引数は、ディレクトリ、ファイル名、またはパスに関連付けられた環境変数への任意の有効な UNIX パスのいずれかとなります。ディレクトリへのパスのみを指定する場合、コピーは指定ディレクトリのファイル内に配置されます。ファイル名は *filename.lst* となり、*filename* は SAS ジョブ名を表します。SAS を対話型で実行し、ディレクトリへのパスのみを指定する場合、出力は sas.lst と言う名前のファイルに書き込まれます。

-NOALTPRINT
前の ALTPRINT 指定を無視するようにします。

詳細

ALTPRINT システムオプションは、SAS プロシジャ出力のコピーの書き込み先を指定します。SAS プロシジャに書き込まれるすべてのメッセージは、*file-specification* で指定される場所にも書き込まれます。印刷用プロシジャ出力を取り込むには、このオプションを使用します。

関連項目:

システムオプション:

- “ALTLOG システムオプション: UNIX” (362 ページ)

その他の参照資料:

- “SAS システムオプションを使用し、出力先を指定する” (100 ページ)

APPEND システムオプション: UNIX

SAS の起動時に使用され、指定システムオプションの最後に既存値に指定値を付加します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: なし

UNIX 固有: 構成ファイル、SAS 起動構文

注: このオプションは、サイト管理者では制限できません。詳細については、“Restricted Options” (*SAS System Options: Reference* 1 章)を参照してください。

参照項目: “APPEND= System Option” (*SAS System Options: Reference*)

構文

-APPEND *system-option new-option-value*

必須引数

system-option
FMTSEARCH、HELPLOC、MAPS、MSG、SASAUTOS、SASHELP、SASSCRIPT、SET、AUTOEXEC、CMPLIB のいずれかになります。

new-option-value
system-option の現在値に付加したい新しい値です。

詳細

デフォルトでは、AUTOEXEC、CMPLIB、FMTSEARCH、HELPLOC、MAPS、MSG、SASAUTOS、SASHELP、SASSCRIPT、SET システムオプションのいずれかを 2 回以上指定する場合、最後に指定した値が SAS で使用される値になります。これらのオプションのいずれかによってすでに指定されたパス名に追加パス名を付加したい場合、追加パス名の付加には APPEND システムオプションを使用する必要があります。たとえば、次の SAS コマンドを入力する場合、SAS によってヘルプファイルが検索される場所は**/apps/help** のみとなります。PROC OPTIONS の出力では**/apps/help** のみが表示されます。

```
sas -helploc /sas/help -helploc /apps/help
```

最初に**/sas/help**、次に**/apps/help** の順に SAS を表示したい場合は、APPEND オプションを使用します。

```
sas -helploc /sas/help -append helploc /apps/help
```

HELPLOC オプションの値の場合、PROC OPTIONS によって次が表示されます。

```
('/sas/help' '/apps/help')
```

関連項目:

システムオプション:

- “APPEND= System Option” (*SAS System Options: Reference*)
- “INSERT システムオプション: UNIX” (391 ページ)

AUTHPROVIDERDOMAIN: UNIX

ドメイン接尾辞を認証プロバイダに関連付けます。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 環境コントロール: 初期化および処理

PROC OPTIONS GROUP= EXECMODES

別名: AUTHPD

デフォルト: NULL

参照項目: “AUTHPROVIDERDOMAIN System Option” (*SAS System Options: Reference*)

AUTOEXEC システムオプション: UNIX

SAS autoexec ファイルを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: autoexec.sas (を参照 (366 ページ))

UNIX 固有: すべて

構文

-AUTOEXEC *file-specification* | **-NOAUTOEXEC**

-AUTOEXEC \ (*file-specification-1* <...*file-specification-n*>\)

必須引数

-NOAUTOEXEC
SAS で autoexec ファイルが処理されないように指定します。

file-specification
デフォルトの autoexec.sas ファイルのかわりに使用する SAS autoexec ファイルを指定します。*file-specification* 引数は、有効な Windows ファイル名か、またはパス名に関連付けられた環境変数になります。詳細については、“SAS Autoexec File ” (*SAS Companion for Windows* 1 章)を参照してください。

詳細

Autoexec ファイル

AUTOEXEC システムオプションは autoexec ファイルを指定します。autoexec ファイルには、SAS の起動時や別の SAS プロセス開始時に自動的に実行される SAS ステー

トメントが含まれます。autoexec ファイルには、任意の SAS ステートメントを含めることができます。たとえば、SAS セッションで定期的にアクセスする SAS ライブラリの LIBNAME ステートメントを autoexec ファイルに含めることができます。AUTOEXEC システムオプションを使用して、デフォルト autoexec ファイルを上書きできます。

SAS では、次の順序で AUTOEXEC システムオプションが検索されます。また、次の場所で検出した最初の AUTOEXEC システムオプションが使用されます。

1. コマンド行
2. SASV9_OPTIONS 環境変数
3. 構成ファイル

SAS では、検出した最初の AUTOEXEC オプションが使用され、他はすべて無視されます。

AUTOEXEC と NOAUTOEXEC システムオプションのいずれも検出されない場合、SAS では次の 3 つのディレクトリ内をこの順序で autoexec ファイルを検索します。

1. カレントディレクトリ
2. ホームディレクトリ
3. **!SASROOT** ディレクトリ(詳細については、“!SASROOT ディレクトリ” (445 ページ)を参照してください。)

   SAS は、最初に検出した autoexec ファイルを SAS セッションの初期化に使用します。

ユーザーセッションの autoexec ファイルの内容を表示する必要がある場合は、SAS の起動時に ECHOAUTO システムオプションを使用します。autoexec ファイルが使用中のデータソースを識別したい場合は、PROC OPTIONS ステートメントを使用してください。

```
proc options option=autoexec value;
run;
```

autoexec ファイルの挿入および付加

INSERT は、実行されるファイルを autoexec ファイルの前に加え、APPEND は、実行されるファイルを autoexec ファイルの後に加えます。autoexec ファイル内にファイルを結合するには、次のシステムオプションを AUTOEXEC システムオプションと一緒に使用します。INSERT (391 ページ) および APPEND (365 ページ)。autoexec ファイルは常に UNIX ファイルです。ファイル名に埋め込み空白または特殊文字が含まれる場合、ファイル名を引用符で囲む必要があります。ファイル名に埋め込み空白または特殊文字が含まれない場合は、1 つ以上のファイル名を指定するときに引用符はオプションです。

次の構文を使用して、autoexec ファイルを結合できます。

```
-autoexec "(/path1/autoexec.sas /path2/autoexec.sas /path3/autoexec.sas)"
```

次の構文で、INSERT システムオプションとともに使用できます。

```
-insert autoexec "a.sas" -insert autoexec "b.sas"
```

次の構文で、APPEND システムオプションとともに使用できます。

```
-append autoexec "a.sas" -append autoexec "b.sas"
```

関連項目:

システムオプション:

- “APPEND システムオプション: UNIX” (365 ページ)
- “INSERT システムオプション: UNIX” (391 ページ)

その他の参照資料:

- “システムオプションを使用し、SAS セッションをカスタマイズする” (18 ページ)

AUTOSAVELOC システムオプション: UNIX

Program Editor の自動保存ファイルの場所を指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ
カテゴリ:	環境コントロール: 表示
PROC OPTIONS GROUP=	ENVDISPLAY
デフォルト:	なし
UNIX 固有:	次の項目の有効な値: *pathname*
参照項目:	“AUTOSAVELOC= System Option” (*SAS System Options: Reference*)

構文

-AUTOSAVELOC *fileref* | *pathname*

AUTOSAVELOC *fileref* | *pathname*

必須引数

fileref
ファイル参照名を自動保存ファイルの保存場所に指定します。

pathname
自動保存ファイルのパス名を、ファイル名を含めて指定します。*pathname* は有効な UNIX パス名にする必要があります。

詳細

デフォルトでは、SAS で Program Editor の自動保存ファイル(pgm.asv)が開いているフォルダに保存されます。自動保存ファイルに別の場所を指定するには、AUTOSAVELOC システムオプションを使用します。

関連項目:

コマンド:

- “SETAUTOSAVE コマンド: UNIX” (242 ページ)

BUFNO システムオプション: UNIX

SAS データセットの処理用に割り当てるバッファ数を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: ファイル: SAS ファイル

PROC OPTIONS GROUP= SASFILES、PERFORMANCE

デフォルト: 1

UNIX 固有: デフォルト値

参照項目: “BUFNO= System Option” (*SAS System Options: Reference*)

構文

-BUFNO *n* | *n*K | *n*M| *n*G | *hex*X | MIN | MAX

BUFNO=*n* | *n*K | *n*M| *n*G| *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
バッファ数を 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、値が `8` の場合は 8 バッファ、値が `.782k` の場合は 801 バッファ、値が `3m` の場合は 3,145,728 バッファが指定されます。

***hex*X**
バッファ数を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、`2dx` の場合は 45 バッファが指定されます。

MIN
バッファ数を 0 に設定し、SAS でデフォルト値の 1 を使用するように要求します。

MAX
バッファ数を 2,147,483,647 に指定します。

詳細

バッファ数はデータセットの永久属性ではありません。有効になるのは、現在の SAS セッションまたはジョブに限られます。

BUFNO=は、入力、出力または更新用に開かれた SAS データセットに適用されます。

BUFNO=を使用すると、特定の SAS データセットに必要な入力/出力処理の数を制限することで、実行時間を改善できます。ただし、実行時間の改善の代償として、メモリ消費量の増加が伴います。

UNIX 版では、割当可能なバッファの最大数は、使用可能なメモリ量によって決定されます。

BUFSIZE システムオプション: UNIX

出力 SAS データセットの永久バッファページのサイズを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: ファイル: SAS ファイル

PROC OPTIONS GROUP= SASFILES、PERFORMANCE

デフォルト: 0

UNIX 固有: 有効範囲

参照項目: “BUFSIZE= System Option” (*SAS System Options: Reference*)

構文

-BUFSIZE *n* | *n*K | *n*M | *n*G | *hex*X | MAX

BUFSIZE=*n* | *n*K | *n*M | *n*G | *hex*X | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
バッファページサイズを 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、値が `8` の場合は 8 バイト、値が `.782k` の場合は 801 バイト、値が `3m` の場合は 3,145,728 バイトが指定されます。

***hex*X**
バッファページサイズを 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、`2dx` の場合はバッファページサイズが 45 バイトに設定されます。

MAX
バッファページサイズを 2,147,483,647 に設定します。

詳細

バッファページサイズの範囲は、1K から 2G–1 までとなります。

SAS データセットの作成時にゼロ以外の値を指定する場合、BASE エンジンによってその値が使用されます。その値が 1 つ以上のオブザベーションが保持できないか、または 1K の倍数ではない場合、エンジンによって値は端数が切り上げられ 1K の倍数となります。

CATCACHE システムオプション: UNIX

開いておける SAS カタログ数を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: ファイル: SAS ファイル

PROC OPTIONS GROUP= SASFILES

デフォルト: 0

UNIX 固有: 次の項目の有効な値: *n*

参照項目: “CATCACHE= System Option” (*SAS System Options: Reference*)

構文

-CATCACHE *n* | *n*K | MIN | MAX

必須引数

***n* | *n*K**
キャッシュメモリ内に 1 (*n*)または 1,024 (*n*K)の倍数で保持できる、開いているファイルディスクリプタの数を指定します。キロバイトの数には小数値を指定できます。たとえば、開いているファイルディスクリプタの数は値が `8` の場合は 8 個、値が `.782k` の場合は 801 個、値が `3k` の場合は 3,072 個が指定されます。

n が 0 より大きい場合、カタログを閉じるかわりに、開いているファイルディスクリプタの指定数がキャッシュメモリ内に SAS によって配置されます。

MIN
キャッシュメモリ内で 0 に保持できる、開いているファイルディスクリプタの数を設定します。

MAX
キャッシュメモリ内で 32,767 に保持できる、開いているファイルディスクリプタの数を設定します。

詳細

CATCACHE システムオプションを使用して開いたままにする SAS カタログ数を指定すると、同じカタログを繰り返し閉じたり開いたりしなくてもよくなります。

関連項目:

“Definitions for Optimizing System Performance” (*SAS Language Reference: Concepts* 12 章)

CLEANUP システムオプション: UNIX

リソース不足状態の処理方法を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: エラー処理

PROC OPTIONS GROUP= ERRORHANDLING

デフォルト: 対話型モードの場合は CLEANUP、それ以外の場合は NOCLEANUP

UNIX 固有: 対話型ラインモードおよびバッチモードで実行時の動作

参照項目: “CLEANUP System Option” (*SAS System Options: Reference*)

構文

-CLEANUP | **-NOCLEANUP**

CLEANUP | **NOCLEANUP**

必須引数

CLEANUP
全体のセッション中に、SAS が実行に不必要なリソースの自動連続クリーンアップの実行を試みるように指定します。不必要なリソースには、ユーザーに表示されないリソース(キャッシュメモリなど)や、ユーザーに表示されるリソース(**Keys** ウィンドウなど)が含まれます。

CLEANUP では、SAS がディスクのクリーンアップを試行する前に確認は要求されません。ただし、ディスク容量不足状態が発生し、表示がプロセスにアタッチされる場合には、CLEANUP オプションがオンでもメニュー選択の確認が要求されます。ディスク容量不足状態の確認を要求されたくない場合は、NOTERMINAL オプションとともに CLEANUP オプションを使用してください。

CLEANUP オプションがオンの場合、SAS では自動連続クリーンアップが実行されます。十分な量のリソースが回復されない場合、リソースに対するリクエストは失敗し、該当するエラーメッセージが SAS ログに書き込まれます。

確認要求を調整するプロセスには表示がアタッチされないため、バッチモードでは CLEANUP がデフォルトになります。

NOCLEANUP
SAS でユーザーがリソース不足状態の処理方法を選択できるよう指定します。NOCLEANUP が有効で、リソース不足のために SAS が実行できない場合、SAS ではユーザーに表示されないリソース(キャッシュメモリなど)のクリーンアップが自動的に試行されます。ただし、ユーザーに表示されるリソース(**Keys** ウィンドウなど)は自動的にクリーンアップされません。かわりに、ディスクのクリーンアップが試行される前に SAS によって確認が要求されます。

詳細

CLEANUP システムオプションは、SAS でリソース不足状態に陥った時に、クリーンアップするアイテムのメニューをユーザーが確認する必要があるかどうかを指定します。バッチモードでは、SAS はこのオプションを無視し、リソース不足状態に陥った場合は、SAS セッションが終了します。

CONFIG システムオプション: UNIX

SAS システムオプションの値を初期化またはオーバーライドする際に使用される構成ファイルを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数、SASV9_CONFIG 環境変数
カテゴリ: 環境コントロール: ファイル
PROC OPTIONS GROUP= ENVFILES
デフォルト: sasv9.cfg (“SAS 構成ファイルの処理の優先順位” (24 ページ)を参照)
UNIX 固有: すべて

構文

-CONFIG *file-specification* | **–NOCONFIG**

必須引数

-CONFIG *file-specification*
読み取り対象の構成ファイルを指定します。*file-specification* は、有効な UNIX ファイル名に対して解決される必要があります。

-NOCONFIG
前に CONFIG 指定がある場合はこれを無視し、デフォルトのシステムオプションを使用する必要があるように指定します。

詳細

構成ファイルには、SAS が起動されるたびに自動的に実行されるシステムオプション指定が含まれます。

構成ファイルを指定すると、デフォルトの構成ファイルリストは無効になります。

関連項目:

“システムオプションを使用し、SAS セッションをカスタマイズする” (18 ページ)

DEVICE システムオプション: UNIX

SAS/GRAPH ソフトウェアのグラフィック出力用のデバイスドライバを指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数、GOPTIONS ステートメント
カテゴリ:	グラフィック: ドライバ設定
PROC OPTIONS GROUP=	GRAPHICS
デフォルト:	なし
UNIX 固有:	有効なデバイスドライバ
参照項目:	“DEVICE= System Option” (*SAS/GRAPH: Reference*)

構文

-DEVICE *device-driver-name*

DEVICE=*device-driver-name*

必須引数

device-driver-name
グラフィック出力用のデバイスドライバ名を指定します。

詳細

UNIX 版で使用可能なデバイスドライバのリストを参照するには、GDEVICE プロシジャを使用します。SAS ウィンドウ環境を使用している場合は、次のステートメントを送信してください。

```
proc gdevice catalog=sashelp.devices;
run;
```

SAS を対話型ラインモードまたはバッチモードで実行している場合は、次のステートメントを送信してください。

```
proc gdevice catalog=sashelp.devices nofs;
list _all_;
run;
```

関連項目:

SAS/GRAPH: Reference

ECHO システムオプション: UNIX

標準出力にエコーされるメッセージを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: なし

UNIX 固有: すべて

構文

-ECHO '*message*' | **-NOECHO**

必須引数

-ECHO '*message*'
コンピュータにエコーされるメッセージのテキストを指定します。メッセージが 2 語以上の場合は、テキストを一重または二重引用符で囲む必要があります。それ以外の場合は、引用符は不要です。

-NOECHO
メッセージがコンピュータにエコーされないように指定します。

詳細

複数の ECHO オプションを指定できます。文字列の検索順序は、SAS での検出順になります。その順序の決まり方に関する詳細については、“複数の場所で指定されているシステムオプションの処理方法” (21 ページ)を参照してください。

たとえば、次のコードを指定できます。

```
-echo 'SAS 9.4 under UNIX is initializing.'
```

SAS の初期化時にメッセージが **Log** ウィンドウに表示されます。

関連項目:

システムオプション:

- “ECHOAUTO System Option” (*SAS System Options: Reference*)

EDITCMD システムオプション: UNIX

HOSTEDIT コマンドで使用するホストエディタを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: 表示

PROC OPTIONS GROUP= ENVDISPLAY

デフォルト: なし

UNIX 固有: すべて

構文

-EDITCMD "*host-editor-pathname editor-option(s)*"

EDITCMD="*host-editor-pathname editor-option(s)*"

詳細

EDITCMD システムオプションは、動作環境に実行されるコマンドを指定します。vi などの端末ベースのエディタを使用している場合、端末エミュレータウィンドウ内部でエディタを実行するコマンドを指定する必要があります。

EDITCMD オプションを定義するには、構成ファイルの一部として SASV9_OPTIONS 環境変数を使用するか、またはコマンド行で定義を SAS に自動的に使用できるようにします。オプションは、引用符で囲んだ文字列として指定する必要があります。一重または二重引用符のいずれかを使用できます。SAS セッション中に EDITCMD オプションの値を変更するには、OPTIONS ステートメントを実行します。

指定するホストエディタは、HOSTEDIT コマンドを実行する時に使用されます。HOSTEDIT コマンドが有効になるのは、ウィンドウ環境で SAS を実行する場合に限られます。

完全なパス名を指定しない場合、$PATH 環境変数で指定したパス名が SAS によって検索されます。たとえば、vi を使用する場合、次を指定することになります。

```
sas -editcmd "/usr/bin/X11/xterm -e /usr/bin/vi"
```

関連項目:

"UNIX 環境でホストエディタがサポートされるように SAS を構成する" (167 ページ)

EMAILSYS システムオプション: UNIX

電子メールの送信に使用する電子メールプロトコルを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ

カテゴリ: 通信: 電子メール

PROC OPTIONS GROUP= EMAIL

デフォルト: SMTP

UNIX 固有: すべて

構文

-EMAILSYS SMTP | *name-of-script*

EMAILSYS=SMTP | *name-of-script*

必須引数

SMTP
SMTP (Simple Mail Transfer Protocol)電子メールインターフェイスを指定します。

name-of-script
SAS 内から電子メールの送信に使用するスクリプトの種類を指定します。電子メール添付ファイルの送信をサポートしていない外部スクリプトもあります。このようなスクリプトは、SAS によってサポートされません。

詳細

EMAILSYS システムオプションは、SAS 内から電子メールの送信に使用する電子メールプロトコルの種類を指定します。SMTP を指定すると、UNIX での電子メール添付ファイルの送信がサポートされますが、ユーザーのサイト構成によっては、EMAILHOST=および EMAILPORT=システムオプションの値を変更する必要がある場合もあります。

EMAILSYS オプションは、SAS セッション中にいつでも設定できます。

関連項目:

システムオプション:

- "EMAILHOST= System Option" (*SAS System Options: Reference*)
- "EMAILPORT System Option" (*SAS System Options: Reference*)

その他の参照資料:

- "Sending E-Mail through SMTP" (*SAS Language Reference: Concepts* 38 章)
- "FILENAME ステートメント(EMAIL)を使用し、電子メールを送信する" (82 ページ)
- "UNIX 環境で SAS セッションからメールを送信する" (164 ページ)

ENGINE=システムオプション: UNIX

SAS ライブラリに使用するデフォルトのアクセス方法を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: ファイル: SAS ファイル

PROC OPTIONS GROUP= SASFILES

デフォルト: V9

UNIX 固有: 次の項目の有効な値: *engine-name*

参照項目: “ENGINE= System Option” (*SAS System Options: Reference*)

構文

-ENGINE *engine-name*

必須引数

engine-name
UNIX 版では次のいずれかとなります。

BASE | V9
SAS 9 から SAS 9.4 までのファイルに使用するデフォルトの SAS エンジンを指定します。

V8
すべての SAS バージョン 9 ファイルを指定します。

V7
すべての SAS バージョン 7 ファイル用の SAS エンジンを指定します。

V6
リリース 6.12 を介してリリース 6.09 用の SAS エンジンを指定します。このエンジンは読み取り専用です。

関連項目:

35 章: “SAS Engines” (*SAS Language Reference: Concepts*)

FILELOCKS システムオプション: UNIX

ファイルのロックのオン/オフおよびファイルをロックできない場合に必要なアクションを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ

カテゴリ: ファイル: 外部ファイル | SAS ファイル

PROC OPTIONS GROUP= EXTFILES、SASFILES、ENVFILES

デフォルト: FAIL

UNIX 固有: すべて

構文

-FILELOCKS *setting path* | *path setting*

-FILELOCKS NONE | FAIL | CONTINUE | RESET

FILELOCKS=(*setting path* | *path setting*)

FILELOCKS=NONE | FAIL | CONTINUE | RESET

必須引数

setting
指定パスの動作環境のロック値を指定します。次の値が有効です。

- NONE
- FAIL
- CONTINUE
- RESET

path
UNIX ディレクトリへのパスを指定します。パスは一重または二重引用符で囲みます。

ヒント 引数の *path* には、環境変数を含めることができます。

NONE
ファイルのロックをオフにします。NONE は、SAS がファイルでの既存のロックを確認せずにファイルを開く指定です。NONE では、ファイルにオペレーティングシステムロックがかかりません。このようなファイルは、共有更新アクセスから保護されません。

ヒント NONE によって内部ロックは抑制されません。

FAIL
ファイルのロックをオンにします。FAIL は、SAS がファイルにオペレーティングシステムロックをかけようとする指定です。ファイルへのアクセスは、ファイルがすでにロックされている場合、またはファイルをロックできない場合に拒否されます。FAIL は、FILELOCKS のデフォルト値です。

CONTINUE
ファイルのロックをオンにします。CONTINUE は、SAS がファイルにオペレーティングシステムロックをかけようとする指定です。ファイルがすでに他のユーザーにロックされている場合は、ファイルを開くことができません。別の何らかの理由によってファイルをロックできない場合(ファイルシステムがロック機能をサポートしていない場合など)、ファイルは開かれ、警告メッセージがログに送られます。

ヒント CONTINUE によって内部ロックは抑制されません。

RESET
以前の FILELOCKS 設定をすべて削除する指定で、グローバル設定はデフォルト値の FAIL にリセットされます。**FILELOCKS=(*setting path*|*path setting*)** の構文を使用する場合、RESET によってリセットされるのは、*path* にあるファイルに限られます。

詳細

ファイルロック機能の基本

前の SAS リリースでは、FILELOCKS システムオプションによってロックできたのは SAS ファイルに限られていました。SAS 9.2 以降では、FILELOCKS システムオプションによって外部ファイルもロックできるようになりました。

FILELOCKS システムオプションでは、FILELOCKS システムオプションで設定したグローバル設定に基づき、外部ファイルと SAS ファイルをロックできます。外部ファイルのロックは、開いているすべてのファイルに適用されます。

パスごとに異なる設定を確立するには、FILELOCKS オプションの複数のインスタンスを使用します。1 つのパスには、別のパスのサブディレクトリを指定できます。この場合、現在指定されている中で最も一致度が高い有効なパスがオペレーティングシステムのファイルロックを制御します。次の例は、構成ファイルで FILELOCKS オプションの複数のインスタンスを指定する方法を示します。

```
filelocks=('/u/myuserid/temp' NONE)
filelocks=('/tmp' CONTINUE)
```

FILELOCKS オプションの値が *path* および *setting* の集合である場合、パスを引用符で囲む必要があります。コマンド行で FILELOCKS を使用する場合、引用符は不要です。

注: データの破損を防ぐため、FILELOCKS を NONE または CONTINUE に設定することはお勧めしません。

path および setting 引数を使用したパスのリセット

設定を特定のディレクトリおよびそのサブツリーに適用するには、*path* および *setting* 引数を指定します。*setting* の値を RESET に設定する場合、*path* および *setting* の各値は削除されます。

たとえば、**filelocks=('/' reset)** の場合、*path* および *setting* の現在の値は削除され、FILELOCKS でデフォルト値 **('/' fail)** にリセットされます。

FILELOCKS を FAIL に設定する場合

FILELOCKS を FAIL(デフォルト値)に設定する場合、次のアクションが発生します。

- SAS では、2 つのセッションで同じ SAS ファイルが更新または出力のために同時に開かれなくなります。
- SAS では、更新または出力のために別の SAS セッションが開いている SAS ファイルが、1 つのセッションで読み取られなくなります。
- SAS では、読み取りモードで別の SAS セッションが開いている SAS ファイルに対して、1 つのセッションで書き込まれなくなります。

関連項目:

システムオプション:

- “WORKINIT システムオプション: UNIX” (436 ページ)

FILELOCKWAIT=システムオプション: UNIX

ロックされたファイルが使用可能になるまで SAS が待機する秒数を設定します。

該当要素:	構成ファイル、SAS 起動
カテゴリ:	ファイル: SAS ファイル
PROC OPTIONS GROUP=	SASFILES
デフォルト:	0
UNIX 固有:	すべて

構文

FILELOCKWAIT=*wait-time*

必須引数

wait-time
ロックされたファイルが使用可能になるまで SAS が待機する時間を秒単位で指定します。

詳細

通常、アクセスするファイルがロックされている場合、SAS ではエラーが返されます。FILELOCKWAIT=システムオプションを使用すると、SAS は、時間制限を設けて、ロックされた SAS ファイルが使用できるようになるのを待機できます。FILELOCKWAIT=に *wait-time* の値を設定すると、SAS は指定された時間までそのファイルが使用可能になるのを待った後、失敗します。制限時間に達すると、SAS はロックされたファイルのエラーを返し、DATA ステップは失敗します。ロックされたファイルの待機に設定できる最大時間は 600 秒(10 分)です。FILELOCKWAIT=を 0 に設定すると、SAS は即座に失敗します。

このオプションは、FILELOCKWAITMAX=システムオプションを使用して FILELOCKWAIT=のデフォルト値を変更可能な、主にシステム管理者が使用します。FILELOCKWAITMAX=では、FILELOCKWAIT=オプションの最大値を設定します。デフォルトの最大値は 600 秒(10 分)ですが、システム管理者は、たとえば 300 秒(5 分)や 1200 秒(20 分)などの制限を設定できます。FILELOCKWAITMAX=の変更では FILELOCKWAIT=の値に影響はありません。FILELOCKWAITMAX=では、単に FILELOCKWAIT=で可能な最大値が設定されるのみです。FILELOCKWAIT=オプションはシステム管理者によって制約を受けることがあります。

FILELOCKWAIT=は、システムオプションですが、LIBNAME オプションでもあります。システムオプションはすべての SAS I/O ファイルに適用されます。LIBNAME オプションはライブラリのメンバにのみ適用されます。これはシステムオプションに優先されます。

関連項目:

システムオプション:

- “FILELOCKS システムオプション: UNIX” (377 ページ)
- “FILELOCKWAITMAX=システムオプション: UNIX” (380 ページ)

FILELOCKWAITMAX=システムオプション: UNIX

ロックされたファイルを SAS が待機する時間の上限を設定します。

該当要素:	構成ファイル、SAS 起動
カテゴリ:	ファイル: SAS ファイル
PROC OPTIONS GROUP=	SASFILES
デフォルト:	600
UNIX 固有:	すべて

構文

FILELOCKWAITMAX=*wait-time*

必須引数

wait-time

ロックされたファイルが使用可能になるまで SAS が待機する時間を秒単位で指定します。

デフォルト 600

範囲 0–600

操作 FILELOCKWAITMAX=システムオプションを指定すると、別のプロセスによってロックされた SAS ファイルが解放されるまで待機している 1 つ以上の SAS/SHARE サーバーおよびクライアントの各セッションに影響を及ぼす可能性があります。1 つ以上の待機条件により、SAS/SHARE Server およびクライアントプロセスの失敗を招くおそれがあります。

SAS/SHARE の処理エラーが発生する可能性を予防するためには、FILELOCKWAITMAX=0 を設定します。このオプションを設定すると、ロックされたファイルが解放されるまでに SAS/SHARE Server およびクライアントが待機する時間が取り消されます。待機時間のキャンセルが、プロセス失敗を防ぐ場合があります。

詳細

FILELOCKWAIT=システムオプションを使用する場合に、FILELOCKWAITMAX=システムオプションでは、ロックされたファイルが使用可能になるのを待って失敗に至るまでの、SAS が待機する時間の最大値を設定できます。FILELOCKWAIT=システムオプションを使用しない場合、FILELOCKWAITMAX=の値は待機時間に影響を与えません。

通常、アクセスするファイルがロックされている場合、SAS ではエラーが返されます。FILELOCKWAIT=システムオプションを使用する場合、SAS は指定された秒数の間そのファイルが使用可能になるのを待った後、失敗します。デフォルトでは、FILELOCKWAIT=の最大値は 600 秒です。

システム管理者は FILELOCKWAITMAX=システムオプションを使用して最大値を変更できます。FILELOCKWAITMAX=0 の設定は、FILELOCKWAIT=オプションの効果をオフにします。

FILELOCKWAIT=は、システムオプションですが、LIBNAME ステートメントオプションでもあります。システムオプションはすべての SAS I/O ファイルに適用されます。LIBNAME オプションはライブラリのメンバにのみ適用されます。LIBNAME オプションがシステムオプションを上書きします。FILELOCKWAITMAX=はシステムオプションのみです。

関連項目:

システムオプション:

- “FILELOCKS システムオプション: UNIX” (377 ページ)
- “FILELOCKWAIT=システムオプション: UNIX” (379 ページ)

FMTSEARCH システムオプション: UNIX

フォーマットカタログの検索順位を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVDISPLAY

デフォルト: (WORK LIBRARY)

UNIX 固有: 次の項目の有効な値: *catalog-specification*

参照項目: “FMTSEARCH= System Option” (*SAS System Options: Reference*)

構文

-FMTSEARCH *(catalog-specification-1 ... catalog-specification-n)*

FMTSEARCH=*(catalog-specification-1 ... catalog-specification-n)*

必須引数

catalog-specification
目的のメンバが検出されるまでのフォーマットカタログの検索順位を指定します。*libref* の値は、*libref* または *libref.catalog* のいずれかとなります。ライブラリ参照名のみを指定する場合、SAS では FORMATS がカタログ名であると仮定されます。

注 *libref* の値は、大文字にする必要があります。

詳細

追加の *catalog-specification* エントリを加えるには、“INSERT システムオプション: UNIX” (391 ページ)または “APPEND システムオプション: UNIX” (365 ページ)を使用します。

関連項目:

システムオプション:

- “APPEND= System Option” (*SAS System Options: Reference*)
- “INSERT= System Option” (*SAS System Options: Reference*)

FONTSLOC システムオプション: UNIX

SAS セッション中に読み込まれる SAS フォントの場所を指定します。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 環境コントロール: 表示

PROC OPTIONS GROUP= ENVDISPLAY

デフォルト: !SASROOT/misc/fonts

UNIX 固有: 有効なパス名

参照項目: “FONTSLOC= System Option” (*SAS System Options: Reference*)

構文

-FONTSLOC "*directory-specification*"

必須引数

"directory-specification"
SAS セッション中に読み込まれる SAS フォントを格納するディレクトリを指定します。*directory-specification* は、二重引用符で囲む必要があります。

詳細

ディレクトリは、動作環境の有効なパス名にする必要があります。

FULLSTIMER システムオプション: UNIX

利用可能なすべてのシステムパフォーマンス統計情報および日時スタンプを SAS ログに書き込むかどうかを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: NOFULLSTIMER

UNIX 固有: すべて

構文

-FULLSTIMER | **-NOFULLSTIMER**

FULLSTIMER | **NOFULLSTIMER**

必須引数

FULLSTIMER
各ステップおよび SAS セッション全体で使用されたホスト依存型リソースのリストを SAS ログに書き込みます。出力には日時スタンプが含まれます。

NOFULLSTIMER
リソースや日時スタンプのリストを SAS ログに書き込みません。

詳細

SAS では、FULLSTIMER から統計情報を取得するに当たりユーザーの動作環境の UNIX システムコールが使用されます。出力には日時スタンプがリストで表示されま

す。統計情報の動作や形式を変更するには、STIMFMT システムオプションを使用します。

次の例は、FULLSTIMER 出力の例です。

ログ 19.1 *FULLSTIMER 出力*

```
NOTE: SAS initialization used:
real time 0.84 seconds
user cpu time 0.03 seconds
system cpu time 0.03 seconds
Memory 236k
OS Memory 5672k
Timestamp 4/12/2013 9:13:39 AM
Page Faults 37
Page Reclaims 0
Page Swaps 0
Voluntary Context Switches 1336
Involuntary Context Switches 1
Block Input Operations 39
Block Output Operations 0
```

注: FULLSTIMER および STIMER システムオプションの両方を設定する場合、FULLSTIMER 統計情報はログに書き込まれます。

FULLSTIMER には次の統計情報が表示されます。

表 19.1 *FULLSTIMER 統計情報の説明*

統計情報	説明
Real Time	SAS ジョブの処理に費やされるリアルタイム時間(時計時間)。Real time は経過時間ともいいます 。
User CPU Time	ユーザーのプログラムで費やされる CPU 時間。
System CPU Time	SAS コードの実行をサポートするオペレーティングシステムのタスク(システムオーバーヘッドタスク)の実行に費やされる CPU 時間。
Memory	1 つのステップの実行に必要なメモリ容量。
OS Memory	ステップの実行中に SAS に利用できるオペレーティングシステムメモリの最大容量。
Timestamp	1 つのステップが実行された日付と時刻。
Page Faults	SAS からアクセスが試行されたものの、メインメモリ内にはなかったため、I/O アクティビティが必要になったページ数。
Page Reclaims	I/O アクティビティなしにアクセスされたページ数。
Page Swaps	1 つのプロセスがメインメモリから交換された回数。
Voluntary Context Switches	ディスクドライブなどのリソースの制約のため、SAS プロセスが一時停止する必要があった回数。

統計情報	説明
Involuntary Context Switches	他のプロセスの実行を許可するため、オペレーティングシステムから強制的に SAS セッションの処理を一時停止させた回数。
Block Input Operations	データをメモリに読み取るために実行される I/O 処理の回数。
Block Output Operations	データをファイルに書き込むために実行される I/O 処理の回数。

これらの統計情報に関する詳細については、**getrusage()** および **times()** UNIX システムコールに関する man ページを参照してください。

注: SAS 9 からは、一部のプロシジャで複数のスレッドが使用されています。複数の CPU を持つコンピュータでは、複数のスレッドで同時にオペレーティングシステムを実行できます。その結果、CPU 時間は FULLSTIMER 出力の処理時間を超過する場合があります。たとえば、SAS プロシジャでは 2 つの別々の CPU で同時に実行される 2 つのスレッドが使用される可能性があります。CPU 時間の値は、次の計算式に従って計算されます。

```
CPU1 time + CPU2 time = total CPU time
1 second + 1 second = 2 seconds
```

CPU1 は、CPU2 が同一の SAS プロセスの別個のスレッドを実行すると同時にスレッドを実行するため、理論上は処理時間の 1 秒間に CPU 時間では 2 秒間を消費することができます。

関連項目:

システムオプション:

- “STIMEFMT システムオプション: UNIX” (425 ページ)
- “STIMER システムオプション: UNIX” (429 ページ)

HELPHOST システムオプション: UNIX

リモートブラウジングシステムが表示されるホストコンピュータの名前を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、または SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ヘルプ

PROC OPTIONS GROUP= HELP

デフォルト: NULL

UNIX 固有: すべて

参照項目: “HELPHOST System Option” (*SAS System Options: Reference*)

構文

HELPHOST="*host*"
-HELPHOST "*host*"

必須引数

"*host*"
リモートブラウジングシステムが表示されるコンピュータの名前を指定します。引用符またはかっこが必要になります。最大文字数は 2048 です。

詳細

HELPHOST オプションを指定しない場合、リモートブラウジングシステムは、X 表示設定で指定されるホストに表示されます。

例

例 1: SAS 起動

UNIX 環境の HELPHOST システムオプションを指定するための構文は、次の例で示されます。

```
sas94/helphost "my.computer.com"
```

例 2: OPTIONS ステートメント: UNIX

OPTIONS ステートメントを使用して HELPHOST システムオプションを指定するための構文は、次の例で示されます。

```
options helphost="my.computer.com";
```

関連項目:

“リモートブラウザサーバーのインストール” (132 ページ)

HELPINDEX システムオプション: UNIX

SAS オンラインヘルプとドキュメントに 1 つ以上のインデックスファイルを指定します。

該当要素:	構成ファイル、SAS 起動
カテゴリ:	環境コントロール: ヘルプ
PROC OPTIONS GROUP=	HELP
デフォルト:	/help/common.hlp/index.txt、/help/common.hlp/keywords.htm、common.hhk
UNIX 固有:	アプレットおよび HTML ファイルは、HELPLOC オプションで指定されるパス内に配置する必要があります。

構文

-HELPINDEX *index-pathname-1* <*index-pathname-2* <*index-pathname-3*>>

必須引数

index-pathname
SAS オンラインヘルプとドキュメントで使用されるインデックスの部分パス名を指定します。*index-pathname* 引数は、次のいずれかまたは全部となります。

/help/*applet-index-filename*
UNIX 環境で SAS ドキュメントの Java アプレットで使用されるインデックスファイルの部分パス名を指定します。*applet-index-filename* のファイル拡張子は.txt にする必要があり、HELPLOC システムオプションによって指定されるパスに配置する必要があります。デフォルトは**`/help/common.hlp/index.txt`** です。

インデックスファイルに必要な形式については、デフォルトのインデックスファイルを参照してください。

/help/*accessible-index-filename*
UNIX または z/OS 環境で SAS オンラインヘルプとドキュメントで使用されるアクセス可能なインデックスファイルの部分パス名を指定します。アクセス可能なインデックスファイルは、Web ブラウザで使用可能な HTML ファイルです。*accessible-index-filename* のファイル拡張子は.htm にする必要があり、HELPLOC システムオプションで指定されるパスに配置する必要があります。デフォルトのパス名は**`/help/common.hlp/keywords.htm`** です。

インデックスファイルに必要な形式については、デフォルトのインデックスファイルを参照してください。

HTML-Help-index-pathname
Windows 環境で SAS オンラインヘルプとドキュメントで使用される Microsoft HTML Help インデックスのパス名を指定します。デフォルトのパス名は**`common.hhk`** です。Microsoft HTML Help のインデックス作成に関する詳細については、Microsoft HTML Help ドキュメントを参照してください。

詳細

SAS によって提供されるインデックスのかわりに使用したいカスタマイズ済みのインデックスがある場合は、HELPINDEX オプションを使用してください。複数のオペレーティングシステムで SAS を起動する 1 つの構成ファイルがある場合は、すべての部分パス名を HELPINDEX オプションで指定できます。各型のパス名は 1 つしか指定できませんが、パス名の順序は重要ではありません。

HELPINDEX オプションで UNIX または z/OS オペレーティング環境のパス名を指定する場合、部分パス名内の**`/help/`**を HELPLOC オプションで指定したパス名に置き換えることで、SAS によって完全パスが決定されます。HELPLOC オプションに複数のパス名が含まれる場合、SAS によって指定インデックスの各パスが検索されます。

たとえば、HELPINDEX の値が**`/help/common.hlp/myindex.htm`** で、HELPLOC の値が**`/u/myhome/myhelp`** の場合、インデックスの完全パスは**`/u/myhome/myhelp/common.hlp/myindex.htm`** となります。

関連項目:

システムオプション:

- “HELPLOC システムオプション: UNIX” (388 ページ)

HELPLOC システムオプション: UNIX

テキストの場所および SAS オンラインヘルプとドキュメントの表示に使用される機能のインデックスファイルを指定します。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 環境コントロール: ヘルプ

PROC OPTIONS GROUP= HELP

デフォルト: !SASROOT/X11/native_help/en

UNIX 固有: デフォルトの *pathname*

構文

-HELPLOC (*pathname*<, *pathname-2* …, *pathname-n*>)

必須引数

pathname
SAS オンラインヘルプとドキュメントが配置される 1 つ以上のディレクトリのパス名を指定します。

詳細

HELPLOC システムオプションに値を指定すると、結合値リストの最初にその値が挿入され、そのリストの最後がデフォルト値になります。この動作により、SAS ヘルプとドキュメントへのアクセス権を失わずにサイトのヘルプにアクセスできます。

パス名を追加するには、INSERT または APPEND システムオプションを使用してください。詳細については、“INSERT システムオプション: UNIX” (391 ページ)および“APPEND システムオプション: UNIX” (365 ページ)を参照してください。

例: HELPLOC システムオプションの使用

次のコマンドには、2 つの HELPLOC 指定が含まれます。

```
sas -insert helploc /app2/help -insert helploc /app1/help -append
```

システムオプションの値は次のとおりです。

```
/app1/help, /app2/help, !SASROOT/X11/native_help
```

関連項目:

システムオプション:

- “APPEND= System Option” (*SAS System Options: Reference*)
- “INSERT= System Option” (*SAS System Options: Reference*)

HELPTOC システムオプション: UNIX

SAS オンラインヘルプとドキュメントの目次ファイルを指定します。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 環境コントロール: ヘルプ

PROC OPTIONS GROUP= HELP

デフォルト: /help/helpnav.hlp/config.txt、/help/common.hlp/toc.htm、common.hhc

UNIX 固有: アプレットおよび HTML ファイルは、HELPLOC オプションで指定されるパス内に配置する必要があります。

構文

-HELPTOC *TOC-pathname-1* < *TOC-pathname-2* < *TOC-pathname-3*>>

必須引数

TOC-pathname
SAS オンラインヘルプとドキュメントで使用される目次の部分パス名を指定します。*TOC-pathname* は次のいずれかまたは全部となります。

/help/*applet-TOC-filename*
UNIX 環境で SAS ドキュメントの Java アプレットで使用される目次の部分パス名を指定します。*applet-TOC-filename* のファイル拡張子は.txt にする必要があり、HELPLOC システムオプションによって指定されるパスに配置する必要があります。デフォルトは**/help/helpnav.hlp/config.txt** です。

インデックスファイルに必要な形式については、デフォルトの目次ファイルを参照してください。

/help/*accessible-TOC-filename*
UNIX または z/OS 環境で SAS オンラインヘルプとドキュメントで使用されるアクセス可能な目次ファイルの部分パス名を指定します。アクセス可能な目次ファイルは、Web ブラウザで使用可能な HTML ファイルです。*accessible-TOC-filename* のファイル拡張子は.htm にする必要があり、HELPLOC システムオプションで指定されるパスに配置する必要があります。デフォルトのパス名は**/help/common.hlp/toc.htm** です。

目次ファイルに必要な形式については、デフォルトの目次ファイルを参照してください。

HTML-Help-TOC-pathname
Windows 環境で SAS オンラインヘルプとドキュメントで使用される Microsoft HTML Help の目次の完全パス名を指定します。デフォルトのパス名は**common.hhc** です。Microsoft HTML Help のインデックス作成に関する詳細については、Microsoft HTML Help ドキュメントを参照してください。

詳細

SAS によって提供される目次のかわりに使用したいカスタマイズ済みの目次がある場合は、HELPTOC オプションを使用してください。複数のオペレーティングシステムで SAS を起動する 1 つの構成ファイルがある場合は、すべての部分パス名を

HELPTOC オプションで指定できます。各型のパス名は 1 つしか指定できませんが、パス名の順序は重要ではありません。

HELPTOC オプションで UNIX または z/OS 動作環境のパス名を指定する場合、部分パス名内の**/help/**を HELPLOC オプションで指定したパス名に置き換えることで、SAS によって完全パスが決定されます。HELPLOC オプションに複数のパス名が含まれる場合、SAS によって目次の各パスが検索されます。

たとえば、HELPTOC の値が**/help/common.hlp/mytoc.htm** で、HELPLOC の値が**/u/myhome/myhelp** の場合、目次の完全パスは**/u/myhome/myhelp/common.hlp/mytoc.htm** となります。

関連項目:

システムオプション:

- “HELPLOC システムオプション: UNIX” (388 ページ)

HOSTINFOLONG システムオプション: UNIX

SAS 開始時に、追加の動作環境情報を SAS ログに書き込む指定です。

該当要素: 構成ファイル、SAS 起動

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

注: このオプションは、サイト管理者は制限できます。詳細については、“Restricted Options” (*SAS System Options: Reference* 1 章)を参照してください。

構文

HOSTINFOLONG | **NOHOSTINFOLONG**

構文説明

HOSTINFOLONG
SAS 開始時に、追加の動作環境情報を SAS ログに書き込む指定です。

NOHOSTINFOLONG
SAS 開始時に、追加の動作環境情報を SAS ログに書き込まない指定です。

詳細

HOSTINFOLONG を指定する場合、SAS は追加の動作環境情報を SAS ログに書き込みます。

関連項目:

システムオプション:

- “CPUID System Option” (*SAS System Options: Reference*)

その他の参照資料:

- “Customizing the Log” (*SAS Language Reference: Concepts* 9 章)

INSERT システムオプション: UNIX

SAS の起動時に使用され、指定システムオプションの先頭に指定値を挿入します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: なし

UNIX 固有: 構成ファイル、SAS 起動構文

注: このオプションは、サイト管理者では制限できません。詳細については、“Restricted Options” (*SAS System Options: Reference* 1 章)を参照してください。

参照項目: “INSERT= System Option” (*SAS System Options: Reference*)

構文

-INSERT *system-option new-option-value*

必須引数

system-option
FMTSEARCH、HELPLOC、MAPS、MSG、SASAUTOS、SASHELP、SASSCRIPT、SET、AUTOEXEC、CMPLIB のいずれかになります。

new-option-value
system-option の現在の値の先頭に挿入したい新しい値です。

詳細

デフォルトでは、AUTOEXEC、CMPLIB、FMTSEARCH、HELPLOC、MAPS、MSG、SASAUTOS、SASHELP、SASSCRIPT、SET システムオプションのいずれかを 2 回以上指定する場合、最後に指定した値が SAS で使用される値になります。これらののオプションのいずれかによってすでに指定されたパス名に追加パス名を付加したい場合、追加パス名の付加には INSERT システムオプションを使用する必要があります。たとえば、次の SAS コマンドを入力する場合、SAS によってヘルプファイルが検索される場所は**/apps/help** のみとなります。PROC OPTIONS の出力では、**/apps/help** のみが表示されます。

```
sas -helploc /apps/help
```

SAS でヘルプファイルの現在のパスと**/sas/help** の両方が参照されるようにして、最初に SAS で参照されるのは**/apps/help** にしたい場合、INSERT オプションを使用する必要があります。

```
sas -insert helploc /apps/help
```

ヘルプファイルの現在のパスが**!SASROOT/X11/native_help** の場合、PROC OPTIONS によって HELPLOC オプションの値に関して次が表示されます。

```
('/apps/help' '!SASROOT/X11/native_help')
```

関連項目:

システムオプション:

- “APPEND システムオプション: UNIX” (365 ページ)
- “APPEND= System Option” (*SAS System Options: Reference*)

JREOPTIONS システムオプション: UNIX

SAS の JRE (Java Runtime Environment)オプションを識別します。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 環境コントロール: 初期化および処理

PROC OPTIONS GROUP= EXECMODES

デフォルト: なし

UNIX 固有: すべて

注意: **SAS に影響する Java オプションの変更は、SAS が動作しなくなる原因になる可能性があります。** JREOPTIONS オプションの設定を変更する前に、変更予定の Java 設定が、SAS が動作しなくなる原因にならないことを SAS テクニカルサポートに問い合わせて確認してください。ベストプラクティスは、ユーザーの Java プロパティのみを変更することです。

構文

-JREOPTIONS (-*JRE-option-1* <-*JRE-option-n*>)

必須引数

-JRE-option

1 つ以上の JRE オプションを指定します。

JRE オプションの先頭はハイフン(-)にする必要があります。複数の JRE オプションを区別するには、スペースを使用します。*JRE-option* の有効な値は、ユーザーの JRE インストール環境によって異なります。JRE オプションに関する詳細については、ユーザーのインストール環境のマニュアルを参照してください。

詳細

JRE オプションは、かっこで囲む必要があります。JREOPTIONS オプションをコマンド行で実行する場合、開始かっこと閉じかっこの前にバックスラッシュ(\)を置く必要があります(下の例参照)。複数の JREOPTIONS オプションを指定する場合、現在定義されている JRE オプションに JRE オプションが SAS によって付加されます。JRE オプションの指定が間違っている場合は無視されます。

例: JRE オプションの使用

```
-jreoptions \(-Dmy.java.property\)

-jreoptions \(-Xmx512m -Xms256m\)
```

LINESIZE システムオプション: UNIX

SAS の **Log** および **Output** ウィンドウの行サイズを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログおよびプロシジャ出力

PROC OPTIONS GROUP= LOG_LISTCONTROL、LOGCONTROL

デフォルト: 対話型モードの場合は表示幅設定、バッチモードの場合は 132

UNIX 固有: デフォルト値

参照項目: “LINESIZE= System Option” (*SAS System Options: Reference*)

構文

-LINESIZE *n* | *hex*X | MIN | MAX

LINESIZE=*n* | *hex*X | MIN | MAX

必須引数

n
行サイズを文字数で指定します。有効な値の範囲は 64 - 256 です。

***hex*X**
行サイズを 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、`2dx` の場合は 45 文字が指定されます。

MIN
行サイズを 64 文字に設定します。

MAX
行サイズを 256 文字に設定します。

関連項目:

“UNIX 環境で出力のコンテンツと表示画面を制御する” (102 ページ)

LOG システムオプション: UNIX

バッチモードで実行時の SAS ログの出力先を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES、LOGCONTROL

デフォルト: カレントディレクトリにある、SAS ソースファイルと同じファイル名で拡張子が.log のファイル

UNIX 固有: すべて

構文

-LOG *file-specification* | **-NOLOG**

必須引数

-LOG *file-specification*
SAS ログの出力先を指定します。*file-specification* は、ディレクトリ、ファイル名またはパスに関連付けられた環境変数への任意の有効な UNIX パスのいずれかとなります。ディレクトリへのパスのみを指定する場合、ログファイルは指定ディレクトリ内に作成されます。このファイルのデフォルト名は *filename*.log となり、*filename* は SAS ジョブ名を表します。

-NOLOG
SAS ログの作成を抑制します。SAS プログラムを完全にデバッグしないかぎり、この値は使用しないでください。

詳細

LOG システムオプションは、バッチモードで実行時の SAS ログの出力先を指定します。LOG システムオプションはバッチモードで有効ですが、対話型モードでは無視されます。

時間、月および曜日などのリアルタイムイベントに基づき、ログを開く/閉じるタイミングやその命名法を調整するには、LOG システムオプションでディレクティブを使用します。有効なディレクティブのリストについては、“LOGPARM= System Option” (*SAS System Options: Reference*)を参照してください。

SAS をバッチモードまたはサーバーモードで起動し、LOGCONFIG LOC=オプションを指定する場合、ログ記録は SAS ログ機能によって実行されます。従来の SAS ログオプション LOGPARM=は無視されます。従来の SAS ログオプション LOG=が尊重されるのは、ログ側の構成ファイルで%S{App.Log}変換文字を指定する場合に限られます。詳細については、*SAS ログ機能: 構成とプログラミングリファレンス*の SAS ログ機能を参照してください。

注: OBJECTSERVER および NOTERMINAL システムオプションを指定して SAS を起動し、ログを指定しない場合、SAS によってすべてのログと代替ログメッセージが廃棄されます。

関連項目:

システムオプション:

- “LOGPARM= System Option” (*SAS System Options: Reference*)

その他の参照資料:

- “The SAS Log” (*SAS Language Reference: Concepts* 9 章)
- “SAS システムオプションを使用し、出力先を指定する” (100 ページ)

LPTYPE システムオプション: UNIX

プリンタへのファイルのルーティングに使用する UNIX コマンドとオプション設定の種類を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: プロシジャ出力

PROC OPTIONS GROUP= LISTCONTROL

デフォルト: なし

UNIX 固有: すべて

構文

-LPTYPE BSD | SYSV

LPTYPE=BSD | SYSV

必須引数

-LPTYPE BSD
ファイルをプリンタに送信する際に、SAS で `lpr` コマンドが使用されるようにします。通常、`lpr` コマンドは、HP-UX (カリフォルニア大学バークレイ校で開発)などの UNIX オペレーティングシステムでサポートされます。

-LPTYPE SYSV
ファイルをプリンタに送信する際に、SAS で `lp` コマンドが使用されるようにします。通常、`lp` コマンドは、Solaris などの UNIX システム V から派生したオペレーティングシステムでサポートされます。

詳細

LPTYPE オプションによって、SAS でファイルの印刷に `lpr` または `lp` UNIX コマンドを使用するかどうかが決定されます。

BSD または SYSV を指定するかどうかが分からな場合は、システム管理者にお問い合わせください。

デフォルトでは、ご使用のオペレーティングシステムがバークレイのバージョンから派生したものである場合は、SAS で `lpr` コマンドが使用されます。それ以外の場合は、`lp` コマンドが使用されます。

関連項目:

システムオプション:

- “PRINTCMD システムオプション: UNIX” (408 ページ)

MAPS システムオプション: UNIX

SAS/GRAPH マップデータセットが格納される SAS ライブラリの名前を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: グラフィック: ドライバ設定

PROC OPTIONS GROUP= GRAPHICS

デフォルト: !SASROOT/maps (インストール済みの!SASROOT/sasv9.cfg ファイルで設定)

UNIX 固有: デフォルト値と *location-of-maps*

参照項目: “MAPS= System Option” (*SAS/GRAPH: Reference*)

構文

-MAPS *location-of-maps*

MAPS=*location-of-maps*

必須引数

location-of-maps
ライブラリ参照名、有効な UNIX パス名またはパス名に関連付けられた環境変数のいずれかを指定します。特定のファイル名を使用しないでください。

詳細

基本

MAPS ライブラリ参照名は再割当することはできますが、これを消去することはできません。

地図ファイルは、使用する前に解凍する必要がある場合があります。圧縮するかどうかを決定するには、DATASETS プロシジャで CONTENTS ステートメントを使用します。

パス名の挿入および付加

デフォルトでは、MAPS システムオプションを 2 回以上指定する場合、最後に指定したオプションが SAS で使用されるオプション値になります。

MAPS システムオプションによってすでに指定されたパス名に追加パス名を付加したい場合、追加パス名の付加には INSERT システムオプションを使用する必要があります。たとえば、次の SAS コマンドを入力する場合、SAS によってヘルプファイルが検索される場所は**/apps/help** のみとなります。PROC OPTIONS の出力では、**/apps/help** のみが表示されます。

```
sas -helploc /apps/help
```

SAS でヘルプファイルの現在のパスと**/sas/help** の両方が参照されるようにして、最初に SAS で参照されるのは**/apps/help** にしたい場合、INSERT オプションを使用する必要があります。

```
sas -insert helploc /apps/help
```

SAS で最初に**/sas/help**、次いで**/apps/help** の順に参照するようにしたい場合は、APPEND を使用する必要があります。

```
sas -helploc /sas/help -append helploc /apps/help
```

ヘルプファイルの現在のパスが**!SASROOT/X11/native_help** の場合、PROC OPTIONS によって HELPLOC オプションの値に関して次が表示されます。

```
('/apps/help' '!SASROOT/X11/native_help')
```

関連項目:

システムオプション:

- “APPEND= System Option” (*SAS System Options: Reference*)
- “INSERT= System Option” (*SAS System Options: Reference*)

MAXMEMQUERY システムオプション: UNIX

特定のプロシジャへの要求ごとに割り当てることができる最大メモリ容量を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: システム管理: メモリ

PROC OPTIONS GROUP= MEMORY

デフォルト: 256M

UNIX 固有: すべて

構文

-MAXMEMQUERY *n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

MAXMEMQUERY=*n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
制限を 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、値が `8` の場合は 8 バイト、値が `.782k` の場合は 801 バイト、値が `3m` の場合は 3,145,728 バイトが指定されます。

***hex*X**
メモリ容量を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、`2dx` の場合はメモリ容量が 45 バイトに設定されます。

MIN
0 バイトを指定します。これは、各 SAS プロシジャによる要求ごとに割当可能な合計メモリ容量に制限がないことを意味します。このようなメモリ割当は、MEMSIZE の値によって制限されます。

MAX
割り当てられるメモリ容量への制限を指定します。メモリ割当(64 ビットマシンの場合は 9,007,199,254,740,992 バイト)は、MEMSIZE の値によって制限されます。

詳細

一部の SAS プロシジャでは、1 つのプロシジャが 1 回に要求可能な仮想メモリの最大ブロック数を指定するに当たり、MAXMEMQUERY オプションが使用されます。それとは対照的に、MEMSIZE オプションは、SAS で常に動的に割り当てられる仮想メモリの合計容量に制限を加えます。こうした仮想メモリは、実メモリとページング領域を組み合わせてサポートされます。動作環境では、必要な仮想メモリの容量が使用可能な実メモリ容量を超過すると、ページングが開始されます。ページングや関連パフォーマンスの問題が発生するのを防ぐには、MAXMEMQUERY および MEMSIZE システムオプションを実メモリのサブセットに設定する必要があります。

MEMSIZE システムオプション: UNIX

SAS セッションによって使用可能な仮想メモリの合計容量に関する制限を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: システム管理: メモリ

PROC OPTIONS GROUP= MEMORY、PERFORMANCE

デフォルト: 2G

UNIX 固有: すべて

構文

-MEMSIZE *n* | *n*K | *n*M | *n*G | *n*T | *hex*X | MAX

必須引数

***n* | *n*K | *n*M | *n*G | *n*T**
制限を 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 1,099,511,627,776 (ギガバイト)、1,099,511,627,776 (テラバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます(たとえば、25G は 268,435,456 バイトと指定します)。

***hex*X**
メモリ容量を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、`0F00000x` の場合は MEMSIZE オプションの値が 15,728,640 バイトに設定されます。値が `0x` の場合、MAX 値を使用することと同義になります。

MAX
SAS の開始時に使用可能な物理メモリとページング領域の容量に応じて、最大限の妥当な値にメモリサイズが設定されるための指定です。

詳細

基本

MEMSIZE システムオプションは、各 SAS セッションに使用可能な合計メモリ容量を制限します。また、実行時に SAS で動的に割当可能な仮想メモリの容量に強制的な制限が加えられます。MEMSIZE を低すぎる値に設定すると、ジョブでエラーが発生し、使用可能なメモリ容量が不足していることを示すエラーが SAS ログに現れます。それとは対照的に、REALMEMSIZE および MAXMEMQUERY システムオプション、SORT プロシジャでの SORTSIZE=オプション、SUMMARY プロシジャでの SUMSIZE=オプションはすべて、プロシジャの調整に使用されます。

SAS セッションの開始時に、MEMSIZE の値が、プロセスに使用可能な仮想メモリの容量よりも大きい場合、SAS ログに通知されます。この場合は、MEMSIZE の値が仮想メモリの容量より小さくなるように調整するか、MAX 値を使用します。MAX 値によって、ページサイズとメモリ制限の両方が自動的に考慮され、それに従って MEMSIZE 値が調整されます。`limit`、`ulimit -a`、または `ulimit -aS` コマンドを使用すると、各ユーザー ID で使用可能な仮想メモリの容量を参照できます。

MEMSIZE に著しく小さい値(6K など)を指定すると、SAS を開始できる最小値まで MEMSIZE 値が自動的に増加されます。

数値が 9,223,372,036,854,775,807 バイトを超える場合、無効として拒否され、SAS は開始できません。

SAS では、ユーザーが MEMSIZE システムオプションで指定する仮想メモリ容量を自動的に確保したり割り当てたりすることはありません。SAS で使用されるのは、プロセスの完了に必要なメモリ容量に限られます。たとえば、DATA ステップで要求されるメモリ容量が 20 MB のみである場合、MEMSIZE が 500 MB に設定されていても、SAS は 20 MB のメモリ容量しか使用しません。SAS ジョブの実行中に大容量のメモリ設定による影響をモニタするには、VMSTAT や top ツールなどのシステムモニタリングツールを使用します。一部のツールでは、アドレス領域がメモリに割り当てられる場合がありますが、そのメモリにページを割り当てることはできません。こうしたツールを使用すると、実際に使用された、メモリ容量を超える値がレポートされます。ユーザーがサードパーティソフトウェアを起動した場合(SAS がロードするデータベースベンダコードなど)、そのサードパーティソフトウェアに対するメモリ割り当ては MEMSIZE では制御されません。サードパーティソフトウェアのメモリ使用量は、top ツールによってレポートできます。

MEMSIZE のサイズ設定

MEMSIZE=MAX を設定すると、MEMSIZE が物理メモリの 80%に設定されます。MEMSIZE を MAX に設定するのが合理的なのは、大容量のメモリを消費するプロセスが、SAS の起動後にアクティブになる可能性がない場合に限られます。たとえば、SAS の複数のインスタンスが同時に実行中で、MEMSIZE 値を MAX に設定した状態ですべてのセッションが起動された場合、当該セッションの 1 つ以上でメモリ不足状態が発生するか、またはオペレーティングシステムで利用可能なページング領域が不足する可能性があります。MEMSIZE=MAX によって、すべてのメモリが割当済みである場合にシステムでページングが行われないようにする値が計算されます。

このオプションの最適な設定は、実行中の他のアプリケーションやサイトで使用可能なシステムリソースに応じて異なります。また、SAS プロセスに使用可能なメモリ容量は、システム管理者が制限することもできます。

MEMSIZE を合理的に達成可能な最大限のメモリ容量に設定する場合、一部のプロシジャが自動的に使用可能なメモリに拡張されます。SAS で使用されるメモリの合計量に関する制限を決めるためには、PROC OPTIONS ステートメントを実行します。

```
proc options option=memsize;
run;
```

MEMSIZE を 0 に設定する場合、これをテストとして使用し、MEMSIZE に適切な値を決めることができます。

MEMSIZE の最適な設定を決定するには、FULLSTIMER オプションと MEMSIZE を 0 に設定した状態で SAS プロシジャまたは DATA ステップを実行します。プロセスによって使用されるメモリ容量に注意し、次に MEMSIZE を大容量に設定します。

比較

一部の SAS プロシジャでは、過度のページ交換を誘発せずにプロシジャが割り当てて使用できる実メモリ容量を指定するに当たり、REALMEMSIZE システムオプションが使用されます。それとは対照的に、MEMSIZE オプションは、SAS で常に動的に割り当てられる仮想メモリの合計容量に制限を加えます。こうした仮想メモリは、実メモリとページング領域を組み合わせてサポートされます。

動作環境では、必要な仮想メモリの容量が使用可能な実メモリ容量を超過すると、ページングが開始されます。ページングや関連パフォーマンスの問題が発生するのを防ぐには、REALMEMSIZE および MEMSIZE システムオプションを実メモリのサブセットに設定する必要があります。

関連項目:

システムオプション:

- “REALMEMSIZE システムオプション: UNIX” (409 ページ)

プロシジャ:

- “SORT プロシジャ: UNIX” (317 ページ)

MSG システムオプション: UNIX

SAS エラーメッセージを格納するライブラリを指定します。

該当要素:	構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数
カテゴリ:	環境コントロール: ファイル
PROC OPTIONS GROUP=	ENVFILES
別名:	SASMSG
デフォルト:	!SASROOT/maps (インストール済みの!SASROOT/sasv9.cfg ファイルで設定)

構文

-MSG *pathname*

-MSG ('*pathname*' '*pathname*' ...)

必須引数

pathname
有効な UNIX パス名に対して解決される必要があります。有効なパス名に対して解決される環境変数を使用できます。

詳細

MSG システムオプションは、SAS エラーメッセージを格納するライブラリを指定します。このオプションはインストールプロセス中に設定されますが、通常はインストール後に変更されます。

追加パス名を付加するには、INSERT または APPEND システムオプションを使用します。詳細については、“INSERT システムオプション: UNIX” (391 ページ)および“APPEND システムオプション: UNIX” (365 ページ)を参照してください。

関連項目:

システムオプション:

- “APPEND= System Option” (*SAS System Options: Reference*)
- “INSERT= System Option” (*SAS System Options: Reference*)

MSGCASE システムオプション: UNIX

SAS によって生成されるメモ、警告およびエラーメッセージが大文字で表示されるように指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: NOMSGCASE

UNIX 固有: すべて

構文

-MSGCASE | **-NOMSGCASE**

必須引数

-MSGCASE
NOTE、警告およびエラーメッセージを大文字で表示します。

-NOMSGCASE
NOTE、警告およびエラーメッセージを大文字と小文字で表示します。

詳細

MSGCASE システムオプションでは、SAS によって生成されるメモ、警告およびエラーメッセージが大文字で表示されるかどうかが指定されます。ユーザー生成メッセージとソースラインは、MSGCASE システムオプションによって影響を受けません。

MSGCASE は NL 形式でサポートされます。NL 形式に関する詳細については、*SAS 各国語サポート(NLS): リファレンスガイド*を参照してください。

MSYMTABMAX システムオプション: UNIX

マクロ変数シンボルテーブルに使用可能な最大メモリ容量を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: マクロ: SAS マクロ

PROC OPTIONS GROUP= MACRO

デフォルト: 4M (インストール済みの!SASROOT/sasv9.cfg ファイルで設定)

UNIX 固有: デフォルト値

参照項目: MSYMTABMAX=システムオプション: *SAS マクロ言語: リファレンス*

構文

-MSYMTABMAX *n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

MSYMTABMAX=*n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
最大メモリ容量を 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、値が 8 の場合は 8 バイト、値が.782k の場合は 801 バイト、値が 3m の場合は 3,145,728 バイトが指定されます。

***hex*X**
使用可能な最大メモリ容量を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、`2dx` の場合は最大メモリ容量が 45 バイトに設定されます。

MIN
使用可能なメモリ容量を最小設定値(0 バイト)に設定します。メモリ容量を最小設定値に設定すると、すべてのマクロシンボルテーブルがディスクに書き込まれます。

MAX
使用可能なメモリ容量を最小設定値に設定します。64 ビットコンピュータの場合、この値は 9,007,199,254,740,992 バイトです。

MVARSIZE システムオプション: UNIX

メモリ内マクロ変数の最大サイズを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: マクロ: SAS マクロ

PROC OPTIONS GROUP= MACRO

デフォルト: 32K (インストール済みの!SASROOT/sasv9.cfg ファイルで設定)

UNIX 固有: デフォルト値

参照項目: MVARSIZE システムオプション: *SAS マクロ言語: リファレンス*

構文

-MVARSIZE *n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

MVARSIZE=*n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
マクロ変数の最大サイズを 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、値が `8` の場合は 8 バイト、値が `.782k` の場合は 801 バイト、値が `3m` の場合は 3,145,728 バイトが指定されます。

***hex*X**
マクロ変数の最大サイズを 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、**2dx** の場合はマクロ変数の最大サイズが 45 バイトに設定されます。

MIN
マクロ変数サイズを最小設定値(0 バイト)に設定します。マクロ変数サイズを最小設定値に設定すると、すべてのマクロシンボルテーブルがディスクに書き込まれます。

MAX
マクロ変数サイズを最大設定値(65,534 バイト)に設定します。

NEWS システムオプション: UNIX

SAS ログに書き込まれるメッセージを格納するファイルを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES、LOGCONTROL

デフォルト: !SASROOT/misc/base/news (インストール済みの!SASROOT/sasv9/cfg ファイルで設定)

UNIX 固有: -NONEWS オプション

参照項目: “NEWS= System Option” (*SAS System Options: Reference*)

構文

-NEWS *file-specification* | **-NONEWS**

必須引数

-NEWS *file-specification*
外部ファイルを指定します。このファイルには、SAS ログ用メッセージが格納されます。

-NONEWS
NEWS ファイルが存在する場合でも、そのコンテンツが SAS ログに表示されないよう指定します。このオプションによって、前の NEWS 指定が無視されます。

詳細

NEWS ファイルのコンテンツは、SAS ログの SAS ヘッダ直後に表示されます。

関連項目:

“The SAS Log” (*SAS Language Reference: Concepts* 9 章)

OBS システムオプション: UNIX

データセット内で SAS が最後に処理するオブザベーションを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: ファイル: SAS ファイル

PROC OPTIONS GROUP= SASFILES

デフォルト: MAX

UNIX 固有: デフォルト値

参照項目: “OBS= System Option” (*SAS System Options: Reference*)

構文

-OBS *n* | *n*K | *n*M | *n*G | *n*T | *hex*X | MIN | MAX

OBS=*n* | *n*K | *n*M | *n*G | *n*T | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G | *n*T**
処理を停止するタイミングを示す数を指定します。いずれかの文字表記法を使用すると、整数 × 指定値の結果になります。つまり、K (キロ)を指定すると整数 × 1,024、M (メガ)は整数 × 1,048,576、G (ギガ)は整数 × 1,073,741,824、T (テラ)は整数 × 1,099,511,627,776 となります。K、M、G、T の値のいずれかの指定に使用する場合、*n* の小数値を指定できます。たとえば、オブザベーションまたはレコードの数は、値が `20` の場合は 20、値が `.782k` の場合は 801、値が `3m` の場合は 3,145,728 に指定されます。

***hex*X**
処理を停止するタイミングを示す数を 16 進数として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、16 進数値 F8 の場合は、248 に相当する 10 進数を指定するために、`0F8x` として指定する必要があります。たとえば、値が `2dx` の場合、45 に相当する 10 進数が指定されます。

MIN
処理を停止するタイミングを示す数字を 0 に設定します。

OBS=0 および NOREPLACE オプションが有効な場合、SAS でまだ特定のアクションを実行できる可能性があります。詳細については、“OBS= System Option” (*SAS System Options: Reference*)を参照してください。

MAX
処理を停止するタイミングを示す数字を 9,223,372,036,854,775,807 に設定します。

OPLIST システムオプション: UNIX

SAS システムオプションの設定を SAS ログに書き込むかどうかを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: NOOPLIST

UNIX 固有: すべて

構文

-OPLIST | -NOOPLIST

詳細

OPLIST システムオプションは、コマンド行で指定したシステムオプションのみをエコーします。構成ファイルや SASV9_OPTIONS 環境変数で指定したシステムオプションはエコーしません。(構成ファイルのコンテンツをエコーしたい場合は、VERBOSE オプションを使用してください) たとえば、SAS を次のコマンドで起動します。

```
sas -nodms -fullstimer -nonews -oplist
```

SAS では次のラインが SAS ログに書き込まれます。

```
NOTE: SAS command line: -nodms -fullstimer -nonews -oplist
```

システムオプション(EMAILPW、METAPASS、PDFOPENPW など)に対して指定されるパスワード値は、SAS ログでは自動的にマスクされます。たとえば、次のコマンドを使用して SAS を起動するとします。

```
sas -nodms -oplist -emailpw foo -metapass xyz -pdfopenpw foobar -stimer
```

コマンドは、ログで次のように表示されます。

```
NOTE: SAS command line:
/tdi/mva-v940m2/usrlibsas/laxno.14w28.20140430.weekly/SASFoundation/9.4/
sasexe/sas -nodms -oplist -emailpw XXXXXXXX -metapass XXXXXXXX
-pdfopenpw XXXXXXXX -stimer -helphost <hostname>
```

関連項目:

システムオプション:

- “VERBOSE システムオプション: UNIX” (433 ページ)

PAGESIZE システムオプション: UNIX

SAS 出力のページを構成する行数を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログおよびプロシジャ出力

PROC OPTIONS GROUP= LOG_LISTCONTROL、LOGCONTROL

デフォルト: 対話型モードの場合は表示される行数、バッチモードの場合は 60

UNIX 固有: デフォルト値および範囲

参照項目: “PAGESIZE= System Option” (*SAS System Options: Reference*)

構文

-PAGESIZE *n* | *n*K | *hex*X | MIN | MAX

PAGESIZE=*n* | *n*K | *hex*X | MIN | MAX

必須引数

***n* | *n*K**
ページを構成する行数を 1 (*n*)または 1,024 (*n*K)の倍数で指定します。キロバイトの数には小数値を指定できます。たとえば、値が `800` の場合は 800 行、値が `.782k` の場合は 801 行、値が `3m` の場合は 3,072 行が指定されます。

***hex*X**
ページを構成する行数を 16 進数の値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、値が `2dx` の場合は 45 行が指定されます。

MIN
ページを構成する行数を最小設定値(15)に設定します。

MAX
ページを構成する行数を最大設定値(32,767)に設定します。

詳細

対話型モードの場合のデフォルト値は、表示される行数です。バッチモードでは 60 がデフォルトになります。

関連項目:

- “UNIX 環境で出力のコンテンツと表示画面を制御する” (102 ページ)
- “The SAS Log” (*SAS Language Reference: Concepts* 9 章)

PATH システムオプション: UNIX

SAS 実行可能ファイルの 1 つ以上の検索パスを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: !SASROOT/sasexe (インストール済みの!SASROOT/sasv9/cfg ファイルで設定)

UNIX 固有: すべて

構文

-PATH *directory-specification*

必須引数

directory-specification
SAS 実行可能ファイルの検索パスを指定します。

詳細

PATH システムオプションは、SAS 実行可能ファイルの検索パスを識別します。検索順序を定義するには、複数の PATH オプションを指定します。パスの検索順序は、SAS での検出順になります。そのため、最も使用頻度の高い製品のパスをリストの最初に指定してください。PATH システムオプションを 2 回以上指定する際にその順序がどのように決まるかに関する詳細については、“複数の場所で指定されているシステムオプションの処理方法” (21 ページ)を参照してください。

PRIMARYPROVIDERDOMAIN システムオプション: UNIX

主要認証プロバイダのドメイン名を指定します。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 環境コントロール: 初期化および処理

PROC OPTIONS GROUP= EXECMODES

別名: PRIMPD

参照項目: “PRIMARYPROVIDERDOMAIN= System Option” (*SAS System Options: Reference*)

PRINT システムオプション: UNIX

バッチモードで実行時の SAS 出力の出力先を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: バッチ SAS プログラムからの SAS 出力は、カレントディレクトリにある、SAS ソースファイルと同じファイル名で拡張子が.lst のファイルに書き込まれます

UNIX 固有: すべて

構文

-PRINT *file-specification* | **-NOPRINT**

必須引数

-PRINT file-specification
SAS プロシジャ出力ファイルの場所を指定します。*file-specification* は、ディレクトリ、ファイル名またはパスに関連付けられた環境変数への任意の有効な UNIX パスのいずれかとなります。ディレクトリへのパスのみを指定する場合、プロシジャ出力ファイルは指定ディレクトリ内に作成されます。このファイルのデフォルト名は *filename*.lst となり、*filename* は SAS ジョブ名を表します。

-NOPRINT
SAS プロシジャ出力ファイルの作成を抑制します。

詳細

PRINT システムオプションは、バッチモードで実行時の SAS 出力の出力先を指定します。PRINT システムオプションはバッチモードで有効ですが、対話型モードでは無視されます。

関連項目:

"SAS システムオプションを使用し、出力先を指定する" (100 ページ)

PRINTCMD システムオプション: UNIX

出力コマンド SAS を使用するように指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数
カテゴリ:	ログおよびプロシジャ出力コントロール: プロシジャ出力
PROC OPTIONS GROUP=	LISTCONTROL
デフォルト:	なし
UNIX 固有:	すべて

構文

-PRINTCMD "*print-command*"

PRINTCMD="*print-command*"

必須引数

print-command
PRINTCMD で使用できるオプションを指定します。

詳細

出力コマンドに受け渡されるオプションの構文は、LPTYPE システムオプションによって制御されます。LPTYPE を BSD に設定する場合、コマンドでは `lpr` コマンドオプションが使用されます。LPTYPE を SYSV に設定する場合、コマンドでは `lp` コマンドオプションが使用されます。

サイトで `lp` または `lpr` 以外の出力コマンド(スプーラ)を使用する場合、*print-command* によってその名前が指定されます。PRINTCMD オプションによって、LPTYPE 設定が上書きされます。

OPTIONS ステートメントで指定する場合、以前に定義されたファイル名に割り当て済みの出力コマンドは PRINTCMD オプションでは変更されません。たとえば、次のコードを検討します。

```
filename pc1 printer;
proc printto print=pc1;
run;
proc print data=sales.week;
run;
```

```
options printcmd="netlp";

filename pc2 printer;
proc printto print=pc2;
run;
proc print data=sales.month;
run;
```

PC2 に関連付けられた出力では、**netlp** コマンドが使用され、PC1 に関連付けられた出力では、デフォルトの出力コマンドが使用されます。

関連項目:

システムオプション:

- “LPTYPE システムオプション: UNIX” (394 ページ)

その他の参照資料:

- “UNIX 環境における出力印刷の概要” (90 ページ)

REALMEMSIZE システムオプション: UNIX

SAS で割当が期待できる実際の(物理)メモリの容量を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: システム管理: メモリ

PROC OPTIONS GROUP= MEMORY

デフォルト: 0

UNIX 固有: 有効な値

構文

-REALMEMSIZE *n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
確保するメモリ容量を 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。値が *n* の場合、小数値となります。たとえば、値が **8** の場合は 8 バイト、値が **.782k** の場合は 801 バイト、値が **3m** の場合は 3,145,728 バイトが指定されます。

***hex*X**
メモリ容量を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、値が **2dx** の場合はメモリ容量が 45 バイトに設定されます。

MIN
値を 0 に指定します(SAS の起動時にメモリ使用量が SAS によって決定されることを示します)。

MAX
メモリサイズを許容可能な最大値に設定するよう指定します。この値はシステムの制限に応じて異なります。

詳細

基本

REALMEMSIZE システムオプションは、実メモリとユーティリティディスク領域の両方を使用可能なプロシジャ(PROC SUMMARY や PROC SORT など)の実メモリに推奨上限値を設定します。この上限値は、仮想メモリのスラッシング回避に役立ちます。

REALMEMSIZE オプションは、絶対に実メモリ容量を超えた値に設定しないでください。実メモリ容量がジョブの実行に必要な量を満たしている場合、MEMSIZE オプションを実メモリ容量を超える値に設定すると、実メモリと仮想メモリを組み合わせてジョブを実行できます。

比較

一部の SAS プロシジャでは、過度のページ交換を誘発せずにプロシジャが割り当てて使用できる実メモリ容量を指定するに当たり、REALMEMSIZE システムオプションが使用されます。それとは対照的に、MEMSIZE オプションは、SAS で常に動的に割り当てられる仮想メモリの合計容量に制限を加えます。こうした仮想メモリは、実メモリとページング領域を組み合わせてサポートされます。

動作環境では、必要な仮想メモリの容量が使用可能な実メモリ容量を超過すると、ページングが開始されます。ページングや関連パフォーマンスの問題が発生するのを防ぐには、REALMEMSIZE および MEMSIZE システムオプションを実メモリのサブセットに設定する必要があります。

関連項目:

システムオプション:

- “MEMSIZE システムオプション: UNIX” (398 ページ)

プロシジャ:

- “SORT プロシジャ: UNIX” (317 ページ)

RSASUSER システムオプション: UNIX

Sasuser ライブラリのメンバが更新または読み取り専用アクセス用に開くことができるかどうかを制御します。

該当要素:	構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数
カテゴリ:	環境コントロール: ファイル
PROC OPTIONS GROUP=	ENVFILES
デフォルト:	NORSASUSER
UNIX 固有:	ネットワークの注意点
参照項目:	“RSASUSER System Option” (*SAS System Options: Reference*)

構文

-RSASUSER | -NORSASUSER

必須引数

-RSASUSER
Sasuser ライブラリへのアクセスを読み取り専用に制限します。

-NORSASUSER
ユーザーは更新アクセス用に Sasuser ライブラリ内のファイルを開くことができるため、ユーザーが Sasuser ライブラリのメンバを共有できないようにします。更新アクセスには、ライブラリメンバへの排他的権限が必要です。

詳細

Sasuser ライブラリが複数のユーザーによって共有されているか、または同一ユーザーが SAS を同時に複数回実行している場合、通常、Sasuser ライブラリは共有されます。デフォルトでは、あるユーザーが Sasuser ライブラリのメンバを更新のために開いている場合、他のすべてのユーザーはその SAS ライブラリメンバへのアクセスが拒否されます。たとえば、あるユーザーが Sasuser.Profile カタログに書き込んでいる場合、他のユーザーは Profile カタログから読み取ることさえできません。

RSASUSER を指定すると、すべてのユーザーにメンバへの読み取り専用アクセスを許可することで、ユーザーグループは Sasuser ライブラリを共有できます。Profile カタログの例では、RSASUSER が有効な場合、すべてのユーザーは読み取り専用アクセス用に Profile カタログを開くことができ、他のユーザーは Profile カタログから同時に読み取ることができます。ただし、どのユーザーも Profile カタログに書き込むことはできません。書き込もうとしても、エラーメッセージが表示されます。

コマンド行から RSASUSER を指定すると、そのセッションのファイルへのアクセスのみが影響を受けます。ユーザーグループが Sasuser ライブラリ内のメンバを共有できるようにするには、Sasuser ライブラリを共有している全ユーザーで共有する、共通 SAS 構成ファイルに RSASUSER をシステム管理者が設定する必要があります。

RSASUSER を指定しても Sasuser ライブラリ内に Profile カタログが存在しない場合、Profile カタログは Work ライブラリ内に作成されます。

注: RSASUSER オプションは Sasuser ライブラリ内に格納されている情報(Profile カタログなど)を共有する場合にきわめて役立ちます。SAS/ASSIST ソフトウェアや、Sasuser ライブラリへの更新アクセスを必要とする SAS モジュールと連動して使用する場合にはそれほど実用的ではありません。

関連項目:

"UNIX 環境での SAS ファイルの共有" (41 ページ)

RTRACE システムオプション: UNIX

SAS セッション中に読み取りまたは読み込みされるリソースのリストを生成します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: なし

UNIX 固有: すべて

構文

-RTRACE ALL | NONE | VER

必須引数

ALL
SAS セッション中に読み取りまたは読み込みされるリソースのリストを生成します。

NONE
すべてのファイルで RTRACE をオフにします。

VER
SAS がロードする各モジュールのバージョン番号やその他のトレース情報を書き込みます。

詳細

RTRACE システムオプションは、SAS セッション中に読み取りまたは読み込みされるリソースのリストを生成します。-RTRACE ALL を指定しても、RTRACELOC システムオプションが指定されていない場合、出力は SAS ログに書き込まれます。

関連項目:

システムオプション:

- “RTRACELOC システムオプション: UNIX” (412 ページ)

RTRACELOC システムオプション: UNIX

SAS セッション中に読み取りまたは読み込みされるリソースのリストの書き込み先となるファイルのパス名を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: なし

UNIX 固有: すべて

ヒント: %p (PID)、%d (日付)、または%t (時間)が指定された場合、RTRACELOC ファイル名を拡張できます。

構文

-RTRACELOC *pathname*

RTRACELOC=*pathname*

必須引数

pathname

RTRACE 情報の書き込み先ファイルを指定します。*pathname* には、RTRACE 出力のパスとパス名を含める必要があります。

出力ファイル名を拡張してプロセス ID、日付、または時間をファイル名に含めるには、それぞれ%p、%d、%t を指定します。システム日付は、YYYYMMDD (年、月、日)形式でファイル名に含められます。時間は、HHMMSSmmm(時、分、秒、ミリ秒)という形式でファイル名に含まれます。たとえば、日付、時間、およびプロセス ID をファイル名に含めるには、次のオプションを指定します。

```
-rtrace all -rtraceloc mytrace.%d.%t.%p
```

この結果、mytrace.20140306.125510942.3808 のようなファイル名になります。

詳細

RTRACELOC システムオプションは、RTRACE 情報の書き込み先ファイルのパス名を指定します。*pathname* にファイル名が含まれない場合、出力は標準出力に送られます。-RTRACE ALL を指定しても、RTRACELOC システムオプションが指定されていない場合、出力は SAS ログに書き込まれます。

関連項目:

システムオプション:

- “RTRACE システムオプション: UNIX” (411 ページ)

SASAUTOS システムオプション: UNIX

自動呼び出しライブラリを指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数
カテゴリ:	環境コントロール: ファイル マクロ: SAS マクロ
PROC OPTIONS GROUP=	ENVFILES MACRO
デフォルト:	SASAUTOS ファイル参照名
UNIX 固有:	次の項目を複数指定するための構文: *directory-specifications*
参照項目:	“SASAUTOS= System Option” (*SAS Macro Language: Reference*)

構文

-SASAUTOS '*directory-specification*' | *fileref*

-SASAUTOS ('*directory-specification-1*' | *fileref-1*, ..., '*directory-specification-n*' | *fileref-n*)

-NOSASAUTOS

SASAUTOS='*directory-specification*' | *fileref*

SASAUTOS =('*directory-specification-1*' | *fileref-1*, ..., '*directory-specification-n*' | *fileref-n*)

NOSASAUTOS

必須引数

directory-specification
自動呼び出しマクロライブラリにパス名を指定します。

fileref
自動呼び出しマクロライブラリに割り当てられた名前(簡易参照)を指定します。

SASAUTOS オプションでは、ライブラリ参照名ではなくファイル参照名が使用されますのでご注意ください。

詳細

各自動呼び出しマクロライブラリは、UNIX ディレクトリ内のファイルで構成されます。*directory-specification* は、UNIX ディレクトリのパス名、ファイル参照名、環境変数のいずれかになります。

ディレクトリのパス名を指定する場合、名前を引用符で囲む必要があります。引用符を省略できるのは、構成ファイル、SAS コマンド、SASV9_OPTIONS 環境変数のいずれかでオプションを指定し、名前がファイル参照名に抽出できる場合に限られます。

ファイル参照名を指定する場合、任意の自動呼び出しマクロを使用しようとする前にそれを定義する必要があります。ファイル参照名は FILENAME ステートメント、環境変数、FILENAME 関数のいずれかで定義できます。“FILENAME ステートメントを使用し、ファイル参照名を外部ファイルまたはデバイスに割り当てる” (73 ページ)を参照してください。

ディレクトリ名、ファイル参照名または環境変数のいずれかを複数指定する方法は、SASAUTOS オプションを指定する場所に応じて異なります。

- SASAUTOS を構成ファイルまたは SASV9_OPTIONS 環境変数で指定する場合、複数の SASAUTOS オプションを使用するか、またはディレクトリ名をかっこで囲みます。名前の区切りには、カンマまたは空白スペースを使用してください。
- SASAUTOS オプションを SAS コマンドで指定する場合、APPEND または INSERT システムオプションを使用して現在の SASAUTOS 値の最後に付加するか、または先頭に挿入します。たとえば、次のコードによって **/users/userid/also** が現在の SASAUTOS 値(**/users/userid/here**)の最後に追加されます。

  ```
  sas -sasautos /users/userid/here -append sasautos /users/userid/also
  ```

  詳細については、“APPEND= System Option” (*SAS System Options: Reference*) 、および “INSERT= System Option” (*SAS System Options: Reference*)を参照してください。
- SASAUTOS オプションを OPTIONS ステートメントまたは **SAS System Options** ウィンドウで指定する場合、ディレクトリ名をかっこで囲む必要があります。名前の区切りには、カンマまたは空白スペースを使用してください。

構成時には、SASAUTOS 用に指定されたすべてのディレクトリは SAS によって結合されます。ただし、セッションの開始後は、新しいディレクトリを指定すると現在の自動呼び出しライブラリはオーバーライドされます。

NOSASAUTOS オプションを指定すると、前のすべての SASAUTOS 指定が SAS で無視されるようになります(SAS コマンド、構成ファイル、SASV9_OPTIONS 環境変数のどれで指定するかに関係ありません)。

SASAUTOS オプションのデフォルト値は SASAUTOS ファイル参照名です。ファイル参照に割り当てられる UNIX ディレクトリはないため、SASAUTOS ファイル参照名を自動呼び出しライブラリとして使用したい場合は、これを定義する必要があります。

例

例 1: OPTIONS ステートメントでの複数の環境変数の指定

次の例は、複数の環境変数を OPTIONS ステートメントで指定する場合に使用する構文を示しています。

```
options sasautos=(AUTODIR, SASAUTOS);
```

指定する環境変数を定義する必要があります。たとえば、次のコードを使用すると、SAS 起動時に AUTODIR 環境変数を定義できます。

```
-set AUTODIR /tmp/sasautos
```

環境変数の設定方法に関する詳細については、“SET システムオプション: UNIX” (417 ページ)を参照してください。

例 2: OPTIONS ステートメントでのファイル参照名の指定

指定するファイル参照名を定義する必要があります。たとえば、FILENAME ステートメントを使用すると、AUTODIR ファイル参照名を定義できます。

```
filename AUTODIR '/tmp/sasautos';
```

ファイル参照名を定義したら、OPTIONS ステートメントでそれを使用して自動呼び出しライブラリを設定します。

```
options sasautos=autodir;
```

関連項目:

システムオプション:

- “APPEND= System Option” (*SAS System Options: Reference*)
- “INSERT= System Option” (*SAS System Options: Reference*)
- “MAUTOSOURCE System Option” (*SAS Macro Language: Reference*)
- “MRECALL System Option” (*SAS Macro Language: Reference*)

SASHELP システムオプション: UNIX

Sashelp ライブラリの場所を指定します。

該当要素:	構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数
カテゴリ:	環境コントロール: ファイル
PROC OPTIONS GROUP=	ENVFILES
デフォルト:	!SASROOT/sashelp (インストール済みの!SASROOT/sasv9.cfg ファイルで設定)
UNIX 固有:	*directory-specification* は環境変数にすることもできます。
参照項目:	“SASHELP= System Option” (*SAS System Options: Reference*)

構文

–SASHELP *directory-specification*

–SASHELP ('*directory-specification*', '*directory-specification* ...)

詳細

このオプションはインストールプロセス中に設定されますが、通常はインストール後に変更されます。環境変数を SASHELP の値として指定することができます。

追加ディレクトリ指定を付加するには、INSERT または APPEND システムオプションを使用します。詳細については、“INSERT システムオプション: UNIX” (391 ページ)、および “APPEND システムオプション: UNIX” (365 ページ)を参照してください。

関連項目:

システムオプション:

- “APPEND= System Option” (*SAS System Options: Reference*)
- “INSERT= System Option” (*SAS System Options: Reference*)

SASSCRIPT システムオプション: UNIX

SAS/CONNECT スクリプトファイルの 1 つ以上の格納場所を指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数
カテゴリ:	通信: ネットワーキングおよび暗号化
PROC OPTIONS GROUP=	COMMUNICATIONS
デフォルト:	!SASROOT/misc/connect
UNIX 固有:	複数のディレクトリ名を指定するための構文

構文

-SASSCRIPT '*directory-name*' | ('*directory-name-1*', …, '*directory-name-n*')

SASSCRIPT='*directory-name*' | ('*directory-name-1*', …, '*directory-name-n*')

詳細

同一の SASSCRIPT オプションに複数のディレクトリ名を指定する方法は、SASSCRIPT オプションを指定する場所に応じて異なります。

- オプションを構成ファイルまたは SASV9_OPTIONS 環境変数で指定する場合、複数の SASSCRIPT オプションを使用するか、またはディレクトリ名をかっこで囲みます。名前の区切りには、カンマまたは空白スペースを使用してください。
- オプションを SAS コマンドで指定する場合、かっこは構文エラーの原因となるため、複数の SASSCRIPT オプションを使用してください。
- オプションを OPTIONS ステートメントまたは **SAS System Options** ウィンドウに指定する場合、ディレクトリ名をかっこで囲む必要があります。名前の区切りには、カンマまたは空白スペースを使用してください。

関連項目:

システムオプション:

- “SASSCRIPT= System Option” (*SAS/CONNECT User's Guide*)

SASUSER システムオプション: UNIX

Sasuser ライブラリの名前を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: ~/sasuser.v94 (インストール済みの!SASROOT/sasv9.cfg ファイルで設定)

UNIX 固有: *pathname* は環境変数にすることができます。

参照項目: “SASUSER= System Option” (*SAS System Options: Reference*)

構文

–SASUSER *pathname*

詳細

pathname によって、ユーザーの Profile カタログが格納される Sasuser ライブラリのディレクトリが識別されます。パス名を指定するには、環境変数を使用します。次に例を示します。

```
sas -sasuser $HOME
```

SET システムオプション: UNIX

環境変数を定義します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: なし

UNIX 固有: すべて

構文

–SET *variable-name-value*

SET=*variable-name-value*

詳細

SET オプションを指定すると、SAS セッション内で有効な環境変数と、SAS セッション内から起動される任意のシェルを定義できます。SET の使い方は、SAS **setenv** コマンドの使い方と類似しています。SAS セッション内からのシステムコマンドの実行に関する詳細については、"SAS セッションからオペレーティングシステムコマンドの実行" (15 ページ)を参照してください。

SET オプションには、**!SASROOT** ディレクトリの名前を指定するという特別な使い方があります。

```
-set SASROOT pathname
```

指定したパス名は**!SASROOT** の展開に使用できます(表 2.3 (52 ページ)を参照)。

SAS セッションの終了後、SET オプションを指定して設定する環境変数は存在できなくなります。

関連項目:

- "UNIX 環境で環境変数を定義する" (439 ページ)
- "!SASROOT ディレクトリについて" (445 ページ)

SORTANOM システムオプション: UNIX

ホストソートユーティリティのオプションを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ソート: プロシジャオプション

PROC OPTIONS GROUP= SORT

デフォルト: なし

UNIX 固有: すべて

構文

SORTANOM=*option(s)*

–SORTANOM *option(s)*

必須引数

option
次のいずれか 1 つ以上になります。

B
SyncSort をシングルコースモードではなく、マルチコールモードで実行するよう指示します (詳細については、**syncsort** のドキュメントを参照してください。)

注 このオプションを使用できるのは **syncsort** のみです。

T
外部並べ替えプロセスに関して SAS ログ統計に書き込みます。

V
ホストソートユーティリティに受け渡されるコマンドをすべて SAS ログに書き込みます。

SORTCUT システムオプション: UNIX

SAS が、内部 SAS ソートではなくホストソートを使用するようになる上限のオブザベーション数のデータサイズを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ソート: プロシジャオプション

PROC OPTIONS GROUP= SORT

デフォルト: 0

UNIX 固有: すべて

構文

-SORTCUT *n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

SORTCUT=*n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
オブザベーションの数を 1 (*n*)、1,024 (*n*K)、1,048,576 (*n*M)、1,073,741,824 (*n*G)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、オブザベーションの数は、値が `800` の場合は 800、値が `.782k` の場合は 801、値が `3m` の場合は 3,145,728 に指定されます。

***hex*X**
オブザベーション数を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、値が `2ffx` の場合はオブザベーション数が 767 に指定されます。

MIN
オブザベーション数を 0 に指定します。

MAX
オブザベーション数を 9,007,199,254,740,992 に指定します。

詳細

SORTPGM=BEST を指定する場合、ホストソートまたは SAS ソートを使用するかどうかを決定するに当たり、SAS では SORTCUT および SORTCUTP オプションの値が使用されます。データセット内のオブザベーション数が SORTCUT で指定する数を超える場合、ホストソートが使用されます。SORTCUT および SORTCUTP の両方が定義されていないか、または 0 に設定されている場合、SAS ソートが使用されます。両方のオプションを指定し、いずれかの条件が true の場合、SAS ではホストソートが選択されます。

関連項目:

システムオプション:

- “SORTCUTP システムオプション: UNIX” (420 ページ)
- “SORTPGM システムオプション: UNIX” (422 ページ)

SORTCUTP システムオプション: UNIX

SAS が、内部 SAS ソートではなくホストソートを使用するようになる上限のデータサイズをバイトで指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ソート: プロシジャオプション

PROC OPTIONS GROUP= SORT

デフォルト: 0

UNIX 固有: すべて

構文

-SORTCUTP *n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

SORTCUTP=*n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
バイト数を 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、値が `8` の場合は 8 バイト、値が `782k` の場合は 801 バイト、値が `3m` の場合は 3,145,728 バイトが指定されます。

***hex*X**
バイト数を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、値が `2dx` の場合は 45 バイトが指定されます。

MIN
0 バイトを指定します。

MAX
9,007,199,254,740,992 バイトを指定します。

詳細

SORTPGM=BEST を指定する場合、ホストソートまたは SAS ソートを使用するかどうかを決定するに当たり、SAS では SORTCUT および SORTCUTP オプションの値が使用されます。並べ替え対象のデータセットが、SORTCUTP で指定するバイト数(またはキロバイト/メガバイト)より大きい場合、SAS ソートのかわりにホストソートが使用されます。ユーザーが指定する値は、2,147,483,647 バイト以下にする必要があります。SORTCUT および SORTCUTP の両方が定義されていないか、または 0 に設定されている場合、SAS ソートが使用されます。両方のオプションを指定し、いずれかの条件が true の場合、SAS ではホストソートが選択されます。

次の数式によって、並べ替え対象のバイト数が計算されます。

*number-of-bytes= ((length-of-obs)+(length-of-all-keys))*number-of-obs*

関連項目:

システムオプション:

- “SORTANOM システムオプション: UNIX” (418 ページ)
- “SORTCUT システムオプション: UNIX” (419 ページ)
- “SORTPGM システムオプション: UNIX” (422 ページ)

SORTDEV システムオプション: UNIX

ホストソートユーティリティによって作成された一時ファイルに使用されるパス名を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ソート: プロシジャオプション

PROC OPTIONS GROUP= SORT

デフォルト: -WORK(インストール済みの!SASROOT/sasv9.cfg ファイル内で設定)と同じ場所

UNIX 固有: すべて

構文

SORTDEV='*directory-specification*'

-SORTDEV *directory-specification*

詳細

SORTDEV オプションは、ホストソートプログラムによって作成された一時ファイルの代替ディレクトリを指定します。

SORTNAME システムオプション: UNIX

ホストソートユーティリティの名前を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ソート: プロシジャオプション

PROC OPTIONS GROUP= SORT

デフォルト: なし

UNIX 固有: すべて

構文

SORTNAME='*host-sort-utility-name*'

-SORTNAME *host-sort-utility-name*

詳細

SORTNAME オプションは、デフォルトのホストソートユーティリティの名前 `syncsort` を指定します。

関連項目:

システムオプション:

- “SORTPGM システムオプション: UNIX” (422 ページ)

SORTPARM システムオプション: UNIX

ホストソートユーティリティにパラメータを指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数
カテゴリ:	ソート: プロシジャオプション
PROC OPTIONS GROUP=	SORT
デフォルト:	なし
UNIX 固有:	すべて

構文

SORTPARM='*parameter(s)*'

–**SORTPARM** '*parameter(s)*'

必須引数

parameter
ソートユーティリティに受け渡す任意のパラメータを指定します。このパラメータの説明については、使用するソートのドキュメントを参照してください。

SORTPGM システムオプション: UNIX

内部 SAS ソートユーティリティを使用するか、ホストソートユーティリティを使用するか、または使用するソートユーティリティを SAS に選択させるかを指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数
カテゴリ:	ソート: プロシジャオプション
PROC OPTIONS GROUP=	SORT
デフォルト:	BEST
UNIX 固有:	すべて

構文

-SORTPGM SAS | HOST | BEST

SORTPGM=SAS | HOST | BEST

必須引数

SAS
SAS で SAS ソートを使用するように指示します。

HOST
SAS で、SORTNAME システムオプションによって指定されたソートを使用するように指示します。

BEST
データセットの並べ替えに当たり、最善のルーチン(SAS ソート、または SORTNAME システムオプションで指定したホストソート)を使用するよう SAS に指示します。SORTCUT および SORTCUTP システムオプションの設定によって、SAS ソートまたはホストソートのどちらが SAS で選択されるかが決まります。

詳細

SORTPGM システムオプションは、SAS ソートを使用するか、ホストソートを使用するか、データセットに最適なソートがどれかを決定させるかを SAS に指示します。

関連項目:

システムオプション:

- “SORTCUTP システムオプション: UNIX” (420 ページ)
- “SORTNAME システムオプション: UNIX” (421 ページ)
- “SORTSIZE システムオプション: UNIX” (423 ページ)

SORTSIZE システムオプション: UNIX

SORT プロシジャに使用可能なメモリ容量を指定します。

該当要素:	構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数
カテゴリ:	ソート: プロシジャオプション システム管理: メモリ
PROC OPTIONS GROUP=	SORT MEMORY
デフォルト:	1G
UNIX 固有:	MAX の値
参照項目:	“SORTSIZE= System Option” (*SAS System Options: Reference*)

構文

–SORTSIZE *n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

SORTSIZE=*n* | *n*K | *n*M | *n*G | *hex*X | MIN | MAX

必須引数

***n* | *n*K | *n*M | *n*G**
バイト数を 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。キロバイト、メガバイトまたはギガバイトの数を 10 進値で指定できます。たとえば、値が `8` の場合は 8 バイト、値が `.782k` の場合は 801 バイト、値が `3m` の場合は 3,145,728 バイトが指定されます。

***hex*X**
メモリ容量を 16 進数値として指定します。値は、先頭が数字(0–9)、次いで 16 進数文字(0–9、A–F)、最後に X が含まれるように指定する必要があります。たとえば、値が `2dx` の場合はメモリ容量が 45 バイトに設定されます。

MIN
0 バイトを指定します(MEMSIZE システムウオプションによって指定された制限を除き、制限がないことを示します)。

MAX
動作環境のアドレス可能な最大メモリを指定します。

詳細

SORT プロシジャでは、並べ替え用に取得または割り当てるメモリ容量に制限を設けるため、SORTSIZE システムオプションが使用されます。また、SAS で SORT プロシジャに使用されるメモリ容量は、MEMSIZE および REALMEMSIZE システムオプションの値に応じて異なります。SORTSIZE オプションとは対照的に、MEMSIZE オプションは、SAS で常に動的に割り当てられる仮想メモリの合計容量に制限を加えます。こうした仮想メモリは、実メモリとページング領域を組み合わせてサポートされます。動作環境では、必要な仮想メモリの容量が使用可能な実メモリ容量を超過すると、ページングが開始されます。ページングや関連するパフォーマンスの問題が発生するのを防ぐには、SORTSIZE システムオプションを実メモリのサブセットに設定する必要があります。MEMSIZE が実メモリのサブセットに設定されている場合、SORTSIZE を MAX に設定できます。多くの場合、SORTSIZE=MAX を設定してもよいのは、SORT プロシジャで使用するメモリ容量が MEMSIZE で制限されているためです。

関連項目:

システムオプション:

- “MEMSIZE システムオプション: UNIX” (398 ページ)

プロシジャ:

- “SORT プロシジャ: UNIX” (317 ページ)

STDIO システムオプション: UNIX

SAS で stdin、stdout、stderr が使用されるかどうかを指定します。

該当要素:	構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数
カテゴリ:	入力コントロール: データ処理
PROC OPTIONS GROUP=	INPUTCONTROL

デフォルト: NOSTDIO

UNIX 固有: すべて

構文

–STDIO | **–NOSTDIO**

詳細

このオプションは、標準出力(stdin)から入力を抽出し、ログを標準エラー(stderr)に書き込み、出力を標準出力(stdout)に書き込むよう SAS に指示します。

このオプションは、バッチモードまたはシェルスクリプトから SAS を実行するように設計されています。このオプションを対話型で指定する場合、SAS でラインモードセッションが開始されます。STDIO オプションは、DMS、DMSEXP、EXPLORER システムオプションをオーバーライドします。

STDIO オプションによって、Stdio、Stdin、Stderr ファイル参照名の割当は影響を受けません。詳細については、“UNIX 環境で SAS によって割り当てられるファイル参照名” (78 ページ)を参照してください。

たとえば、次の SAS コマンドでは、*myinput* ファイルはソースプログラムとして使用され、*myoutput* および *mylog* ファイルはそれぞれプロシジャ出力およびログに使用されています。

```
sas -stdio <myinput> myoutput> mylog
```

C シェルを使用する場合、かっこを使用する必要があります。

```
(sas -stdio <myinput> myoutput)>& output_log
```

関連項目:

“UNIX 環境での、SAS ログと SAS プロシジャのデフォルトの出力先” (91 ページ)

STIMEFMT システムオプション: UNIX

FULLSTIMER および STIMER 出力で時間の表示に使用される形式を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: 512M

UNIX 固有: すべて

構文

-STIMEFMT *value(s)*

STIMEFMT=*value(s)*

必須引数

value

STIMEFMT で使用するオプションを指定します。次のオプションを使用できます。

日時スタンプオプション

日時スタンプオプションの説明は次のとおりです。

TS	日時スタンプを STIMER および FULLSTIMER として常に表示するよう指定します。
TSFULL	日時スタンプを FULLSTIMER の一部として表示するよう指定します。TSFULL はデフォルトです。
TSOFF	STIMER および FULLSTIMER の日時スタンプをオフにします。

Memory

通常は FULLSTIMER の一部として表示されます。デフォルトのメモリ出力はキロバイト単位で表示されます。メモリには次のオプションを使用できます。

MEMFULL	メモリ統計情報を STIMER の一部ではなく、FULLSTIMER の一部として書き込みます。
MEM	メモリ統計情報を FULLSTIMER および STIMER の一部として書き込みます。
KB	メモリをキロバイト単位で書き込みます。
MB	メモリをメガバイト単位で書き込みます。
GB	メモリをギガバイト単位で書き込みます。
C	メモリ表示の数字にカンマを追加します。
NC	メモリ表示の数字にカンマを追加しません。

経過時間および CPU 時間

時間、分、秒、あるいは STIMER と FULLSTIMER に最適な表示のいずれかに設定できます。

Z | H | HOURS	時刻を時間:分:秒の形式で書き込みます。
M | MINUTES	時刻を分:秒の形式で書き込みます。
S | SECONDS	時刻を秒形式で書き込みます。
HMS	時間と分の先頭に 0 を残した形式で書き込みます。

カウンタ

追加カウンタを FULLSTIMER の一部として表示できるように指定します。

E | ENABLE	追加カウンタを有効化します。
D | DISABLE	追加カウンタを無効化します。

Help

STIMEFMT オプションのヘルプへのアクセスに使用される 2 つの値を提供します。

FMT	使用可能な日時スタンプ形式をリストで表示します。
OPT	使用可能な他のオプション値をリストで表示します。

詳細

STIMEFMT の基本

STIMEFMT システムでは、STIMER および FULLSTIMER システムオプションによって生成される出力の形式をカスタマイズできます。STIMEFMT を使用すると、次のタスクを実行できます。

- 使用可能な形式をリストで表示:

```
options stimefmt=fmt;
```

- 使用可能な他のオプションをリストで表示:

```
options stimefmt=opt;
```

- STIMER の日時スタンプをオン/オフ:

```
options stimefmt=tson | tsoff | tsfull;
```

- 必要に応じてオプションの組み合わせ:

```
options stimefmt=(tson YYNNDDS);
```

- メモリ値をカンマでの区切り:

```
options stimefmt=c;
```

- 値を指定する際にカンマを使用しない:

```
options stimefmt=nc;
```

- メモリの単位を選択:

```
options stimefmt=GB | MB | KB;
```

- STIMER および FULLSTIMER のメモリレポート作成をオンにする:

```
options stimefmt=mem;
```

- 日時スタンプで時刻表示を設定する:

```
options stimefmt=TOD | TIME | TIMEAMPM;
(TOD and TIME specify military time.)
```

- 時間または分で CPU または処理時間の表示を制御する

日時スタンプの表示形式

日時スタンプの型式は、SAS でサポートされる標準形式に設定できます。その形式には次が含まれます。

```
ABS. (Absolute seconds since Jan. 1, 1970)


DATE. DATE9.


DDMMYY. DDMMYY10. DDMMYYB.
DDMMYYB10. DDMMYYC. DDMMYYC10.
DDMMYYD. DDMMYYD10. DDMMYYN.
DDMMYYN10. DDMMYYP. DDMMYYP10.
DDMMYYS. DDMMYYS10.


ISO. (ISO Standard Time)
```

```
MMDDYY. MMDDYY10. MMDDYY.
MMDDYYB10. MMDDYYC. MMDDYYC10.
MMDDYYD. MMDDYYD10. MMDDYYN.
MMDDYYN8. MMDDYYP. MMDDYYP10.
MMDDYYS. MMDDYYS10.


NLDATM. NLDATMAP.


YYMMDD. YYMMDD10. YYMMDDB.
YYMMDDB10. YYMMDDC. YYMMDDC10.
YYMMDDD. YYMMDDD10. YYMMDDN.
YYMMDDN8. YYMMDDP. YYMMDDP10.
YYMMDDS. YYMMDDS10.


TOD. (Writes time as military time.)
TIME. (Writes time as military time.)
TIMEAMPM. (Writes time as AM and PM.)
```

OPTIONS ステートメントの構文は次の表に記載されています。

```
options stimefmt=fmt;
```

ここで *fmt* は有効な SAS 形式です。

STIMEFMT オプションの複数値の使用

STIMEFMT オプションは、複数の値を同時に指定し、ユーザーが複数の設定を設定できるようにします。複数の値は、かっこで囲む必要があります。たとえば、次のようになります。

```
options stimefmt=(h YYMMDD. gb c);
```

STIMEFMT オプションの設定の表示

PROC OPTIONS は、STIMEFMT のすべての設定に関する現在の状態を常に表示します。次の例は、PROC OPTIONS を実行する際のログ出力を示しています。

```
proc options option=stimefmt;
run;
```

ログ 19.2 PROC OPTIONS からのログ出力

```
 SAS (r) Proprietary Software Release 9.4

STIMEFMT=(NLDATM2. HMS TIMEAMPM KB MEMFULL TSFULL NC)
Specifies the format that is used to display the FULLSTIMER
and STIMER output for timestamp, memory, CPU and elapsed time
statistics.
NOTE: PROCEDURE OPTIONS used (Total process time):
real time 0.00 seconds
user cpu time 0.00 seconds
system cpu time 0.00 seconds
memory 21.37k
OS Memory 11932.00k
Timestamp 4/12/2013 01:51:52 PM
Step Count 14 Switch Count 0
Page Faults 0
Page Reclaims 1
Page Swaps 0
Voluntary Context Switches 0
Involuntary Context Switches 0
Block Input Operations 0
Block Output Operations 0
```

STIMEFMT をデフォルト値にリセット

STIMEFMT の設定をデフォルト値にリセットするには、次の OPTIONS ステートメントを実行します。

```
options stimefmt=normal;
```

関連項目:

システムオプション:

- “FULLSTIMER システムオプション: UNIX” (383 ページ)
- “STIMER システムオプション: UNIX” (429 ページ)

STIMER システムオプション: UNIX

システムパフォーマンス統計情報のサブセットを SAS ログに書き込むかどうかを指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: STIMER

UNIX 固有: すべて

構文

–STIMER | –NOSTIMER

STIMER | NOSTIMER

必須引数

STIMER
処理時間と CPU 時間のみを SAS ログに書き込みます。

NOSTIMER
統計情報を SAS ログに書き込みません。

詳細

STIMER システムオプションは、SAS に使用可能なシステムのすべてのパフォーマンス統計情報のサブセットを SAS ログに書き込むかどうかを指定します (STIMEFMT を指定すると、出力が影響を受けます)。次の例は、STIMER 出力の例です。

ログ 19.3 STIMER 出力

```
real time 1.34 seconds
cpu time 0.04 seconds
```

STIMER は次の統計情報を表示します。

表 19.2 STIMER 統計情報の説明

統計情報	説明
real time	SAS ジョブの処理にかかる時間。Real time は経過時間ともいいます 。
CPU time	SAS コードを実行したり、SAS プロセスのかわりにシステムオーバーヘッドタスクを実行したりするために費やした合計時間。この値は、FULLSTIMER からのユーザー CPU とシステム CPU 統計情報の組み合わせとなります。

FULLSTIMER および STIMER システムオプションの両方を設定する場合、FULLSTIMER 統計情報はログに書き込まれます。

注: SAS 9 からは、一部のプロシジャで複数のスレッドが使用されています。複数の CPU を持つコンピュータでは、複数のスレッドで同時にオペレーティングシステムを実行できます。その結果、CPU 時間は STIMER 出力の処理時間を超過する場合があります。たとえば、SAS プロシジャでは 2 つの別々の CPU で同時に実行される 2 つのスレッドが使用される可能性があります。CPU 時間の値は、次の計算式に従って計算されます。

```
CPU1 time + CPU2 time = total CPU time
1 second + 1 second = 2 seconds
```

CPU1 は、CPU2 が同一の SAS プロセスの別個のスレッドを実行すると同時にスレッドを実行するため、理論上は処理時間の 1 秒間に CPU 時間では 2 秒間を消費することができます。

関連項目:

システムオプション:

- “FULLSTIMER システムオプション: UNIX” (383 ページ)
- “STIMEFMT システムオプション: UNIX” (425 ページ)

SYSIN システムオプション: UNIX

バッチモードで実行時の SAS ソースコードのデフォルトの場所を指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: なし

UNIX 固有: すべて

構文

–SYSIN *filename* | **-NOSYSIN**

必須引数

-SYSIN *filename*
外部ファイルを指定します。*filename* の値は、有効な UNIX ファイル名にする必要があります。

-NOSYSIN
SAS の起動、autoexec ファイルの処理、SAS の終了の順に実行し、ユーザーにコマンドプロンプトを返します。

詳細

このオプションは、バッチモードの使用時にのみ適用されます。ファイル名の直後にキーワード **SAS** がある場合は、ファイル名の前に SYSIN オプションを指定する必要はありません。たとえば、次の 2 つの SAS コマンドには同じ機能があります。

```
 sas saspgms/report1.sas
sas -sysin saspgms/report1.sas
```

SYSIN システムオプションの構文を使用すると、NOSYSIN を指定できます。NOSYSIN を指定する場合、SAS の起動、autoexec ファイルの処理、SAS の終了の順に実行され、ユーザーにコマンドプロンプトが返されます。次は、構文の例を示します。

```
sas -nosysin -autoexec mysas.sas
```

完全な SAS セッションを実際に実行せずに autoexec ファイルをテストする場合は、このオプションが有効です。

関連項目:

“UNIX 環境で SAS セッションを開始する” (4 ページ)

SYSPRINT システムオプション: UNIX

印刷出力の出力先を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: プロシジャ出力

PROC OPTIONS GROUP= LISTCONTROL および ODSPRINT

デフォルト: デフォルトのシステムプリンタ

UNIX 固有: すべて

構文

–SYSPRINT *destination* | '*destination-option-list*'

SYSPRINT=*destination* | '*destination-option-list*'

必須引数

destination
サイトのハードコピーデバイスの名前です。使用可能な出力先のリストについては、システム管理者にご相談ください。

destination-option-list
`lp` (または `lpr`)コマンドに受け渡すオプションのリストです。

詳細

SYSPRINT オプションは、デフォルトのシステムプリンタ以外の印刷出力の出力先を指定します。オプションを `lp` (または `lpr`)コマンドに受け渡すには、オプションリストを使用します。

注: ファイル参照名を割り当てると、SYSPRINT オプションにクエリが送信されます。SYSPRINT オプションの値を後で変更しても、ファイル参照名はその変更内容を反映しません。

詳細については、“UNIX 環境でデフォルトの印刷コマンドを変更する” (102 ページ)を参照してください。

関連項目:

コマンド:

- “PRINTCMD システムオプション: UNIX” (408 ページ)

その他の参照資料:

- “UNIX 環境における出力印刷の概要” (90 ページ)

USER システムオプション: UNIX

デフォルトの永久 SAS ライブラリの名前を指定します。

該当要素: 構成ファイル、SAS 起動、OPTIONS ステートメント、**SAS System Options** ウィンドウ、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: なし

UNIX 固有: *pathname* は有効な UNIX パス名にする必要があります。

参照項目: “USER= System Option” (*SAS System Options: Reference*)

構文

-USER *pathname*

USER='*pathname*' | *libref*

必須引数

pathname
デフォルトの永久 SAS ライブラリを格納するディレクトリを識別します。これは、ディレクトリ名にする必要があります。

libref
デフォルトの永久 SAS ライブラリを格納するディレクトリに関連付けられるライブラリ参照名です。すでに割り当てられているはずです。

関連項目:

“1 レベル名を使用した永久ファイル(ユーザーライブラリ)へのアクセス” (59 ページ)

VERBOSE システムオプション: UNIX

SAS でシステムオプション設定を SAS ログに書き込むかどうかを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: ログおよびプロシジャ出力コントロール: SAS ログ

PROC OPTIONS GROUP= LOGCONTROL

デフォルト: NOVERBOSE

UNIX 固有: すべて

構文

–VERBOSE | **–NOVERBOSE**

必須引数

-VERBOSE
構成ファイル、SAS コマンド、SASV9_OPTIONS 環境変数から SAS システムオプションの設定を SAS ログに書き込みます。CONFIG オプションでは、VERBOSE によって構成ファイルの名前がリストで表示されます。

-NOVERBOSE
システムオプションの設定を SAS ログに書き込みます。

詳細

SAS の前のリリースでは、VERBOSE システムオプションからの出力は、オプションと値の簡易リストとして表示されていました。このリストは、SAS 起動時のウィンドウに表示されていました。ENTER キーを押すと、1 回ごとにリストを 1 行ずつ進めることができました。スペースバーを押すと、リストをページごとに進めることができました。Q キーを押すとリスト全体が表示され、プロンプトが返されました。

SAS 9.4 以降、システムオプションと値のリストは引き続き作成されます。また、SAS ではオプションの設定場所を識別するリストが作成されます。このリストは、グローバルジャーナルリスト、SAS ログの順に書き込まれます。グローバルジャーナルファイルに書き込む利点は、SAS で初期化に失敗した場合に、SAS ログが作成されなかった場合でも、出力は使用できる点です。

関連項目:

システムオプション:

- “OPLIST システムオプション: UNIX” (404 ページ)

その他の参照資料:

- “システムオプションを使用し、SAS セッションをカスタマイズする” (18 ページ)

WORK システムオプション: UNIX

Work ライブラリの場所を指定します。

該当要素:	構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数
カテゴリ:	環境コントロール: ファイル
PROC OPTIONS GROUP=	ENVFILES
デフォルト:	インストール済みの!SASROOT/sasv9.cfg ファイル内で設定
UNIX 固有:	すべて
注:	このオプションは、サイト管理者は制限できます。詳細については、“Restricted Options” (*SAS System Options: Reference* 1 章)を参照してください。
参照項目:	“WORK= System Option” (*SAS System Options: Reference*)

構文

–WORK *filename* | *directory*

必須引数

filename
ディレクトリとオプションのキーワードのリストを含むファイルを指定します。現在の SAS セッションの Work ライブラリの場所として、そのファイルにあるリストから 1 つのディレクトリを SAS が選択します。

directory
現在の SAS セッションの Work ライブラリの場所として、ディレクトリを指定します。

詳細

基本

filename オプションを使用する場合、SAS はそのファイルを開き、現在の SAS セッションの Work ライブラリの場所として使用するディレクトリを 1 つ選択します。ディレクトリは、ランダム、または使用可能なスペースに基づくかのいずれかで SAS が選択します。ユーザーの選択を生かしたい場合は、METHOD キーワードを使用します。

directory オプションを使用する場合、Work ライブラリの場所として指定されたディレクトリを使用して SAS は引き続きその初期化を行います。

Work ライブラリの割当をより動的に実行

filename オプションには、Work ライブラリに使用できるディレクトリのリストが含まれます。個別の SAS Work ライブラリが 1 つのディレクトリ内に格納されます。METHOD=RANDOM は、Work ライブラリのディレクトリがディレクトリのリストからランダムに選ばれることを示します。SAS は、セッションごとに 1 つのディレクトリを Work ライブラリの場所として選択します。この選択によって、複数のハードウェアシステム全体で I/O 読み込みのバランスを取ることができます。使用可能な最大限の領域を指定するには、METHOD=SPACE を使用します。METHOD キーワードが指定されていない場合、SAS のデフォルト設定でディレクトリのランダムな選択が行われます。

例

例 1: 複数の異なるディスクボリューム全体の処理負荷の分散

次の例は、複数の異なるディスクボリューム全体の I/O 処理負荷を分散する方法を示しています。この場合、METHOD=RANDOM を使用します。**/sasinfo/workfiles** という名前のファイルには次の情報が含まれます。

```
/disk1/sastempfiles
/disk2/sastempfiles
/disk3/sastempfiles
method=random
```

特定の SAS セッションの Work ライブラリは、disk1、disk2、disk3 のいずれかに配置されます。構成ファイルまたはコマンド行には、次の構文が含まれます。

```
-work /sasinfo/workfiles
```

例 2: 最大空き領域を持つディレクトリの選択

データを処理する際には、最大空き領域を持つディレクトリを選択できます。この場合、METHOD=SPACE を使用します。次の例では、**/sasinfo/workfiles** には次のディレクトリが含まれます。

```
/disk1/sastempfiles
/disk2/sastempfiles
/disk3/sastempfiles
method=space
```

Work ライブラリは、最大空き領域を持つディスクに配置されます。

関連項目:

システムオプション:

- “WORKINIT システムオプション: UNIX” (436 ページ)

WORKINIT システムオプション: UNIX

Work ライブラリを初期化します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: WORKINIT

UNIX 固有: WORKINIT は、前回のセッションからファイルを消去しません

参照項目: “WORKINIT System Option” (*SAS System Options: Reference*)

構文

–WORKINIT | **–NOWORKINIT**

必須引数

-WORKINIT
WORK オプションで指定したディレクトリ内に新しいサブディレクトリが作成されるように指定します。

-NOWORKINIT
WORK オプションで指定したディレクトリがシステムで使用されるように指定します。

- システムで古いサブディレクトリが検出されなかった場合は、新しいサブディレクトリが作成されます。
- システムで複数の古いサブディレクトリが検出された場合、最新のサブディレクトリが使用されます。
- ファイルのロックが有効な場合(“FILELOCKS システムオプション: UNIX” (377 ページ)を参照)、ロックされていない最新のディレクトリが自動的に検索されます。何も検出されない場合は、新しいディレクトリが作成されます。

詳細

WORKINIT オプションは、SAS 起動時に Work ライブラリを初期化するかどうかを制御します。

関連項目:

システムオプション:

- “FILELOCKS システムオプション: UNIX” (377 ページ)
- “WORK システムオプション: UNIX” (434 ページ)

WORKPERMS システムオプション: UNIX

SAS Work ライブラリの初期作成時に、その権限を設定します。

該当要素: 構成ファイル、SAS 起動

カテゴリ: 環境コントロール: ファイル

PROC OPTIONS GROUP= ENVFILES

デフォルト: 700

UNIX 固有: すべて

構文

–WORKPERMS *permission-value*

必須引数

permission-value
SAS Work ディレクトリの権限を示す 8 進数値を指定します。値は、UNIX ディレクトリの権限を設定する任意の 8 進数値になります。値の例として、マスクなし、700、755、770、775、777 が挙げられます。

詳細

WORKPERMS システムオプションを使用すると、最初に SAS Work ライブラリを作成する際に、現在のファイルモード作成マスク値を変更または削除できます。つまり、*permission-value* の値を変更すると、新しい Work ライブラリ用の権限を変更できます。

XCMD システムオプション: UNIX

SAS セッションで X コマンドが有効であるかどうかを指定します。

該当要素: 構成ファイル、SAS 起動、SASV9_OPTIONS 環境変数

カテゴリ: 環境コントロール: 表示

PROC OPTIONS GROUP= ENVDISPLAY

デフォルト: XCMD

UNIX 固有: すべて

構文

-XCMD | **-NOXCMD**

必須引数

-XCMD
現在の SAS セッションで X コマンドが有効であるように指定します。

-NOXCMD
現在の SAS セッションで X コマンドが有効でないように指定します。

詳細

XCMD システムオプションは、現在の SAS セッションで X コマンドが有効であるかどうかを指定します。

NOXCMD システムオプションを使用する場合は、複数の SAS ステートメント、オブジェクト、機能のいずれかを使用することはできません。こうしたステートメント、オブジェクト、機能の例は次のとおりです。

- FILENAME ステートメント内の PIPE デバイス型
- CALL SYSTEM ルーチン
- %SYSEXEC マクロ
- シェルレベルコマンドを実行するために SAS で使用される機能

関連項目:

CALL ルーチン:

- “CALL SYSTEM ルーチン: UNIX” (271 ページ)

コマンド:

- “X コマンド: UNIX” (249 ページ)

マクロ:

- “%SYSEXEC” (297 ページ)

ステートメント:

- “FILENAME ステートメント: UNIX” (331 ページ)

その他の参照資料:

- “SAS セッションからオペレーティングシステムコマンドの実行” (15 ページ)

20 章

UNIX での環境変数

UNIX 環境で環境変数を定義する

UNIX 環境変数について

UNIX 環境変数は現在のシェルと作成されるサブシェルの両方に適用されます (たとえば、ジョブをバックグラウンドに送ったり、スクリプトを実行した場合)。環境変数の値を変更した場合、その変更はそれ以降のシェルに送られますが、それ以前の親シェルには送られません。

SAS セッションにて、SASV9_OPTIONS 環境変数を使用してシステムオプションを指定でき、SASV9_CONFIG 環境変数を使用して構成ファイルを指定できます。SAS セッションの初期化後に環境変数に加えた変更はいずれも認識されません。

さまざまなステートメントやコマンドにてファイル参照名およびライブラリ参照名などの環境変数を使用することができます。ファイル参照名およびライブラリ参照名は環境変数名の大文字、数字、そして小文字から構成されます。他の文字は SAS では認識されません。詳細については、“UNIX 環境におけるライブラリ参照名としての環境変数の使用” (54 ページ)または“UNIX 環境で環境変数を使用してファイル参照名を割り当てる” (77 ページ)を参照してください。

注: SAS/ACCESS 製品はローディング時に必要な環境変数を初期化します。詳細については、SAS/ACCESS 製品のドキュメントを参照してください。

シェルに対する環境変数の定義

環境変数の定義

環境変数を定義する方法は実行中のシェルによります。(どのシェルが実行中か決定するには、コマンドプロンプトで **ps** または **echo $SHELL** とタイプしてシェル環境変数の現在値を確認します)。

Bourne シェルと Korn シェル

Bourne シェルおよび Korn シェルでは、**export** コマンドを使用して1つまたはそれ以上の変数を環境にエクスポートします。たとえば、次のコマンドは **scname** 変数の値をすべてのそれ以降のシェルスクリプトに有効にします。

```
$ scname=phonelist
$ export scname
```

Korn シェルでは、これらのコマンドを 1 つのコマンドに結合できます。

```
$ export scname=phonelist
```

scname の値を変更した場合、新しい値はシェル変数と環境変数の両方に影響します。変数をエクスポートしない場合、定義してシェルスクリプトのみがその値にアクセスします。

C シェル

C シェル(csh および tcsh)では、**setenv**(環境の設定)コマンドにて環境変数の設定(定義およびエクスポート)をします。たとえば、このコマンドは以前に示したコマンドと同等です。

```
% setenv scname phonelist
```

環境変数の値の表示

個別の環境変数の値を表示するには、**echo** コマンドとパラメータ代替を使用します。例としては、SHELL 環境変数の現在の値を返す **echo $SHELL** があります。**env**(または **printenv**) コマンドを使って、すべての環境変数と現在の値を表示することができます。

ディクショナリ

PATHENCODING 環境変数: UNIX

エンコーディングが SAS セッションエンコーディングと異なる場合、外部ファイル参照用とディレクトリ参照用のエンコーディングを指定します。

カテゴリ: 環境コントロール

デフォルト: なし

要件 パス名の文字には、セッションエンコーディングと、PATHENCODING によって指定されるエンコーディングの両方で認識される文字を含める必要があります。

UNIX 固有: すべて

詳細

“シェルに対する環境変数の定義” (440 ページ)の指示に従って PATHENCODING 環境変数の値を設定します。

割り当てるエンコーディング値によって、SAS プログラム内からアクセスされる外部ファイル参照とディレクトリ参照のためのエンコーディングが指定されます。外部ファイルエンコーディングとディレクトリエンコーディングが SAS セッションエンコーディングと異なる場合、この環境変数に値を指定します。SAS は外部ファイルとディレクトリを参照する時にデフォルトのセッションエンコーディングを使用します。PATHENCODING 環境変数は、外部ファイルおよびディレクトリ参照に代替のエンコーディングを提供します。PATHENCODING は、ディスクにあるファイルにのみ有効です。PATHENCODING 環境変数が有効なエンコーディング値である場合、SAS では、パス名が SAS セッションエンコーディングから指定されたエンコーディングにトランスコードされます。

UNIX で有効なエンコーディング値のリストについては、“UNIX Encoding Values” (*SAS National Language Support (NLS): Reference Guide* 23 章)を参照してください。

SAS プログラム内で指定するパス名は、SAS セッションエンコーディングで入力する必要があります。PATHENCODING 環境変数に指定したエンコーディングでパス名を指定しないでください。

SAS 9.4 のメンテナンスリリース 2 では、英語を使用する SAS セッションで UTF-8 の PATHENCODING 値を指定するには、パス名の文字に、セッションエンコーディングと、PATHENCODING によって指定されるエンコーディングの両方で認識される文字を含める必要があります。結果として、英語(LANG=EN)を使用する SAS セッションで UTF-8 の PATHENCODING 値を指定するには、UTF-8 または SAS_U8 の SAS セッションエンコーディングを指定する必要があります。

SASV9_CONFIG 環境変数: UNIX

SAS セッションの開始時に参照される構成ファイルを指定します。

カテゴリ: 環境コントロール

デフォルト: なし

UNIX 固有: すべて

詳細

“シェルに対する環境変数の定義” (440 ページ)の指示に従って SASV9_CONFIG 環境変数の値を設定します。

SASV9_CONFIG に割り当てるファイル指定によって、SAS セッションで使用されるパスと構成ファイル名が指定されます。構成ファイルには、SAS セッションで使用する SAS システムオプションがすべて含まれます。たとえば、Korn シェルでは、次のようにホームディレクトリの custom.cfg ファイルを SASV9_CONFIG に割り当てることもできます。

```
> export sasv9_config=/u/<user_id>/custom.cfg
```

SASV9_OPTIONS 環境変数: UNIX

SAS セッションの開始時に自動的に使用される SAS システムオプションのリストを指定します。

カテゴリ: 環境コントロール

デフォルト: なし

UNIX 固有: すべて

詳細

SAS セッションの開始時に自動的に使用される SAS システムオプションのリストを指定するには、SASV9_OPTIONS 環境変数を使用します。通常は、SAS で作業をするたびごとに同じ SAS システムオプションを設定するという場合は、これが役立ちます。

5 部

付録

付録 1

!SASROOT ディレクトリ

!SASROOT ディレクトリについて

SAS をインストールすると、そのディレクトリ構造はファイルシステムのディレクトリに配置されます。このディレクトリは **SASHOME** と言います。**SASHOME** ディレクトリは、ファイルシステムの任意の場所に配置できます。**SASHOME** のデフォルトの場所は**/usr/local/SAS** です。従来の**!SASROOT** ディレクトリ(SAS Foundation)は、**SASHOME** に格納されているサブディレクトリに自動的にインストールされます。**!SASROOT** のデフォルトディレクトリは **SASHOME/SASFoundation/9.x** です。ここでは、**x** は SAS リリースのバージョンを示します。

!SASROOT ディレクトリのコンテンツ

!SASROOT ディレクトリには、SAS の使用に必要なファイルが含まれます。このディレクトリは、起動点、構成ファイル、サンプルプログラム、カタログ、データセットおよび実行可能なファイルを含みます。SAS を使用するには、このディレクトリの構成を知る必要はありません。

システムにインストール可能な SAS 製品がすべてインストールされている場合、**!SASROOT** ディレクトリは次の表に示しているファイルおよびディレクトリを含みます。

表 A1.1 *!SASROOT ディレクトリ内の SAS ファイル*

SAS ファイル	内容の説明
sas	SAS のデフォルト起動点です。
sassetup	これによって、SAS ライセンスの更新ができます。
setinit.sas	ライセンス情報の更新に使用された SAS ファイルです。
sasv9.cfg	SAS のデフォルトのシステム構成ファイルです。このファイルを編集しないでください。(sasv9_local.cfg を参照)

SAS ファイル	内容の説明
sasv9_local.cfg	このファイルには、ユーザー指定のシステムオプションを追加します。このファイルはデフォルトのシステム構成ファイルのオプションをオーバーライドし、SAS の再インストールやアップグレード時にオプションが失われないようにします。

表 A1.2 !SASROOT ディレクトリ内 SAS サブディレクトリ

SAS サブディレクトリ	内容の説明
bin	**NLS** ディレクトリにある各言語に対する呼び出しスクリプトを含みます。このディレクトリは、SAS が必要とする環境変数を設定する sasenv スクリプトも含みます。sasenv_local は、ユーザーが変更するファイルです。また、sasenv_local は SAS が環境変数を処理する際に読み込む最後のファイルとなります。
dbcs	DBCS のインストールに使用されるサブディレクトリを含みます。
install	sassetup によって使用されるデータファイルおよびサブフォルダが格納されている管理サブフォルダを含みます。インストール後の処理時に、SAS レジストリの作成に使用されるデータファイルが格納されているレジストリおよび sasregord サブフォルダも含みます。
maps	地図の作成時に、SAS/GRAPH ソフトウェアによって使用される SAS データセットを含む SAS ライブラリです。SAS/GRAPH ソフトウェアと一緒に一部の地図が提供されます。追加の地図は、SAS Map データライブラリシリーズに含まれています。
misc	その他のコンポーネントを含みます。このディレクトリには、SAS/CONNECT ソフトウェアのスクリプトファイルや SAS/SHARE ソフトウェアのシンクライアントインターフェイスなど、さまざまな SAS 製品のコンポーネントも含まれます。このディレクトリでは、SAS のインストールプログラムが必要とするテンプレートファイルを含む **DEPLOYMENT** ディレクトリがあります。これらのテンプレートファイルは変更しないでください。**! SASROOT** ディレクトリには、sassetup ユーティリティによって使用されるプログラムスクリプトが格納されている **SASSETUP** ディレクトリもあります。
nls	各国語およびローケルサポート用のサブディレクトリを含みます。このディレクトリは **DBCS**(全角文字セット)、**DE** (–LOCALE ドイツ語)、**EN** (–LOCALE 英語(アメリカ))、**ES** (–LOCALE スペイン語)、**FR** (–LOCALE フランス語)、**HU** (–LOCALE ハンガリー語)、**IT** (–LOCALE イタリア語)、**JA** (–LOCALE 日本語−プライマリエンコディング)、**JA.SJIS** (–LOCALE 日本語−セカンダリエンコディング)、**KO** (–LOCALE 韓国語)、**NO** (–LOCALE ノルウェー語)、**PB** (–LOCALE pt_BR—ポルトガル語(ブラジル))、**PL** (–LOCALE ポーランド語)、**RU** (–LOCALE ロシア語)、**SV** (–LOCALE スウェーデン語)、**U8** (–LOCALE 英語(アメリカ)UTF8 サポート)、**ZH** (–LOCALE zh_CN—中国語)、および **ZT** (SAS でサポートされる繁体字のサブフォルダ)を含みます。この中のほとんどのフォルダは、言語固有の呼び出しスクリプトで SAS が起動された時、NLS 固有の内容を SAS で利用可能にする sasv9.cfg 構成ファイルを含みます。各言語ディレクトリは、インストール時に生成された SAS レジストリおよび SAS デスクトップデータセットが格納される **SASCFG** サブディレクトリを含みます。次のリストは、各国語サポート用の SAS **NLS** サブディレクトリのスナップショットです。追加のロケールがサポートされるようになった時点でこのリストは更新されます。

SAS サブディレクトリ	内容の説明
perl	sassetup プログラムおよび SAS 機能テストツールによって使用される Perl バイナリおよびライブラリを含みます。
samples	さまざまな SAS 製品のサンプルプログラムを含みます。これらのプログラムは、製品のサブディレクトリによって整理され、一部の SAS 製品のサンプルを含まない場合があります。
sasautos	事前定義の SAS マクロを含みます。“UNIX 環境で自動呼び出しライブラリを使用する” (298 ページ)を参照してください。
sasexe	各種 SAS 製品の実行可能なファイルを含みます。
sashelp	オンラインヘルプファイル、メニュー、グラフデバイスの説明およびウィンドウをサポートする SAS プロシジャが使用するカタログを含む SAS ライブラリです。
sasmsg	SAS によって使用されるすべてのメッセージおよび NOTE が格納されたファイルを含みます。
saspgm	SAS 製品の各種コンポーネントを含みます。
sastest	SAS 機能テストツールによって使用されるファイルを含みます。
utilities	man ページおよびユーティリティプログラムを含みます。詳細については、“UNIX 環境の Utilities ディレクトリ” (449 ページ)を参照してください。
X11	X Window System で SAS を実行するときに必要なファイルを含みます。bitmap ファイル、オンラインヘルプファイルおよびリソースファイルを含みます。

付録 2

システム管理者用ツール

UNIX 環境の Utilities ディレクトリ

!SASROOT/utilities ディレクトリは次の重要なサブディレクトリを含みます。

man
SAS のオンラインマニュアルページを含みます。“マニュアルページのインストール” (449 ページ)ではこれらのページを UNIX**man** コマンドを通してどのようにユーザーにアクセスさせるかを説明しています。

bin
管理ツールの実行可能ファイルを含みます。“/utilities/bin ディレクトリのユーティリティ” (450 ページ)ではこのディレクトリにあるツールのいくつかを説明します。

src/auth
ソースファイルおよび UNIX 認証 API のドキュメントを含みます。API を使用して、管理者は、カスタム認証方法を UNIX 環境での SAS 認証に追加できます。詳細については、“UNIX Authentication API” (450 ページ)を参照してください。

マニュアルページのインストール

utilities/man ディレクトリでのマニュアルページを読むためには、システムの他のマニュアルファイルの場所の **man1** サブディレクトリに、このファイルをコピーします。この場所は通常**/usr/man** または**/usr/local/man** です。UNIX **man man** コマンドを実行して、システムの適切なパス名を特定します。正しいパス名を確認し、次のコマンドを使用して SAS マニュアルファイルをコピーします。

```
cp -r sasroot/utilities/man/* pathname
```

pathname はシステムマニュアルファイルのディレクトリ場所です。

たとえば、次のコマンドを使用して!SASROOT ディレクトリからユーザーのシステムの man ディレクトリの man1 ファイルへ SAS マニュアルファイルをコピーすることで、オンラインヘルプにアクセスすることが可能です。

```
cp /usr/local/SASHome/SASFoundation/9.4/utilities/man/* /usr/local/man/man1
```

このコマンドを発行した後、man sas コマンドを使用してオンラインヘルプにアクセスできます。

使用するシステムの MANPATH 環境変数がすでに定義されている場合、または独自の MANPATH 環境変数を設定できる場合は、そのディレクトリをシステムの MANPATH 環境変数に追加することもできます。

UNIX Authentication API

UNIX Authentication Application Programming Interface (API)は、UNIX 環境で稼働する SAS にユーザー認証、ID、および権限の検証を提供する事前定義されたルーチンのセットです。ソースファイルにより、認証、ID、権限の検証プロセスにサイト固有動作を追加する機能が提供されます。

!SASROOT/utilities/src/auth/docs.pdf ファイルには、カスタム認証の実装方法と API 自体のドキュメントが含まれています。また、このドキュメントには、SAS ユーザー認証/ID と動作環境が提供する認証機能との統合方法についても説明されています。カスタム動作を実装する必要のある管理者はこのファイルを読んで、その指示に従ってください。

/utilities/bin ディレクトリのユーティリティ

次の表は、/utilities/bin ディレクトリのいくつかのツールを簡単に説明しています。これらのユーティリティに関する情報は、UNIX man コマンドを使用して得ることができます。

表 A2.1 *システム管理者用ツール*

ツール名	説明
authcustom.so	サイト固有認証用 sasauth モジュール
authldap.so	LDAP 認証用 sasauth モジュール
authpam.so	PAM 認証用 sasauth モジュール
bdm	バッチドライバモニタ
cfgpeh	スタンドアロンのスクランブルコマンド
cleanwork	関連する SAS プロセスは終了したのに残存している Work ディレクトリや Utility ディレクトリ、またはその両方を削除するツール (“cleanwork コマンド” (452 ページ)を参照)
docsetup	インストーラが起動するドキュメント設定ユーティリティ

ツール名	説明
elsconf	ELS 構成をチェックするツール
elssrv	ELS サーバー。サブプロセスを起動するツール
jproxy	SAS 内で Java ファシリティを起動するのに使用されるツール
ke2j	2 バイト入力ユーティリティ
ke2s	2 バイト入力ユーティリティ
kj2e	2 バイト入力ユーティリティ
ks2e	2 バイト入力ユーティリティ
loadmgr	アプリケーションロードマネージャ
motifxsassm	Motif X セッションマネージャ
objspawn	オブジェクトスポーナー
patchname	指定された実行可能ファイルにある **SASROOT** ディレクトリ名のリセット
rbrowser	プラットフォーム用リモートブラウザサーバー(Linux のみでサポートされていますが、他のすべての Motif プラットフォームでも動作します)
reshelper	X Windows 用リソースヘルパ
sasauth	ユーザー ID および認証ユーティリティ
sasauth.conf	sasauth の構成ファイル。使用される認証モジュールおよび他のオプションを指定します
sasm.elm.mime	SAS からの電子メールをサポートするスクリプト
sasmailer	SAS からの電子メールをサポートするスクリプト
sasperm	ユーザー許可ユーティリティ
sastcpd	TCP/IP アクセスデーモン
sasumgmt	ユーザー名およびパスワードを取得し、ユニコードにトランスコードまたはデコードします。そして SAS 認証サービスをコールしてユーザー認証をします。そして認証が正常に終了したか失敗したかを示す終了ステータスと共に終了します。
saswujms	日本語入力サーバー
setuid	ディレクトリ

ツール名	説明
setuid.sh	一部のコマンドを、root として実行されるように設定するスクリプト
tkdef.so	SASROOT および TKPATH に SAS 9.4 のパッチが適用される場所 (他の実行可能ファイルは SAS 9.2 であるためパッチ対象ではありません)

cleanwork コマンド

関連する SAS プロセスは終了したのに残存している Work ディレクトリ、Utility ディレクトリ、またはその両方を削除します。

cleanwork *directory<-n, -hostmatch>*

directory
Work ディレクトリ、**Utility** ディレクトリ、または両方のディレクトリを含むディレクトリを指定します。cleanwork コマンドには、複数のディレクトリパスを指定できます。ディレクトリ名は、WORK システムオプションに指定されている値、または UTILLOC システムオプションで指定した値と一致している必要があります。

ヒント cleanwork コマンドが root で実行されていない場合、ユーザー許可はディレクトリを削除することを認めない可能性があります。

-n
削除可能なエントリを含むディレクトリを SAS がリストする指定です。

-hostmatch
Network File System (NFS)にて有効となっている可能性がある Work ディレクトリを削除することのできるホスト名を指定します。

詳細

cleanwork コマンドは UTILLOC システムオプションにて割り当てられた作業ライブラリまたはディレクトリに割り当てられたあらゆるサブディレクトリを削除します。cleanwork は機能していない SAS プロセスに関連したファイルのみを削除します。各サブディレクトリ名の形式は次のようになります。

`SAS_workcode_nodename`

`SAS_utilcode_nodename`

コード
12 文字コードです。最初の 4 文字はランダムに生成された番号です。次の8文字は関連する SAS プロセスのプロセス ID の 16 進数表現を基にしています。有効プロセスに関連するファイルは削除されません。

nodename
SAS プロセスが実行されている UNIX システムの名前を指定します。

たとえば、nodename*jupiter* にて作業している場合、cleanwork コマンドは *jupiter* の無効プロセスを持つ全てのディレクトリを削除します。cleanwork は、プロセスが有効である場合にリンクされていないそのプロセスに関連するディレクトリを削除しません。この場合、プロセスを手動で終了し cleanwork を再度実行する必要があります。

関連項目:

"Work ライブラリ" (58 ページ)

付録 3

テキスト編集コマンド

テキスト編集コマンド

テキスト編集に固有のコマンドは、ウィンドウで編集機能を実行するためテキスト編集コマンドと呼びます。テキスト編集コマンドは次の 2 種類に分類されます。

- 行コマンド
- コマンド行コマンド

ほとんどの行コマンドは、テキストを再配置または再フォーマットを行います。行コマンドは、テキスト行またはテキストブロックの移動、削除、コピー、配置などのタスクを実行します。コマンド行コマンドは、テキストの再配置および再フォーマットだけでなく、コマンド効果の切り替えやテキストの大文字小文字のデフォルトの設定の変更などといったタスクも実行します。

この章では、UNIX 版に固有ではないが UNIX 環境で使用できるコマンドについて説明します。これらのコマンドは、テキスト編集コマンドに対応しているあらゆる動作環境で使用できます。

ディクショナリ

AUTOADD コマンド

自動行追加を管理します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

AUTOADD <ON | OFF>

引数なし

AUTOADD コマンドを ON または OFF に切り替えます。このコマンドを発行すると現在の設定が切り替わります。現在の設定が ON のときに AUTOADD コマンドを発行すると、設定が OFF に変わります。現在の設定が OFF のときに AUTOADD コマンドを発行すると、設定が ON に変わります。

必須引数

ON
ウィンドウの AUTOADD コマンドが有効になり、行が自動追加されます。

OFF
AUTOADD コマンドが無効になり、行は自動追加されません。

詳細

AUTOADD コマンドは、スクロール時に既存テキストを通過するときに空白行を追加するのかを管理します。追加された行数は VSCROLL コマンドの設定により決まります。VSCROLL コマンドはデフォルトの前後スクロール幅を決定するコマンドです。

関連項目:

コマンド:

- “AUTOFLOW コマンド” (457 ページ)
- “AUTOSPLIT コマンド” (458 ページ)
- “AUTOWRAP コマンド” (459 ページ)

AUTOFLOW コマンド

テキストを挿入、コピー、貼り付ける場合に、自動的に整列するかどうかを管理します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

AUTOFLOW <ON | OFF>

引数なし

AUTOFLOW コマンドを切り替えます。このコマンドを発行すると現在の設定が切り替わります。現在の設定が ON のときに AUTOFLOW コマンドを発行すると、設定が OFF に変わります。現在の設定が OFF のときに AUTOFLOW コマンドを発行すると、設定が OFF に変わります。

必須引数

ON
ウィンドウの AUTOFLOW コマンドが有効になり、ウィンドウにテキストが挿入されるとそのテキストが自動的に整列されます。

OFF
ウィンドウの AUTOFLOW コマンドが無効になり、テキストがウィンドウに挿入されるとき以前の位置を保持します。

詳細

AUTOFLOW コマンドは、INCLUDE、PASTE または COPY コマンドとともに挿入されるテキストが自動的に整列するかどうかを管理します。テキストが整列されるとき、前回の INDENT コマンドと BOUNDS コマンドの実行時に指定された設定により左右の境界が決まります。AUTOFLOW コマンドは、ウィンドウに挿入されるすべてのテキストを管理します。パラグラフ境界で中止されません。

比較

AUTOFLOW コマンドは、ウィンドウに挿入されるテキストを整列するかどうかを制御します。TF コマンドはすでにウィンドウ上に表示されているテキストを整列します。

関連項目:

コマンド:

- “AUTOSPLIT コマンド” (458 ページ)
- “AUTOWRAP コマンド” (459 ページ)
- “BOUNDS コマンド” (460 ページ)
- “INDENT コマンド” (471 ページ)
- “TF コマンド” (487 ページ)

AUTOSCROLL コマンド

出力を表示するために Log と Output ウィンドウをスクロールする頻度を指定します。

UNIX 固有: 有効な引数およびデフォルト値

構文

AUTOSCROLL <*n*>

オプション引数

n
ウィンドウに収まらないデータ行を受け入れるときにスクロールする行数を指定します。

詳細

AUTOSCROLL コマンドは、Log ウィンドウと Output ウィンドウに書き込まれるときの行のスクロールを管理します。Log ウィンドウと Output ウィンドウでの AUTOSCROLL のデフォルト値は 1 です。AUTOSCROLL が一度に 1 行ずつ表示する場合は処理が遅くなります。処理を迅速にするには、autoexec.sas ファイルの AUTOSCROLL 値をより大きく指定します。0 の値を指定すると処理が最適化され、スクロールが最速になります(xterm ウィンドウのジャンプスクロールと同様)。AUTOSCROLL コマンドを autoexec.sas ファイルに追加するには、DM コマンドを使用する必要があります。次の例は、Log ウィンドウと Output ウィンドウでのスクロール幅を最大化したものを示しています。

```
dm 'output; autoscroll 0; log; autoscroll 0; pgm;';
```

AUTOSPLIT コマンド

ENTER が押された場合やキャリッジリターンが入力された場合に、カーソル位置でテキストを分割するかどうかを制御します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

AUTOSPLIT <ON | OFF>

引数なし

AUTOSPLIT コマンドの動作を切り替えます。AUTOSPLIT コマンドの最初の発行時に、現在の設定が切り替わります。現在の設定が ON のとき AUTOSPLIT コマンドを発行すると、設定が OFF に変わります。現在の設定が OFF のとき AUTOSPLIT コマンドを発行すると、設定が ON に変わります。

オプション引数

ON

AUTOSPLIT コマンドが有効なウィンドウでは、ENTER が押された場合やキャリッジリターンが入力された場合に、自動的にカーソル位置でテキストが分割されます。

OFF

AUTOSPLIT コマンドが無効なウィンドウでは、ENTER が押された場合やキャリッジリターンが入力された場合に、カーソル位置でテキストは自動的には分割されません。

詳細

AUTOSPLIT コマンドは、ENTER が押された場合やキャリッジリターンが入力された場合に、カーソル位置でテキストを分割するかどうかを制御します。カーソル位置以降にある行のテキストはすべて、次の行の左境界に移動します。カーソル位置は、新しい行の最初の文字に置かれます。

比較

AUTOSPLIT コマンドの有効時にキャリッジリターンを入力することと、TS コマンドをデフォルトの数値引数 1 で発行することは同じです。AUTOSPLIT コマンドの有効時のキャリッジリターンの結果は、TC コマンドで置き換えるか、UNDO コマンドで元に戻すことができます。

関連項目:

コマンド:

- “AUTOSCROLL コマンド” (458 ページ)
- “AUTOWRAP コマンド” (459 ページ)
- “TF コマンド” (487 ページ)
- “TS コマンド” (488 ページ)

AUTOWRAP コマンド

テキストを挿入、コピーまたは書き出す場合に折り返すかどうかを管理します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

AUTOWRAP <ON | OFF>

引数なし

AUTOWRAP コマンドを切り替えます。最初の AUTOWRAP コマンドの発行時に、現在の設定が切り替わります。現在の設定が ON のときに AUTOWRAP コマンドを発行すると、設定が OFF に変わります。現在の設定が OFF のときに AUTOWRAP コマンドを発行すると、設定が ON に変わります。

オプション引数

ON

ウィンドウの AUTOWRAP コマンドが有効になり、テキストがウィンドウに挿入されたときまたは外部ファイルに移動されたとき、折り返されます。

OFF

ウィンドウの AUTOWRAP コマンドが無効になります。行の長さによっては、テキストをウィンドウに挿入したときまたは外部ファイルに移動したときに切り捨てられる可能性があります。

詳細

AUTOWRAP コマンドを有効にすると、INCLUDE コマンドまたは COPY コマンドを使用できます。これらコマンドにより、ウィンドウの境界を越える行の長さを含むファイルをウィンドウに挿入することができます。ファイル内のテキストは切り捨てられません。ただし、ファイル内の行は単語の境界で分割されます。また、AUTOWRAP コマンドにより、FILE コマンドを使用して、行の長さがファイルの境界を越えるテキストを外部ファイルに送れます。ファイル内のテキストは切り捨てられません。行は単語の境界で分割されます。AUTOWRAP コマンドを無効にすると、テキスト行の長さおよびウィンドウまたはファイルの行の長さにより、テキストが切り捨てられることがあります。

関連項目:

コマンド:

- “AUTOFLOW コマンド” (457 ページ)
- “AUTOSPLIT コマンド” (458 ページ)

BOUNDS コマンド

テキストが整列されるときの左右の境界を設定します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

BOUNDS *<left right>*

引数なし

BOUNDS コマンドは、現在の境界設定を示すメッセージを表示します。

オプション引数

left
左の境界のカラム位置を設定します。

right
右の境界のカラム位置を設定します。

詳細

BOUNDS コマンドによりテキストの左右境界がリセットされます。テキストのカラム位置がリセットされます。テキストはすでにウィンドウ内にあり、TF コマンドを使用して整列する必要があります。BOUNDS コマンドは、AUTOFLOW コマンドの有効時に INCLUDE、COPY および PASTE コマンドによりウィンドウ内に挿入されるテキストの左右境界を設定します。AUTOFLOW コマンドの有効時、TS コマンドによるテキストの分割の際にこの左境界の設定が維持されます。

たとえば、カラム 10 から 60 まで間にテキストを整列する場合は、次のコマンドを指定します。

```
bounds 10 60
```

この BOUNDS コマンドの発行後にテキストが整列される際、別の BOUNDS コマンドが発行されるか、INDENT コマンドが ON に設定されるまで、カラム 10 から 60 までの間に配置されます。

INDENT コマンドを ON に設定すると、左境界の現在の設定より優先されます。左境界の設定を確実に使用するには、INDENT コマンドを OFF に設定してください。

比較

BOUNDS コマンドは TF コマンドの動作と TS コマンドの動作に影響します。BOUNDS コマンドは、INDENT コマンドと同様、左境界を設定できます。ただし、BOUNDS コマンドは右境界も設定できます。テキストの整列時、INDENT コマンドを ON に設定すると常に左境界が設定されますが、この左境界は BOUNDS コマンドで設定する左境界より優先されます。

関連項目:

コマンド:

- “AUTOFLOW コマンド” (457 ページ)
- “INDENT コマンド” (471 ページ)
- “TF コマンド” (487 ページ)
- “TS コマンド” (488 ページ)

C コマンド

1 つのテキスト行をコピーします。

カテゴリ: テキスト編集、行コマンド

構文

C

intervening text

A | B

引数なし

C コマンドは、ウィンドウ内の新しい位置にテキスト行を 1 つコピーします。

オプション引数

A
コピーするテキスト行のコピー先をマークします。この場合は、A 引数を入力した行の後ろになります。コピーする行の前にも後にも A 引数を指定できます。

B
コピーするテキスト行のコピー先をマークします。この場合は、B 引数を入力した行の前になります。コピーする行の前にも後にも B 引数を指定できます。

比較

C コマンドと CC コマンドでは、ウィンドウ内の任意の場所をテキスト行のコピー先として指定できます。R コマンドと RR コマンドは、このコマンドが指定された行のすぐ後ろに行やテキスト行のブロックを繰り返します。

関連項目:

コマンド:

- “CC コマンド” (463 ページ)
- “R コマンド” (481 ページ)
- “RR コマンド” (482 ページ)

CAPS コマンド

テキストの大文字小文字のデフォルトの設定を変更します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

CAPS <ON | OFF>

引数なし

CAPS コマンドを切り替えます。最初の CAPS コマンドの発行時に、現在の設定が切り替わります。現在の設定が ON のときに CAPS コマンドを発行すると、設定が OFF に変わります。現在の設定が OFF のときに CAPS コマンドを発行すると、設定が ON に変わります。

必須引数

ON
CAPS コマンドを切り替えます。CAPS コマンドを有効にした後で入力する文字は大文字されます。FIND コマンドと CHANGE コマンドの文字列も、引用符内に囲まれていない場合は、全角に大文字されます。

OFF
CAPS コマンドを無効にします。CAPS コマンドを無効にした後で入力する文字の大文字小文字は変更されません。

詳細

CAPS コマンドは、これから入力するテキストまたはウィンドウ内で変更したテキストの大文字小文字を変更します。CAPS ON を指定してテキストを入力する場合、ENTER を押すと即座にそのテキストは大文字に変更されます。SAS セッションを終了するか、または別の CAPS コマンドで設定を変更するまでは、設定はウィンドウに対して有効です。WSAVE コマンドを使うと、現在の SAS セッションの域を超えて CAP コマンドの設定を保存できます。

比較

CAPS ON コマンドは、既存のテキストの大文字小文字を変更する CU コマンドと CCU コマンド、CL コマンドと CCL コマンドに類似しています。ただし CAPS コマンドは、既存のテキストの文字設定ではなく、テキストの大文字小文字のデフォルトの設定を変更するものです。

関連項目:

コマンド:

- “CL コマンド” (465 ページ)
- “CCL コマンド” (464 ページ)
- “CU コマンド” (466 ページ)
- “CCU コマンド” (465 ページ)

CC コマンド

テキスト行のブロックをコピーします。

カテゴリ: テキスト編集、行コマンド

構文

CC

block of text

CC

intervening text

A | B

引数なし

CC コマンドは、ウィンドウ内の新しい位置にテキスト行のブロックをコピーします。

必須引数

A
コピーするテキスト行のコピー先をマークします。この場合は、A 引数を入力した行の後ろになります。コピーする行の前にも後にも A 引数を指定できます。

B
コピーするテキスト行のコピー先をマークします。この場合は、B 引数を入力した行の前になります。コピーする行の前にも後にも B 引数を指定できます。

詳細

C コマンドと CC コマンドでは、ウィンドウ内の任意の場所をテキスト行のコピー先として指定できます。R コマンドと RR コマンドは、このコマンドが指定された行のすぐ後ろに行やテキスト行のブロックを繰り返します。

関連項目:

コマンド:

- “C コマンド” (461 ページ)
- “R コマンド” (481 ページ)
- “RR コマンド” (482 ページ)

CCL コマンド

指定のテキスト行の文字をすべて小文字に変更します。

カテゴリ: テキスト編集、行コマンド

構文

CCL

block of text

CCL

引数なし

CCL コマンドは、テキスト行のブロックの文字をすべて小文字に変更します。

詳細

CL コマンドと CCL コマンドは、既存のテキストを小文字に変更します。CAPS OFF コマンドはテキストのデフォルト設定を小文字にし、新しく挿入されたテキストの大文字小文字を変更します。CU コマンドと CCU コマンドは、既存のテキストを大文字に変更するもので、CL コマンドと CCL コマンドとは逆の変更を行います。

関連項目:

コマンド:

- “CL コマンド” (465 ページ)

- “CAPS コマンド” (462 ページ)
- “CU コマンド” (466 ページ)
- “CCU コマンド” (465 ページ)

CCU コマンド

指定のテキスト行ブロックの文字をすべて大文字に変更します。

カテゴリ: テキスト編集、行コマンド

構文

CCU

block of text

CCU

引数なし
CCU コマンドは指定したテキスト行のブロックの文字をすべて大文字に変更します。

詳細

CU コマンドと CCU コマンドは、CAPS ON コマンドに類似しています。CU コマンドと CCU コマンドは、既存のテキストを大文字に変更します。CAPS ON コマンドではテキストのデフォルト設定を大文字にし、新しく挿入されたテキストの大文字小文字を変更します。CL コマンドと CCL コマンドは、既存テキストを小文字に変更するもので、CU コマンドと CCU コマンドとは逆の変更を行います。

関連項目:

コマンド:

- “CU コマンド” (466 ページ)
- “CAPS コマンド” (462 ページ)
- “CL コマンド” (465 ページ)
- “CCL コマンド” (464 ページ)

CL コマンド

指定のテキスト行の文字をすべて小文字に変更します。

カテゴリ: テキスト編集、行コマンド

構文

CL *<n>*

引数なし

CL コマンドは、指定のテキスト行の文字をすべて小文字に変更します。

必須引数

n

小文字に変更するテキストの行数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

CL コマンドと CCL コマンドは、既存のテキストを小文字に変更します。CAPS OFF コマンドではテキストのデフォルト設定を小文字にします。新しく挿入されたテキストの大文字小文字が変更されます。CU コマンドと CCU コマンドは、既存のテキストを大文字に変更するもので、CL コマンドと CCL コマンドとは逆の変更を行います。

関連項目:

コマンド:

- “CCL コマンド” (464 ページ)
- “CAPS コマンド” (462 ページ)
- “CU コマンド” (466 ページ)
- “CCU コマンド” (465 ページ)

CU コマンド

指定のテキスト行の文字をすべて大文字に変更します。

カテゴリ: テキスト編集、行コマンド

構文

CU <*n*>

引数なし

CU コマンドは、指定のテキスト行の文字をすべて大文字に変更します。

オプション引数

n

大文字に変更するテキストの行数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

CU コマンドと CCU コマンドは、CAPS ON コマンドに類似しています。CU コマンドと CCU コマンドは、既存のテキストを大文字に変更します。CAPS ON コマンドではテキストのデフォルト設定を大文字にし、新しく挿入されたテキストの大文字小文字を変更します。CL コマンドと CCL コマンドは、既存テキストを小文字に変更するもので、CU コマンドと CCU コマンドとは逆の変更を行います。

関連項目:

コマンド:

- “CAPS コマンド” (462 ページ)
- “CCU コマンド” (465 ページ)
- “CL コマンド” (465 ページ)
- “CCL コマンド” (464 ページ)

CURSOR コマンド

コマンド行にカーソルを移動します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

CURSOR

引数なし

CURSOR コマンドはカーソルをコマンド行に移動させるものです。CURSOR コマンドは、ファンクションキーで実行する設計となっています。

詳細

CURSOR コマンドは HOME キーとほぼ同じに使用できます。

比較

CURSOR コマンドは HOME キーを押した場合と同様の結果が得られます。

D コマンド

指定した行を削除します。

カテゴリ: テキスト編集、行コマンド

構文

D <*n*>

引数なし

D コマンドは指定した行のみを削除します。

必須引数

n

削除する行数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。

関連項目:

コマンド:

- “DD コマンド” (468 ページ)

DD コマンド

指定した行ブロックを削除します。

カテゴリ: テキスト編集、行コマンド

構文

DD

block of lines

DD

引数なし

DD コマンドはテキスト行のブロックを削除します。

関連項目:

コマンド:

- “D コマンド” (467 ページ)

DICT コマンド

補助辞書を挿入、解放、作成します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

DICT INCLUDE *dictionary-name* | FREE *dictionary-name* | CREATE *dictionary-name <size>*

必須引数

INCLUDE *dictionary-name*
指定した補助辞書を現在の SAS セクションで使用できるようにします。1 レベル名のみが有効です。補助辞書の検索は、まず SASUSER.PROFILE カタログから行われます。その後、SASHELP.BASE カタログが検索されます。補助辞書が見つからない場合は、エラーメッセージが発行されます。SASHELP.BASE カタログの補助辞書の使用時、補助辞書への変更は保存されません。SASUSER.PROFILE カタログの補助辞書の使用時、補助辞書への変更は保存されます。

FREE *dictionary-name*
指定した補助辞書を解放します。FREE 引数を指定した DICT コマンドを発行するか、または現在の対話型ウィンドウタスクを終了するまでは、新規作成された辞書

は SASUSER.PROFILE カタログに保存されません。補助辞書に変更を加えた場合は、FREE 引数を指定した DICT コマンドを発行すると変更が保存されます。SASHELP.BASE カタログの補助辞書の使用時、これら変更は保存されません。

CREATE ***dictionary-name***
指定した補助辞書を新規作成します。補助辞書は最初は空の状態です。補助辞書は解放時に SASUSER.PROFILE カタログに保存されます。1 レベル名のみが有効です。

size
補助辞書のサイズ(バイト)を指定します。デフォルト値は 9,808 バイトです。

詳細

DICT コマンドは、補助辞書を挿入、解放、作成します。SPELL コマンドはスペルチェックを行い、認識されなかった単語にフラグを付けます。さらに、SPELL コマンドでは、辞書の作成と更新が可能です。

関連項目:

コマンド:

- “SPELL コマンド” (485 ページ)

FILL コマンド

fill-character を現在のカーソル位置から挿入します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

FILL <*'fill-character'*> <*n*>

引数なし

FILL コマンドは、fill-character とその反復回数を示すメッセージを表示します。

オプション引数

'fill-character'
ユーザー定義の fill-character を指定します。一重引用符で囲む必要があります。この反復文字は変更を加えるまで有効です。

n
fill-character の正確な反復回数を指定します。この文字数は変更を加えるまで有効です。

詳細

FILL コマンドは、現在のカーソル位置から fill-character を挿入します。fill-character は、行の終わりまでか、または次の空白でない文字の前のスペースまで表示されます。デフォルトでは、fill-character は普通、アンダースコアまたはハイフンです。FILL 引数を使用すると、fill-character とその繰り返し回数を変更できます。

FILL コマンドの発行は、ファンクションキーで行うのが最も簡単です。fill-character をカーソル位置に挿入するには、ファンクションキーの1つを FILL コマンドの発行に設定します。カーソルをプログラムエディタフィールドに移動させて、設定したファンクションキーを押します。fill-character が表示されます。

ここで、デフォルトの変更方法の例を示します。次のコマンドを発行すると、デフォルトは疑問符 10 個になります。

```
fill '?' 10
```

変更した fill-character は、SAS セッション中またはこの fill-character を変更するまで有効です。WSAVE コマンドを使用すると、この設定を永久保存することができます。

I コマンド

1 つ以上の空白行を挿入します。

カテゴリ: テキスト編集、行コマンド

構文

I <A | B> <*n*>

引数なし

I コマンドは、このコマンドを発行した行の直後に 1 つ以上の空白行を挿入します。

オプション引数

A
このコマンドを発行した行の直後に 1 つ以上の空白行を挿入します。I コマンドと A 引数の間には文字を指定できません。

B
このコマンドを発行した行の直前に 1 つ以上の空白行を挿入します。I コマンドと B 引数の間には文字を指定できません。

n
挿入する空白行の数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。A 引数または B 引数を使用する場合、*n* 引数は最後に指定します。たとえば、行 00009 には、PROC PRINT ステートメントが含まれています。次の I コマンドは、そのテキスト行の前に 3 つの空白行を挿入します。

```
ib3 9 proc print data=final.educ;
```

詳細

I コマンドは、1つ以上の空白行を挿入します。デフォルトでは、行は空白です。MASK コマンドでコンテンツを定義できます。I コマンドの発行はファンクションキーで行うのが最も簡単です。

比較

MASK コマンドは I コマンドとともに使用できます。I コマンドは1つ以上の空白行を挿入しますが、これは MASK コマンドが設定したコンテンツを含めることができます。

関連項目:

コマンド:

- “MASK コマンド” (479 ページ)

INDENT コマンド

テキストの整列時に左境界のインデントを保持します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

INDENT <ON | OFF>

引数なし

INDENT コマンドを切り替えます。INDENT コマンドの初回発行時に、現在の設定が切り替わります。現在の設定が ON のときに INDENT コマンドを発行すると、設定が OFF に変わります。現在の設定が OFF のときに INDENT コマンドを発行すると、設定が ON に変わります。INDENT コマンドを再発行すると、以前の設定に戻ります。

オプション引数

ON
ウィンドウの INDENT コマンドを有効にします。

ヒント INDENT ON コマンドはすべての行をインデントします。

INDENT コマンドを有効にして TF コマンドを発行すると、パラグラフの全行が、そのパラグラフの最初の行と同じインデントが行われます。

OFF
ウィンドウの INDENT コマンドを無効にします

詳細

次の場合に、INDENT コマンドは現在の左境界インデントを使用するように指定します。

- ウィンドウの既存テキストが TF コマンドにより整列される場合
- AUTOFLOW コマンドの有効時にテキストをウィンドウに挿入する場合
- ウィンドウの既存テキストが TS コマンドにより分割される場合

比較

左境界は、INDENT コマンドと BOUNDS コマンドにより設定できます。ただし、テキストの整列時、INDENT コマンドを有効にすると必ず左境界が設定され、BOUNDS コマンドで設定された左境界より優先されます。

例

例 1

この例では 4 行のテキスト行が示されています。TF コマンドは最初の行の番号フィールドに入力されています。パラグラフの最初の行がインデントされています。INDENT コマンドは ON に設定されていて、デフォルトの境界は 1 および 50 です。

```
tf 01 The purpose of Monday's meeting is to review
00002 the documentation plan and gather your responses. Please
00003 send a representative
00004 if you are unable to attend.
```

例 2

次の例は、ENTER を押して TF コマンドを発行した場合の結果を示しています。すべての行にインデントが適用されています。右境界は 50 です。

```
tf 01 The purpose of Monday's meeting is to review
00002 the documentation plan and gather your responses. Please
00003 send a representative
00004 if you are unable to attend.
```

関連項目:

コマンド:

- “AUTOFLOW コマンド” (457 ページ)
- “BOUNDS コマンド” (460 ページ)
- “TF コマンド” (487 ページ)
- “TS コマンド” (488 ページ)

JC コマンド

指定したテキスト行を中央揃えにします。

カテゴリ: テキスト編集、行コマンド

構文

JC <*n*>

引数なし

JC コマンドは、左右境界の設定基づいて、この行コマンドを指定したテキスト行を中央揃えにします。

オプション引数

n

指定したテキスト行を中央揃えにする場合のカラム位置を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

JC コマンドは指定したテキスト行を中央揃えします。数値引数を指定しない場合は、中央揃えは BOUDNS コマンドにより設定された現在の境界設定に基づきます。数値引数は、これらの境界設定より優先されます。

比較

JL コマンド、JJL コマンド、JR コマンド、JJR コマンドと同様、JC コマンドおよび JJC コマンドはテキストを配置します。

関連項目:

コマンド:

- “JJC コマンド” (473 ページ)
- “BOUNDS コマンド” (460 ページ)
- “JL コマンド” (476 ページ)
- “JJL コマンド” (474 ページ)
- “JR コマンド” (477 ページ)
- “JJR コマンド” (475 ページ)

JJC コマンド

指定したテキストブロックの各テキスト行をそれぞれ中央揃えにします。

カテゴリ: テキスト編集、行コマンド

構文

JJC

block-of-text

JJC

オプション引数

block-of-text
中央揃えにするテキストブロックを指定します。

詳細

JJC コマンドは指定したテキストブロックを中央揃えにします。このブロックの各行はそれぞれ中央揃えにされます。中央揃えは、BOUNDS コマンドで設定された現在の境界設定に基づいて行われます。

比較

JL コマンド、JJL コマンド、JR コマンド、JJR コマンドと同様、JC コマンドおよび JJC コマンドはテキストを配置します。

関連項目:

コマンド:

- “JC コマンド” (472 ページ)
- “BOUNDS コマンド” (460 ページ)
- “JL コマンド” (476 ページ)
- “JJL コマンド” (474 ページ)
- “JR コマンド” (477 ページ)
- “JJR コマンド” (475 ページ)

JJL コマンド

指定したテキストブロックを左揃えにします。

カテゴリ: テキスト編集、行コマンド

構文

JJL <*n*>

block-of-text

JJL <*n*>

引数なし

JJL コマンドは指定したテキストブロックを左揃えにします。配置は、左右境界の設定に基づいて行われます。

オプション引数

n

指定したテキストブロックを左揃えにする場合のカラム位置を指定します。デフォルトでは、*n* 引数は左境界設定です。*n* 引数の後に続けて、スペースを 1 つ挿入します。ブロックの範囲を示すコマンドの最初または最後の行、あるいはその両方に数値引数を指定できます。最初と最後の両方の行に引数を指定する場合は、最初の数値引数が使用されます。

block-of-text

左揃えにするテキストブロックを指定します。

詳細

JJL コマンドは指定したテキストブロックを左揃えにします。数値引数を指定しない場合、左揃えは、BOUDNS コマンドで設定された現在の境界設定に基づきます。数値引数は、これらの境界設定より優先されます。

比較

JC コマンド、JJC コマンド、JR コマンド、JJR コマンドと同様、JL コマンドおよび JJL コマンドはテキストを配置します。

関連項目:

コマンド:

- “JL コマンド” (476 ページ)
- “BOUNDS コマンド” (460 ページ)
- “JC コマンド” (472 ページ)
- “JJC コマンド” (473 ページ)
- “JR コマンド” (477 ページ)
- “JJR コマンド” (475 ページ)

JJR コマンド

指定したテキストブロックを右揃えにします。

カテゴリ: テキスト編集、行コマンド

構文

JJR <*n*>

block-of-text

JJR <*n*>

引数なし

JJR コマンドは指定したテキストブロックを右揃えにします。配置は、左右境界の設定に基づいて行われます。

オプション引数

n
指定したテキストブロックを右揃えにする場合のカラム位置を指定します。デフォルトでは、*n* 引数は右境界設定です。*n* 引数の後に続けて、スペースを 1 つ挿入します。ブロックの範囲を示すコマンドの最初または最後の行、あるいはその両方に数値引数を指定できます。最初と最後の両方の行に引数を指定する場合は、最初の数値引数が使用されます。

block-of-text
右揃えにするテキストブロックを指定します。

詳細

JJR コマンドは指定したテキストブロックを右揃えにします。数値引数を指定しない場合、右揃えは、BOUDNS コマンドで設定された現在の境界設定に基づきます。数値引数は、これらの境界設定より優先されます。

比較

JC コマンド、JJC コマンド、JL コマンド、JJL コマンドと同様、JR コマンドおよび JJR コマンドはテキストを配置します。

関連項目:

コマンド:

- “JR コマンド” (477 ページ)
- “BOUNDS コマンド” (460 ページ)
- “JC コマンド” (472 ページ)
- “JJC コマンド” (473 ページ)
- “JL コマンド” (476 ページ)
- “JJL コマンド” (474 ページ)

JL コマンド

指定したテキスト行を左揃えにします。

カテゴリ: テキスト編集、行コマンド

構文

JL <*n*>

引数なし

JL コマンドは指定したテキスト行を左揃えにします。配置は、左右境界の設定に基づいて行われます。

オプション引数

n

指定したテキスト行を左揃えにする場合のカラム位置を指定します。デフォルトでは、*n* 引数は左境界設定です。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

JL コマンドは指定したテキスト行を左揃えにします。数値引数を指定しない場合、左揃えは、BOUDNS コマンドで設定された現在の境界設定に基づきます。数値引数は、これらの境界設定より優先されます。

比較

JC コマンド、JJC コマンド、JR コマンド、JJR コマンドと同様、JL コマンドおよび JJL コマンドはテキストを配置します。

関連項目:

コマンド:

- “JJL コマンド” (474 ページ)
- “BOUNDS コマンド” (460 ページ)
- “JC コマンド” (472 ページ)

- “JJC コマンド” (473 ページ)
- “JR コマンド” (477 ページ)
- “JJR コマンド” (475 ページ)

JR コマンド

指定したテキスト行を右揃えにします。

カテゴリ: テキスト編集、行コマンド

構文

JR <*n*>

引数なし

JR コマンドは指定したテキスト行を右揃えにします。配置は、左右境界の設定に基づいて行われます。

オプション引数

n

指定したテキスト行を右揃えにする場合のカラム位置を指定します。デフォルトでは、*n* 引数は右境界設定です。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

JR コマンドは指定したテキスト行を右揃えにします。数値引数を指定しない場合、右揃えは、BOUDNS コマンドで設定された現在の境界設定に基づきます。数値引数は、これらの境界設定より優先されます。

比較

JC コマンド、JJC コマンド、JL コマンド、JJL コマンドと同様、JR コマンドおよび JJR コマンドはテキストを配置します。

関連項目:

コマンド:

- “JJR コマンド” (475 ページ)
- “BOUNDS コマンド” (460 ページ)
- “JC コマンド” (472 ページ)
- “JJC コマンド” (473 ページ)
- “JL コマンド” (476 ページ)
- “JJL コマンド” (474 ページ)

KEYS コマンド

ファンクションキーにタスクを割り当てることができます。

カテゴリ: テキスト編集、コマンド行コマンド

構文

KEYS

比較

KEYS コマンドにより、**KEYS** ウィンドウにアクセスして、タスクをファンクションキーに割り当てることができます。

M コマンド

テキスト行を 1 つ移動します。

カテゴリ: テキスト編集、行コマンド

構文

M <*n*>

intervening-text

A | B

オプション引数

n
移動するテキストの行数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。*n* 引数がない場合は、この行コマンドを含むテキスト行のみを移動します。

A
移動するテキスト行の移動先をマークします。この場合、A 引数を入力した行の後ろになります。移動する行の前にも後ろにも A 引数を指定できます。

B
移動するテキスト行の移動先をマークします。この場合は、B 引数を入力した行の前になります。移動する行の前にも後ろにも B 引数を指定できます。

詳細

M コマンドは、指定したテキスト行をウィンドウ内の新しい位置に移動します。

関連項目:

コマンド:

- “MM コマンド” (480 ページ)

MASK コマンド

新規行のコンテンツを定義します。

カテゴリ: テキスト編集、行コマンド

構文

MASK

引数なし

MASK コマンドは、I コマンドで作成される新規行のコンテンツを定義、表示および編集できるようにします。

詳細

MASK コマンドは、I コマンドで作成される新規行のコンテンツを定義、表示および編集できるようにします。新規行のデフォルトの設定は空白行です。新規行を表示または編集するには、行の番号フィールドに MASK と入力し、ENTER を押します。MASK コマンドで定義したコンテンツを含む行が挿入されます。これで、この行を編集できます。I コマンドを発行するたびに、MASK コマンドで定義したコンテンツを含む行が挿入されます。

MASK コマンドで定義したコンテンツについては、変更を加えない限り、現在の SAS セッション中はこのウィンドウに対し有効です。コンテンツを変更するには、テキストを上書きします。デフォルト(空白行)に戻す場合は、次のタスクのうち 1 つを実行します。

- MASK 行のテキストフィールドにあるすべての文字を消去する。
- CLEAR コマンドに MASK を引数として指定して発行する。

  ```
  clear mask
  ```

  MASK 行のコンテンツを消去すると、ログに MASK 行が消去されたことを示す Note が表示されます。

  RESET コマンドまたは D コマンドを使って、MASK コマンドのコンテンツを非表示にできます。非表示にしても MASK コマンドは有効です。たとえば、次の場合、MASK コマンドは有効です。

  - スクロール中に MASK 行を通り過ぎて表示されない場合
  - MASK 行のテキスト内の文字を空白にしないで D コマンドまたは RESET コマンドを発行する場合

  Program Editor ウィンドウなどでは、WSAVE コマンドを使って MASK コマンドのコンテンツを永久保存できます。

関連項目:

コマンド:

- “D コマンド” (467 ページ)
- “RESET コマンド” (482 ページ)

MM コマンド

テキストブロックを移動します。

カテゴリ: テキスト編集、行コマンド

構文

MM

block-of-text

MM

intervening-text

A | B

必須引数

A
移動するテキスト行の移動先をマークします。この場合、A 引数を入力した行の後ろになります。移動する行の前にも後ろにも A 引数を指定できます。

B
移動するテキスト行の移動先をマークします。この場合は、B 引数を入力した行の前になります。移動する行の前にも後ろにも B 引数を指定できます。

関連項目:

コマンド:

- “M コマンド” (478 ページ)

NUMBERS コマンド

行番号を追加または削除します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

NUMBERS <ON | OFF>

引数なし

NUMBERS コマンドを切り替えます。NUMBERS コマンドの初回発行時に、現在の設定が切り替わります。現在の設定が ON のときに NUMBERS コマンドを発行すると、設定が OFF に変わります。現在の設定が OFF のときに NUMBERS コマンドを発行すると、設定が ON に変わります。

オプション引数

ON
ウィンドウの NUMBERS コマンドが有効になり、プログラムエディタの行に番号が付きます。

OFF
ウィンドウの NUMBERS コマンドが無効になり、プログラムエディタの行には番号が付きません。

詳細

NUMBERS コマンドは、テキスト編集が可能なウィンドウのデータ行の行番号を追加または削除します。NUMBERS コマンドを発行して行番号を削除すると、対象の行番号は消えてテキスト全体が左にシフトします。NUMBERS コマンドを発行して行番号を追加すると、対象の行番号が左に表示されてテキスト全体が右にシフトします。NUMBERS コマンドのエイリアスは NUMS です。

R コマンド

指定した行を繰り返します。

カテゴリ: テキスト編集、行コマンド

構文

R <*n*>

引数なし

R コマンドは、指定した行を 1 回繰り返します。

オプション引数

n
指定した行を繰り返す回数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

R コマンドは、指定した行をその直後に繰り返します。デフォルトでは 1 回です。

比較

R コマンドと RR コマンドは、対象の行または行ブロックを、これらのコマンドを指定した行の直後に繰り返します。C コマンドと CC コマンドは、1 つ以上の行をウィンドウ内の任意の場所にコピーできます。

関連項目:

コマンド:

- “RR コマンド” (482 ページ)
- “C コマンド” (461 ページ)
- “CC コマンド” (463 ページ)

RESET コマンド

保留中の行コマンドを削除します。

カテゴリ: テキスト編集、コマンド行コマンド

構文

RESET

引数なし

RESET コマンドは保留中の行コマンドを削除します。

詳細

RESET コマンドは保留中の行コマンドを削除します。また、MASK コマンド発行時に作成された MASK 行も削除します。MASK コマンドの設定ではなく、MASK 行の表示が削除されます。行ブロックのコマンド(MM コマンドまたは CC コマンドなど)を中止する場合は、RESET コマンドを発行すれば保留中のコマンドを削除できます。

比較

RESET コマンドの結果は、D コマンドと同じです。RESET コマンドも、MASK コマンドの MASK 行の表示を削除します。

関連項目:

コマンド:

- “D コマンド” (467 ページ)

RR コマンド

行ブロックを繰り返します。

カテゴリ: テキスト編集、行コマンド

構文

RR <*n*>

block of text

RR <*n*>

引数なし

RR コマンドは行ブロックを 1 回繰り返します。

必須引数

n
指定した行ブロックを繰り返す回数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。ブロックの範囲を示すコマンドの最初または最後の行、あるいはその両方に数値引数を指定できます。最初と最後の両方の行に引数を指定する場合は、最初の数値引数が使用されます。

詳細

RR コマンドは、指定した行ブロックをその直後に繰り返します。デフォルトでは 1 回です。

比較

R コマンドと RR コマンドは、対象の行または行ブロックを、これらのコマンドを指定した行の直後に繰り返します。C コマンドと CC コマンドは、1 つ以上の行をウィンドウ内の任意の場所にコピーできます。

関連項目:

コマンド:

- “R コマンド” (481 ページ)
- “C コマンド” (461 ページ)
- “CC コマンド” (463 ページ)

>コマンド(右へシフト)

指定したテキスト行を右にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

> <*n*>

引数なし

>コマンドは、指定したテキスト行を右に、スペース 1 つ分シフトします。

オプション引数

n
指定したテキスト行がシフトするスペース数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

>コマンドは、指定したテキスト行を右に、スペース 1 つ分以上シフトします。行は *n* 引数で指定するスペース数分シフトしますが、ウィンドウの右境界を超えてシフトすることはありません。このテキストシフトコマンドではシフト時に文字が失われることはありません。

比較

<コマンドと<<コマンドは、>コマンドと>>コマンドとは逆方向にテキストをシフトします。)コマンド、))コマンド、(コマンドと((コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト幅によっては文字が失われることがあります。

関連項目:

コマンド:

- “>>コマンド(ブロックを右へシフト)” (484 ページ)

>>コマンド(ブロックを右へシフト)

指定したテキストブロックを右にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

>> <*n*>

block of text

>> <*n*>

引数なし

>>コマンドは、指定のテキスト行ブロックをスペース 1 つ分右にシフトします。

オプション引数

n

指定のテキスト行ブロックがシフトするスペース数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。ブロックの範囲を示すコマンドの最初または最後の行、あるいはその両方に数値引数を指定できます。最初と最後の両方の行に引数を指定する場合は、最初の数値引数が使用されます。

詳細

>>コマンドは、指定のテキスト行ブロックをスペース 1 つ分以上右にシフトします。テキスト行のブロックは *n* 引数で指定するスペース数分シフトしますが、ウィンドウの右境界を超えてシフトすることはありません。このテキストシフトコマンドではシフト時に文字が失われることはありません。

比較

<コマンドと<<コマンドは、>コマンドと>>コマンドとは逆方向にテキストをシフトします。)コマンド、))コマンド、(コマンドと((コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト幅によっては文字が失われることがあります。

関連項目:

コマンド:

- “>コマンド(右へシフト)” (483 ページ)

SPELL コマンド

スペルチェックを行い、認識されなかった単語にフラグを付けます。

カテゴリ: テキスト編集、コマンド行コマンド

構文

SPELL <ALL <SUGGEST> >

SPELL <NEXT | PREV | SUGGEST>

SPELL <REMEMBER <*dictionary-name*> >

引数なし

SPELL コマンド行にカーソルがある場合は最初の単語をチェックします。それ以外の場合、SPELL コマンドはカーソル位置にある単語をチェックします。KEYS コマンドを使って SPELL コマンドをファンクションキーに割り当てます。このファンクションキーを使用すると、カーソル位置にある単語のスペルチェックが行われます。その単語が認識されたら OK のメッセージが表示されます。そうでない場合は、その単語が認識されなかったことを示すメッセージが表示されます。

オプション引数

ALL

すべての単語のスペルチェックを行います。すべての単語が認識される場合は、認識されなかった単語は見つからなかったことを示すメッセージが表示されます。

認識されなかった単語があれば、**SPELL: Unrecognized Words** ウィンドウが表示され、認識されなかった単語とその行番号が表示されます。このウィンドウには最初に、辞典を指定するための空白フィールドが表示されます。新しい辞書名または既存の辞書名を入力します。**Tools** ⇨ **Remember** を選択し、認識されなかった単語を辞書に追加します。

ヒント SPELL ALL SUGGEST コマンドを指定すると、見つかった認識されなかった単語ごとに **SPELL: Suggestions** ウィンドウが表示されます。

SUGGEST

SPELL: Suggestions ウィンドウを呼び出すと、認識されなかった最後の単語とその行番号、および変更後の単語の候補が表示されます。このウィンドウで **Tools** ⇨ **Remember** を選択すると、この認識されなかった単語を辞書に追加できます。その後、このウィンドウは閉じられ、前のウィンドウに戻され、追加した単語が認識されたことを示すメッセージが表示されます。

認識されなかった単語を変更するには、1 つの候補にカーソルを合わせて ENTER を押します。選択した候補の単語が強調表示されます。**Tools** ⇨ **Replace** を選択します。**Program Editor** ウィンドウに戻ると、この認識されなかった単語が変更されているのがわかります。認識されなかった単語を一括置換する場合は、まずフレーズ **ALL OCCURRENCES** にカーソルを合わせて ENTER を押します。フレーズ ALL OCCURRENCES が強調表示されます。**Program Editor** ウィンドウに戻ると、この認識されなかった単語が一括変更されたことがわかります。

別名 ?

NEXT
現在のカーソル位置の直後にある、認識されない単語を見つけます。現在のカーソル位置からファイルの最後までにある単語がすべて認識された場合は、ファイルの最後に到達したことを示すメッセージが表示されます。そうでない場合は、認識されなかった単語があることを示すメッセージが表示され、カーソルがその認識されなかった単語に移動します。

PREV
カーソル位置の直前にある、認識されない単語を見つけます。現在のカーソル位置からファイルの始めまでにある単語がすべて認識された場合、ファイルの始めに到達したことを示すメッセージが表示されます。そうでない場合は、認識されなかった単語があることを示すメッセージが表示され、カーソルがその認識されなかった単語に移動します。

REMEMBER ***dictionary-name***
最後の認識されなかった単語を補助辞書に追加します。*dictionary-name* は補助辞書の名前です。この認識されない単語が補助辞書に追加されたことを示すメッセージが表示されます。使用している補助辞書が1つだけであれば、*dictionary-name* は省略できます。補助辞書を指定せずに、*dictionary-name* を省略する場合、この認識されなかった単語は、現在の SAS セッションのみ有効な一時辞書に保存されます。

SPELL: Unrecognized Words ウィンドウまたは **SPELL: Suggestions** ウィンドウで単語を選択して強調表示してから、**Tools** ⇨ **Remember** を選択します。追加した単語が認識されるようになります。

別名 ADD

詳細

SPELL コマンドはスペルチェックを行い、認識されなかった単語にフラグを付けます。SPELL コマンドを使って次のタスクを実行できます。

- 認識されなかった単語の候補を確認する
- 認識されなかった単語を補助辞書に追加する
- 認識されなかった単語を候補と置換する

SPELL コマンドはデフォルトの辞書で単語の照合を行います。ただし、デフォルトの辞書以外に、1 つ以上の補助辞書を指定して使用できます。

作成する辞書はすべて、SASUSER.PROFILE カタログに保存されます。SPELL REMEMBER コマンドを使って辞書を更新する場合、更新は現在の SAS セッションの一時辞書に保存されます。辞書名を指定しない場合は一時辞書が作成されます。SASHELP.BASE カタログの辞書を指定する場合は、この辞書に更新が保存されます。

比較

SPELL コマンドはスペルチェックを行い認識されなかった単語にフラグを付け、DICT コマンドは補助辞書の挿入または作成を行います。SPELL コマンドによる補助辞書の作成と更新も可能です。SPELL コマンドを使うと、永続的な補助辞書が作成されます。SPELL コマンドが使用する単語リストは、補助辞書に含まれる単語の記録として機能します。

関連項目:

コマンド:

- “DICT コマンド” (468 ページ)

TC コマンド

2 つのテキスト行を連結します。

カテゴリ: テキスト編集、行コマンド

構文

TC

引数なし

TC コマンドは 2 つのテキスト行を連結します。

詳細

TC コマンドは 2 つのテキスト行を連結します。2 つのテキスト行を連結するには、行の番号フィールドに TC と入力し、Enter または Return を押してください。2 行目のテキストが 1 行目に移動します。1 行目のテキストと 2 行目のテキストの間にスペースは挿入されません。1 行目の最後の単語と 2 行目の最初の単語の間にスペースを設けるには、2 列目のテキストの開始位置を 2 カラム目にします。

このコマンドはテキストの切り捨ては行いません。

比較

TC コマンドは、カーソル位置でテキストを分割する TS コマンドとは逆のコマンドです。TF コマンドに似ていますが、末尾にある空白を削除してパラグラフにテキストを整列するのではなく、境界でテキストを分割する点が異なっています。

関連項目:

コマンド:

- “TF コマンド” (487 ページ)
- “TS コマンド” (488 ページ)

TF コマンド

空白行の区切りまたはテキストの最後までのテキストを整列します。

カテゴリ: テキスト編集、行コマンド

構文

TF <A> <*n*>

引数なし

TF コマンドは、左右境界設定に従って、最初の空白行またはテキストの最後までのテキストを整列します。

オプション引数

A

テキストの末尾の空白を削除して連続しますが、空白行は削除せずに、パラグラフにテキストの最後まで整列します。この引数は、数値引数のように、TF コマンドと同じ行で指定する必要があります。TF コマンドと A 引数の間には文字を配置できません。

n

右境界を指定します。BOUNDS コマンドで設定した右境界より一時的に優先されます。*n* 引数の後に続けて、スペースを 1 つ挿入します。

TS コマンド

カーソル位置でテキストを分割します。

カテゴリ: テキスト編集、行コマンド

構文

TS <*n*>

引数なし

TS コマンドは、カーソル位置でテキスト行を分割し、残りのテキストを新しい行に移動します。

オプション引数

n

残りのテキストを移動するために下がる行数を指定します。デフォルトは 1 行です。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

TS コマンドは、カーソル位置でテキスト行を分割し、残りのテキストを左境界で始まる新しい行に移動します。数値引数を指定する場合、TS コマンドによりそのテキストは指定した行数だけ下に移動します。AUTOFLOW コマンドを有効にすると、TS コマンドは BOUNDS コマンドが指定する左境界を使用します。INDENT コマンドを有効にすると、TS コマンドは左境界の現在のインデントを使用します。AUTOFLOW コマンドを無効にすると、左境界と、左境界の現在のインデントがリセットされます。

次の例では、SAS プログラム内の 2 つのステートメントを分割して、各ステートメントを別々の行に配置します。行 0001 に TS コマンドを入力して、最初のステートメントの後にカーソルを合わせます(Enter または Return を押す前の状態)。

```
ts 01 proc print data=temp; run;
```

Enter または Return を押すと、次の結果が表示されます。

```
00001 proc print data=temp;
00002 run;
```

比較

デフォルト数値引数 1 を設定した TS コマンドの結果は、AUTOSPLIT コマンドを有効にしてキャリッジリターンを入力または Enter もしくは Return を押した場合と同じです。TS コマンドは、2 つのテキスト行を結合する TC コマンドや、空白行またはテキストの最後までのテキストを配列する TF コマンドとは対照的です。AUTOFLOW コマンドが有効な場合、TS コマンドは、BOUNDS コマンドと INDENT コマンドの両方から影響を受けます。

関連項目:

コマンド:

- “AUTOSPLIT コマンド” (458 ページ)
- “I コマンド” (470 ページ)
- “TC コマンド” (487 ページ)
- “TF コマンド” (487 ページ)

UNDO コマンド

アクションをキャンセルします。

構文

UNDO

引数なし
UNDO コマンドは、テキスト編集ができるアクティブウィンドウ内の最新のアクションをキャンセルします。

詳細

UNDO コマンドは、テキスト編集ができるアクティブウィンドウ内の最新のアクションをキャンセルします。このアクションは、テキストを入力または変更するコマンドです。1 つ以上のアクションを元に戻す場合は、UNDO コマンドを連続発行する必要があります。1 度に元に戻せるアクションは 1 つで、最新のアクションで始まり、そこからさかのぼって 1 つずつ元に戻します。

注: UNDO コマンドは SUBMIT コマンドを元には戻せません。サブミットした SAS ステートメントの影響を元には戻せません。

比較

SUBMIT コマンドを元には戻せませんが、RECALL コマンドを使えばサブミットしたステートメントを **Program Editor** ウィンドウに呼び戻せます。

CC コマンドを使ってテキストブロックをコピーして貼り付けてから UNDO コマンドを発行すると、コピーして貼り付けたテキストブロックが削除されます。DD コマンドを使ってテキストブロックを削除してから UNDO コマンドを発行すると、削除したテキストブロックが復元されます。

<コマンド

指定したテキスト行を左にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

< <*n*>

引数なし

<コマンドは、指定したテキスト行をスペース 1 つ分左にシフトします。

オプション引数

n

指定したテキスト行がシフトするスペース数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

<コマンドは、指定したテキスト行をスペース 1 つ分以上左にシフトします。行は *n* 引数で指定するスペース数分シフトしますが、ウィンドウの左境界を超えてシフトすることはありません。このテキストシフトコマンドではシフト時に文字が失われることはありません。

比較

>コマンドと>>コマンドは、<コマンドと<<コマンドとは逆方向にテキストをシフトします。)コマンド、))コマンド、(コマンドと((コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト幅によっては文字が失われることがあります。

関連項目:

コマンド:

- “<<コマンド” (490 ページ)

<<コマンド

指定したテキストブロックを左にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

<< <*n*>

block-of-text

<< <*n*>

引数なし

<<コマンドは、指定のテキスト行ブロックをスペース 1 つ分左にシフトします。

オプション引数

n

指定のテキスト行ブロックがシフトするスペース数を指定します。*n* 引数の後に続けて、スペースを 1 つ挿入します。ブロックの範囲を示すコマンドの最初または最後の行、あるいはその両方に数値引数を指定できます。最初と最後の両方の行に引数を指定する場合は、最初の数値引数が使用されます。

詳細

<<コマンドは、指定のテキスト行ブロックをスペース 1 つ分以上左にシフトします。テキスト行のブロックは *n* 引数で指定するスペース数分シフトしますが、ウィンドウの左境界を超えてシフトすることはありません。このテキストシフトコマンドではシフト時に文字が失われることはありません。

比較

>コマンドと>>コマンドは、<コマンドと<<コマンドとは逆方向にテキストをシフトします。)コマンド、))コマンド、(コマンドと((コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト幅によっては文字が失われることがあります。

(Command

指定したテキスト行 1 つを左にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

(<*n*>

引数なし

(コマンドは、指定したテキスト行をスペース 1 つ分左にシフトします。

オプション引数

n

指定したテキスト行がシフトするスペース数を指定します。デフォルトではスペースは 1 つです。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

(コマンドは、指定したテキスト行をスペース 1 つ分以上左にシフトします。現在行の先頭を超えてシフトすると、文字が失われます。

比較

)コマンドと))コマンドは、テキストをこれらコマンドと逆方向にシフトします。<コマンド、<<コマンド、>コマンドと>>コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト時に文字が失われることはありません。

関連項目:

コマンド:

- “((コマンド” (492 ページ)

((コマンド

指定のテキスト行ブロックを左にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

((<*n*>

block-of-text

((<*n*>

引数なし

((コマンドは、指定のテキスト行ブロックをスペース 1 つ分左にシフトします。

オプション引数

n

指定のテキスト行ブロックがシフトするスペース数を指定します。デフォルトではスペースは 1 つです。*n* 引数の後に続けて、スペースを 1 つ挿入します。ブロックの範囲を示すコマンドの最初または最後の行、あるいはその両方に数値引数を指定できます。最初と最後の両方の行に引数を指定する場合は、最初の数値引数が使用されます。

詳細

((コマンドは、指定のテキスト行ブロックをスペース 1 つ分以上左にシフトします。現在行の先頭を超えてシフトすると、文字が失われます。

比較

)コマンドと))コマンドは、テキストをこれらコマンドと逆方向にシフトします。<コマンド、<<コマンド、>コマンドと>>コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト時に文字が失われることはありません。

関連項目:

コマンド:

- “(Command” (491 ページ)

)コマンド

指定したテキスト行 1 つを右にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

) <*n*>

引数なし

)コマンドは、指定したテキスト行を右にスペース 1 つ分シフトします。

オプション引数

n

指定したテキスト行がシフトするスペース数を指定します。デフォルトではスペースは 1 つです。*n* 引数の後に続けて、スペースを 1 つ挿入します。

詳細

)コマンドは、指定したテキスト行を右にスペース 1 つ分以上シフトします。現在の行の最後を超えてシフトさせると、文字が失われます。

比較

(コマンドと((コマンドは、テキストをこれらコマンドと逆方向にシフトします。<コマンド、<<コマンド、>コマンドと>>コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト時に文字が失われることはありません。

関連項目:

コマンド:

- “))コマンド” (493 ページ)

))コマンド

指定のテキスト行ブロックを右にシフトします。

カテゴリ: テキスト編集、行コマンド

構文

)) <*n*>

block-of-text

)) <*n*>

引数なし

))コマンドは、指定のテキスト行ブロックをスペース 1 つ分右にシフトします。

オプション引数

n

指定のテキスト行ブロックがシフトするスペース数を指定します。デフォルトではスペースは 1 つです。*n* 引数の後に続けて、スペースを 1 つ挿入します。ブロックの範囲を示すコマンドの最初または最後の行、あるいはその両方に数値引数を指定できます。最初と最後の両方の行に引数を指定する場合は、最初の数値引数が使用されます。

詳細

(コマンドと((コマンドは、テキストをこれらコマンドと逆方向にシフトします。<コマンド、<<コマンド、>コマンドと>>コマンドは同じようなテキストシフトコマンドですが、これらのコマンドではシフト時に文字が失われることはありません。

関連項目:

コマンド:

- “)コマンド” (492 ページ)

付録 4

ASCII システムでの EBCDIC データの使用

EBCDIC と ASCII のデータについて

EBCDIC と ASCII のデータ表現の概要

*拡張 2 進化 10 進交換コード(EBCDIC)*は、IBM メインフレームマシンに対する 8 ビット文字エンコーディング方式です。*情報交換用米国標準コード(ASCII)*は、Windows、UNIX、および Macintosh マシンを含む、他のほとんどのマシンに対する 7 ビット文字エンコーディング方式です。

1 バイト、すなわち 8 ビットのデータを表すために、16 進数が使用されます。2 進法では、各ビットが取り得る値は 0 または 1 です。したがって、4 ビットの集合は 16 (2^4)の可能値を取り得ます。つまり、2 桁の 16 進数で、1 バイトのデータを表せます。EBCDIC と ASCII のエンコーディング方式では、各文字が 2 桁の 16 進数で表現されます。(これは主に、西洋言語の 1 バイトエンコーディング方式に関して言えることです。この他に、日本や韓国のデータに使用されるエンコーディング方式のように、1 文字を 2 バイトの記憶域に保存するエンコーディング方式もあります)。

次の例に示すように、各エンコーディング方式では、同じデータが異なる方法で表されます。

- EBCDIC システムでは、数字の 4 が 16 進値'F4'x で表されます。ASCII システムでは、数字の 4 が 16 進値'34'x で表されます。
- EBCDIC システムでは、16 進値'50'x が記号&を表します。ASCII システムでは、同じ 16 進値が文字 P を表します。

SAS では、ファイルの読み取り時に、ファイル内のデータが、SAS セッションに対する ENCODING=オプションと一致するエンコーディングであることが必要とされます。たと

えば、Windows マシンでは、米国英語ロケールの 1 バイト SAS セッションに対するデフォルトエンコーディングは LATIN1 です。SAS では、その Windows マシンのファイル内のデータが LATIN1 エンコーディングを使用する必要があります。ただし、ファイルの作成元が EBCDIC マシンであり、Windows マシンで保存されている場合、SAS では、その他のエンコーディング情報が提供されなければ、ファイルからのデータが誤って解釈されることになります。このような理由で、作成元が EBCDIC システムのデータは、ASCII システム(たとえば、Windows マシンなど)で使用する前に、特定のステップを実行して変換しておく必要があります。EBCDIC データを ASCII システムで使用可能にするための主な方法を 2 つ次に示します。

- ASCII システムで、EBCDIC システムからデータを直接読み取る。
- データ変換の有無にかかわらず、FTP プログラムを使用してデータを移動する。

EBCDIC ファイル構造

EBCDIC システムから ASCII システムへのデータ移動方法を決定する際、EBCDIC ソースファイルの構造を考慮します。EBCDIC システムには、固定長レコードから成るファイルや可変長レコードから成るファイルがあります。どちらのタイプのファイルにも、ファイルについての情報から成るヘッダーが含まれます。ヘッダーには、レコードが固定長と可変長のどちらなのかを示すレコード形式属性が含まれます。固定長レコードから成るファイルのヘッダーには、各レコードの長さをバイト単位で示す論理レコード長属性が含まれます。

SAS では、このレコード形式属性は、FILENAME ステートメントの RECFM=オプションに相当します。固定長レコードから成るファイルにアクセスするには、RECFM=F を指定します。可変長レコードから成るファイルにアクセスするには、RECFM=V を指定します。同様に、論理レコード長属性は、LRECL=オプションに相当します。

可変長レコードから成るファイルのヘッダーの論理レコード長属性では、最大レコード長が示されます。可変長レコードから成るファイル内の各レコードは、*レコード記述語(RDW)*から始まります。RDW は、4 バイトの 2 進整数フィールドです。RDW の最初の 2 バイトは、現在のレコードの長さを示します。RDW の最後の 2 バイトには、オペレーティングシステムで使用される情報が含まれます。レコードの長さには、レコードの先頭にある 4 バイトの RDW も含まれます。EBCDIC ファイルでは(ヘッダーか RDW のどちらかで)各レコードの長さが指定されるので、EBCDIC ファイルにレコードの終わりインジケータはありません。

可変長レコードから成るファイルには、*ブロック記述語(BDW)*も含まれます。RDW と同様に、BDW は 4 バイトの 2 進整数フィールドです。最初の 2 バイトはブロックサイズを示し、最後の 2 バイトはオペレーティングシステムで使用されます。各ブロックには、複数のレコードを含められます。ファイルの作成時にブロックサイズが指定されなかった場合、デフォルトブロックサイズは、論理レコード長に 4 を加えた値になります。あるいは、ブロックのサイズは、ブロックに含まれるバイト数になります。これは、ブロック内のレコード長(RDW から取得)の合計に 4(BDW の長さ)を加えた値です。

ASCII ファイル構造

ASCII システムでは、ファイルに、ファイルについての情報(レコードの形式や長さなど)から成るヘッダーは含まれません。ASCII ファイルの RECFM 属性は可変(RECFM=V)で、レコード長(LRECL)は無制限です。EBCDIC ファイルのようにレコード長を定義するかわりに、ASCII ファイルでは、レコードの終わりインジケータを使用して、レコードの終わりにフラグを設定します。Windows マシンでは、レコードの終わりインジケータは、キャリッジリターン(CR)およびラインフィード(LF)文字です。UNIX マシンでは、LF がレコードの終わりを示します。Macintosh マシンでは、CR がレコードの終わりを示します。他の種類のマシンでは、異なる文字の組み合わせを使用して、レコー

ドの終わりを識別します。すべての ASCII マシンについて、CR の 16 進値は'0D'x で、LF の 16 新値は'0A'x です。

ASCII マシンで SAS によりディスクからファイルを読み取る場合、一部のファイル属性については、定義がされていないため、デフォルト値を使用する必要があります。デフォルト RECFM 値は V (可変長レコード)で、デフォルト LRECL 値は 32767 です。つまり、SAS は、ASCII ファイルから入力をスキャンし、INPUT ステートメントに基づいてデータを解析して変数値にし、レコードの終わりインジケータを探します。指定文字数内(LRECL に基づく)でレコードの終わりが見つからない場合、SAS では、レコードが切り捨てられ、ログにメッセージが出力されます。たとえば、LRECL が 256 に設定されており、なおかつ 300 文字のレコードが存在するとします。SAS では、INPUT ステートメントに基づいて最初の 256 文字が読み取られた後、最後の 44 文字が破棄されます。ログのメッセージは“One or more lines have been truncated”となります。現在の LRECL 値を無効化するには、INFILE ステートメントの LRECL=オプションを使用します。

数値

文字データとして保存された場合、10 進数の 0 から 9 まではそれぞれ 1 バイトの記憶域を占めます。1 つの 8 ビットバイトには、2 つの 4 ビット*ニブル*が含まれます。各ニブルは、16 (2^4)の可能値を取り得ます。最初のニブルが*高位ニブル*、2 番目が*低位ニブル*です。EBCDIC システムと ASCII システムでは、高位ニブルは標準値を取ります。EBCDIC では、10 進数は高位ニブル F で表されます。ASCII では、10 進数は高位ニブル 3 で表されます。つまり、EBCDIC システムでは、0 から 9 までの数字が'F0'x から'F9'x までの 16 進値で表されます。ASCII システムでは、0 から 9 までの数字が'30'x から'39'x までの 16 進値で表されます。このエンコーディング方式では、10 進数を文字として処理します。

10 進数の文字形式保存に代わる方法として、他にも EBCDIC システムで使用できるエンコーディング方式があります。たとえば、パック 10 進エンコーディング方式では、1 バイトの記憶域で 2 桁の 10 進数が表されます。ゾーン 10 進エンコーディング方式では、1 バイトの記憶域で 1 桁の 10 進数が表され、値全体の符号がこの 1 バイトの記憶域内に含まれます。(10 進数と値全体の符号が保存されるバイトは、マシンの種類によって、最初のバイトの場合も最後のバイトの場合もあります)。

ソースデータが ASCII システムで正しく解釈されるように、ソース EBCDIC システムで使用される数値エンコーディングを知っておく必要があります。つまり、SAS においては、数値データ用の正しい入力形式を指定する必要があります。

EBCDIC システムから ASCII システムへのデータ移動

ASCII システムでの EBCDIC データへのアクセスの概要

ASCII システムで EBCDIC データにアクセスするには、いくつかの方法があります。たとえば、ASCII マシンには、EBCDIC システムで作成された 3480 または 3490 カートリッジテープを読み取れる周辺機器が備えられていることもあります。これらの機器では、テープのデータを ASCII マシンのアプリケーションに直接読み込めます。あるいは、これらの機器で、テープからデータをコピーして、ASCII マシンのハードドライブに保存することもできます。

データの移動と変換でより多く見られる方法は、FTP プログラムを使用したデータ転送です。デフォルトでは、ほとんどの FTP プログラムで、データの転送時に、EBCDIC データが ASCII に変換されます。ソースデータに文字データ(文字としてエンコーディングされた数字も含む)しか含まれていない場合は、これがお勧めの方法です。変換プ

ロセス中に、FTP プログラムでは、ASCII システムに対して適切なレコードの終わりインジケータが作成されます。変換後に、INFILE ステートメントを使用して、ASCII システムの新規作成ファイルにアクセスできます。INPUT ステートメントを使用して、ファイルのデータ読み取り時に使用する正しい入力形式値を指定します。

注: EBCDIC ソースデータがすべて文字データとしてエンコードされた場合でも、変換中に一部の文字が正しく解釈されないこともあります。これらの文字の正しい解釈は、EBCDIC マシンで使用されるエンコーディング方式に依存します。ベストプラクティスとしては、SAS が変換ファイルから読み取るデータを表示して、データが正しく変換されたことを確認します。

パック 10 進またはゾーン 10 進エンコーディング方式の使用時などで、EBCDIC ファイルに、文字データとしてエンコードされていない数値データが含まれている場合、デフォルト FTP 変換が正しく機能しません。数値データには、標準文字データに似ているものもあります。この場合、FTP 変換では、ASCII 文字を EBCDIC 数値データに正しく割り当てられません。詳細については、“パック 10 進数値データの間違った変換例”を参照してください。

注: パック 10 進エンコードデータを EBCDIC から ASCII に正しく変換する方法はありません。パック 10 進、ゾーン 10 進、またはその他の数値エンコーディング方式が EBCDIC システムで使用される場合、データを変換するには他の方法を使用する必要があります。詳細については、“可変長レコードから成る EBCDIC ファイルの変換” (499 ページ)を参照してください。

場合によっては、ある 1 バイトの EBCDIC データが、ASCII で、行の終わりフラグまたはファイルの終わりフラグとして解釈されることもあります。SAS で、可変長レコードから成るファイルの読み取り中に、これらの 16 進値の 1 つが検出された場合、意図しない結果がもたらされることがあります。指定入力形式に基づく予想データ値に応じて、無効データエラーから DATA ステップの予期しない終了などが発生します。

パック 10 進数値データの間違った変換例

この例では、パック 10 進数値データを、文字データとしてエンコードされた場合と同様に変換した場合、結果的に発生し得る問題を示します。EBCDIC データファイルに、パック 10 進値('505C'x)として保存された、数値 505 が含まれているとします。EBCDIC ファイルブラウザまたはエディタでファイルを参照すると、文字&*が表示されます。これは、'50'x が&に対応し、'5C'x が*に対応するためです。FTP プログラムでは、&文字が解釈され、ASCII 値'26'x に変換されます。FTP プログラムでは、*文字は ASCII 値'2A'x に変換され、結果の変換値は'262A'x になります。ASCII での正しいパック 10 進値は'000505'x です。入力データが、予想されたパック 10 進入力形式に準拠していないため、SAS では、データが無効であるというエラーがログに出力されます。無効データが検出されるたびに、SAS では、エラーがログに書き込まれて、入力バッファのコンテンツと、対応する DATA ステップ変数が出力されます。

表 A4.1 *パック 10 進数値データの間違った変換*

ステップ	アクション	値
1	FTP プログラムで、EBCDIC パック 10 進数値‘505’ が読み取られます。	'505C'x
2	FTP プログラムで、値が、標準 EBCDIC 文字として解釈されます。	&*
3	FTP プログラムで、標準 ASCII16 進数文字に変換されます。	'262A'x

ステップ	アクション	値
4	(指定入力形式値に基づいて)パック 10 進数値データが予想されるため、SAS では、データに無効フラグが設定されます。	???

固定長レコードから成る EBCDIC ファイルの変換

バイナリでのファイルの FTP 送信

固定長レコードから成る EBCDIC ファイルを変換する際は、FTP を使用してファイルをバイナリ転送します。次に、FILENAME または INFILE ステートメントとともに RECFM=F を指定し、EBCDIC システムでのファイル設定と同じ値を LRECL に割り当てます。次の入力形式でフォーマットされた入力スタイルを使用します。

- $EBCDIC*w*. (文字入力データの場合)
- S370F*xxxw.d* (数値入力データの場合)

  注: 370F*xxxw.d* 入力形式は多数存在します。各自のデータの種類に合う入力形式を選択します。詳細については、*SAS Formats and Informats: Reference* (SAS 9.3 以降)、または *SAS Language Reference: Dictionary* (SAS の旧バージョン)を参照してください。

ソースファイルをバイナリ転送しているため、レコードの終わりインジケータを追加する処理はありません。このため、EBCDIC システムでソースファイルに対して指定された正確なバイト数を指定する必要があります。ソースファイルに、ASCII のコンテキストでレコードの終わりインジケータやファイルの終わりインジケータとして解釈されるべきバイトが存在する場合、SAS では、そのバイトは単にデータとして処理されます。

例: 固定長レコードから成る EBCDIC ファイルを ASCII ファイルに変換

次のコードでは、以前 FTP で EBCDIC システムから ASCII システムにバイナリ転送されたファイル fixed.txt が読み取られます。ソースファイルには、60 バイト長の固定長レコードが含まれます。この例の入力形式に基づくと、各レコードの最後の 3 バイトには、パック 10 進エンコーディング方式を使用して保存された数値データが含まれます。

```
filename test1 'c:\fixed.txt' recfm=f lrecl=60;
data one;
infile test1;
input @1 name $ebcdic20.
@21 addr $ebcdic20.
@41 city $ebcdic15.
@56 state $ebcdic2.
@58 zip $s370fpd3.;
run;
```

可変長レコードから成る EBCDIC ファイルの変換

可変長レコードから成る EBCDIC ファイルの変換の概要

可変長レコードから成る EBCDIC ファイルを変換する際は、FTP プログラムを使用できます。FTP プログラムでは、BDW と RDW が削除され、ASCII システムで必要とさ

れるレコードの終わりインジケータが追加されます。ファイル内のデータは EBCDIC から ASCII に変換されます。EBCDIC ファイル内のデータがすべて文字としてエンコードされている場合、通常、このプロセスは正しく機能します。

注: EBCDIC ソースデータがすべて文字データとしてエンコードされた場合でも、変換中に一部の文字が正しく解釈されないこともあります。これらの文字の正しい解釈は、EBCDIC マシンで使用されるエンコーディング方式に依存します。ベストプラクティスとしては、SAS が変換ファイルから読み取るデータを表示して、データが正しく変換されたことを確認します。

パック 10 進またはゾーン 10 進エンコーディング方式の使用時などで、EBCDIC ファイルに、文字データとしてエンコードされていない数値データが含まれている場合、デフォルト FTP 変換が正しく機能しません。詳細については、“ASCII システムでの EBCDIC データへのアクセスの概要” (497 ページ)を参照してください。変換中の誤った解釈を防ぐには、データを ASCII エンコーディングに変換せずに、FTP でファイルをバイナリ転送します。データがバイナリ転送されて変換されない場合は、BDW と RDW の情報が自動的に削除されることに注意してください。これにより、SAS でデータの正常な読み取りに必要な情報が削除されます。

EBCDIC システムからファイルを直接読み取る

ASCII マシンと EBCDIC マシン間の直接アクセスが可能な場合、ベストプラクティスはファイルを直接読み取ることです。直接アクセスは、EBCDIC テープを読み取れる、ASCII マシンの周辺機器によって可能になります。FILENAME ステートメントで、FTP アクセス方法によってファイルにアクセスできます。この EBCDIC データアクセス方法には、いくつかの利点があります。

- ファイルの前処理が不要
- ソースファイルのコピーが不要
- FTP アクセス方法は、固定長レコードと可変長レコードに有効
- DATA ステップ処理が予想どおりに機能する

主な欠点は、この方法では、データにリモートアクセスするため、処理により多くの時間を要することです。

3480 または 3490 カートリッジテープ読み取り装置が ASCII マシンに取り付けられている場合、この EBCDIC データアクセス方法が適用されます。この場合、EBCDIC マシンではファイルの前処理は必要ありません。テープから直接読み取るには、RECFM=S370VB を設定し、$EBCDIC*w*.および S370F*xxxw*.*d* 入力形式を使用します。

FILENAME ステートメントで、FTP アクセス方法およびソースファイル名、ならびに HOST=、USER=、および PASS=オプションの値を指定します。HOST=オプションでは EBCDIC マシンの名前、USER=ではログオンに使用するユーザーアカウント、PASS=ではログオンに使用するパスワードを指定します。FTP アクセス方法では、ASCII マシンの FTP プログラムを使用して、ASCII マシンと EBCDIC マシン間の接続を確立します。SAS システムは、指定されたユーザーアカウントとパスワードで、メインフレームマシンに接続し、ログオンします。FTP プログラムで、ファイルが転送されます。

注: PASS=オプションを指定する場合、パスワードは、SAS プログラムでテキストとして保存されます。パスワードは、SAS ログには表示されません。PASS=オプションに代わる方法として、PROMPT オプションを指定して、SAS プログラムの実行時にプロンプトでパスワードを指定することもできます。

可変長レコードから成る EBCDIC ファイルについては、S370V および RCMD=オプションも指定する必要があります。S370V オプションは、ソースファイル内のレコードが可変長であることを示します。RCMD=オプションについては、RCMD="SITE RDW"を指定して、FTP プロセスでファイル転送中に RDW 情報を保持する必要があることを示します。

EBCDIC マシンに接続の問題が発生した場合は、DEBUG オプションを追加して、FTP サーバーとの間で送受信された情報メッセージを参照します。

例: FTP アクセス方法で EBCDIC ソースファイルを直接読み取る

この例では、FTP アクセス方法を使用して、可変長レコードから成る EBCDIC ファイルを EBCDIC マシンから直接読み取る方法を示します。ユーザーは、MVS ログオンパスワードの入力を要求されます。ZIP コードは、EBCDIC 数字 5 桁の入力で、1 バイトにつき 1 桁で表されます。コメントセクションは、200 文字以内で長さはさまざまです。データは、読み取り後に、データセットのコンテンツを確認するために出力されます。

```
filename test1 ftp "'SASEBCDIC.VB.TEST1'" host='MVS' user='SASEBCDIC' PROMPT
s370v rcmd='site rdw';
data one;
infile test1;
input @1 name $ebcdic20.
@21 addr $ebcdic20.
@41 city $ebcdic10.
@51 state $ebcdic2.
@54 zip s370ff5.
@60 comments :$ebcdic200.;
run;

proc print;
run;
```

可変長レコードから成る EBCDIC ファイルを IEBGENER で再フォーマット

ASCII マシンと EBCDIC マシン間の直接アクセスができないとします。これはつまり、ASCII マシンに EBCDIC データを読み取る周辺機器がないということです。この場合、メインフレームマシンでファイルを再フォーマットすると、データを変換できます。ファイルの形式を変更すると、FTP プログラムが、SAS でデータを正しく読み取るために必要な RDW 情報を削除するのを防げます。ファイルを再フォーマットした後で、ASCII マシンにファイルをバイナリ転送できます。

ソースファイルを再フォーマットするには、EBCDIC マシンで IEBGENER プログラムを使用します。このプログラムを使用して、ヘッダー情報が変更されたファイルの正確なコピーを作成します。具体的には、IEBGENER を使用して、RECFM 値を V (ブロック単位の可変長レコード)から U (未定義レコードおよび非ブロック化)に変更します。この変更を行うと、FTP プログラムでは、ファイル転送中に RDW 情報が削除されなくなります。

IEBGENER プログラムを実行する際には、必須引数に加えて、次の優先値を指定します。

```
SYSUT1 DCB=(RECFM=U,BLKSIZE=32760)
SYSUT2 DCB=(RECFM=U,BLKSIZE=32760) DISP=(NEW,CATLG)
```

注: SYSUT1 に対して RECFM および BLKSIZE の元の値は使用しないでください。

ASCII マシンで FTP プログラムを使用して、新しいバージョンのファイルをバイナリ転送します。SAS で、FILENAME または INFILE ステートメントを使用して、転送ファイルを読み取ります。オプションを適切に設定します。

- 元のファイルのレコード形式が可変(RECFM=V)だった場合は、RECFM=オプションを S370V に設定します。元のファイルのレコード形式が可変かつブロック化(RECFM=VB)だった場合は、RECFM=オプションを S370VB に設定します。RECFM=オプションを S370V または S370VB として指定すると、SAS に対して、各レコードの RDW 情報を処理し、各レコードの正しいバイト数を入力するよう指定したことになります。

- LRECL=オプションについては、元のファイルと同じ値を指定します。LRECL=オプションに値を指定しなければ、SAS では、デフォルト LRECL 値(32767)が使用されます。データレコードがデフォルト LRECL 値よりも長い場合、デフォルト値を使用すると、SAS でデータレコードの切り捨てが引き起こされる可能性があります。

“バイナリでのファイルの FTP 送信” (499 ページ)に記載された入力形式でフォーマットされた入力スタイルを使用します。

例: ファイルと変更済みヘッダーデータの読み取り

この例では、EBCDIC ファイルから生成されたファイルを、ファイル形式を変更するために変更されたヘッダーと一緒に読み取ります。変更ファイルは、SAS 処理のために ASCII マシンに転送されました。詳細については、“可変長レコードから成る EBCDIC ファイルを IEBGENER で再フォーマット” (501 ページ)を参照してください。

Comment 変数に 60 文字まで指定できる(ただし、それよりも短い可能性が高い)ため、INFILE ステートメントに TRUNCOVER オプションが含められます。TRUNCOVER オプションを指定しなければ、INPUT ステートメントは、レコードの終わりを越えて読み取ろうとします。変数が上限に達するまで、次のレコードのデータが続けて Comment 変数に割り当てられていきます。デフォルト値が、ファイル内の最長レコードを処理するのに十分であるため、LRECL=オプションは指定されません。データは、読み取り後に、確認のために出力されます。

```
filename test1 'c:\vbtest.xfr' recfm=s370vb;
data one;
infile test1 truncover;
input @1 name $ebcdic14.
@15 addr $ebcdic18.
@33 zip s370ff5.
@38 comment $ebcdic60.;
run;

proc print;
run;
```

構造化 COBOL ファイルから EBCDIC データを読み取る

構造化 COBOL ファイルについて

構造化 COBOL ファイルは、OCCURS DEPENDING ON 句を使用して生成されます。この種類のファイルのレコードは可変長です。また、ファイルが FTP でバイナリ転送された場合、BDW または RDW 情報は存在しません。各レコードは、レコードヘッダー(レコードの固定長部分)、インデックス変数、および 1 つ以上のデータセグメントの 3 部分に分割されます。ファイルのドキュメントで、レコードヘッダー、インデックス変数、およびデータセグメントの長さが提供されます。レコードヘッダーは、各レコードに対して同じ長さです。これには、後に続くすべてのデータセグメントに関する情報が含まれます。インデックス変数では、現在のレコードのデータセグメント数が提供されます。レコードの残り部分には、データセグメントが含まれます。

レコードの構造のおかげで、SAS では、これらのファイルのデータを読み取れます。レコードの長さは、ヘッダー長、インデックス長、およびインデックス値と各データセグメントのサイズとの積の合計です。データセグメントごとに、SAS で、セグメントが読み取られ、ヘッダーのコピーと現在のデータセグメントが、SAS データセットの新しいオブザベーションに出力されます。

構造化 COBOL ファイルを読み取る際は、FILENAME ステートメントに RECFM=N を指定します。これにより、SAS に対して、標準的なファイル構造に準拠しないデータのストリームを読み取ることを示します。SAS が入力のバッファ処理を試行しないため、

SAS によるデータストリーム読み取り時には、レコード長に対する制限はいずれも無視されます。SAS では、SAS ログにステートメントが書き込まれ、RECFM=N の場合、SAS がバッファなしでデータストリームを読み取ることが通知されます。

SAS では、COBOL ファイル全体が、1 つの長いレコードとして読み取られます。したがって、一部のデータをスキップしたり、スペースの後ろに移動したりする必要がある場合、INPUT ステートメントで相対カラムポインタを使用する必要があります。ファイルのコンテンツが 1 つの入力レコードとして処理されるため、行ホルダは無視されます。*@column* ポインタは、これらのファイルでは機能しません。

注意:

RECFM=N を指定した場合、*@column* ポインタは使用しないでください。 *@column* ポインタを使用すると無限ループが開始されて、プログラムを停止するか、使用可能なディスク領域がなくなるまで、SAS で同じデータの読み取りと出力が繰り返されます。

例: 構造化 COBOL ファイルからデータを読み取る

この例では、IEBGENER を使用した最初のファイル処理なしで、EBCDIC ファイルが FTP でバイナリ転送されました。各レコードのレコードヘッダー(固定長)部分は 59 バイトの長さで、文字データと数値データの組み合わせが含まれます。インデックス変数は 2 バイトです。インデックス変数とレコードの残り部分を区切るためにスペースがもう 1 つ(1 バイト)あります。レコードのデータセグメント部分は、13 バイト長の 1 回以上の繰り返しから成ります。各繰り返しには、文字データと数値データの組み合わせが含まれます。

```
filename test1 'c:\VB.TEST' recfm=n;

data one;
infile test1;
input name $ebcdic20. addr $ebcdic20. city $ebcdic10. st $ebcdic2. +1
zip s370ff5. +1 idx s370ff2. +1;
do i = 1 to idx;
input cars $ebcdic10. +1 years s370ff2. ;
output;
if i lt idx then input +1 ;
end;
run;
```

用語集

cat
concatenate (連結)という意味の UNIX コマンド。ファイルコンテンツをリスト化したり、ファイルを連結したりする場合にこのコマンドがよく使用されます。

I/O 時間
入力/出力時間の短縮形。I/O 時間とは、データをストレージから作業用メモリに移動する際にかかる時間(入力時間)と、結果をメモリからストレージまたは表示デバイス(ターミナルやプリンタなど)に移動する際にかかる時間(出力時間)のことです。

lp
UNIX では、ラインプリンタデーモンを介してプリンタ出力先に出力を送信する場合に line-printer コマンドがよく使用されます。

motif
UNIX 環境で使用される X Window System のグラフィカルユーザーインターフェイス(GUI)。

path
特定のファイルやディレクトリに至る階層ファイルシステムを通過する経路。

PID
プロセス ID を参照してください。

sasauth
ユーザー認証および識別関数を実行する SAS サブプロセス。sasauth プロセスは! SASROOT/utilities/bin ディレクトリに格納されています。

sasperm
ユーザーのリソースアクセス特権を決定する SAS サブプロセス。

sasroot
SAS がユーザーのサイトまたはコンピュータにインストールされる場所のディレクトリまたはフォルダの名前を表す用語。

sasuser プロファイルカタログ
SAS ウィンドウ環境の属性に関する情報が SAS によって保存される SAS カタログ。たとえば、このカタログにはファンクションキーの定義、グラフィックアプリケーションのフォント、ウィンドウ属性、対話型 SAS プロシジャによって使用されるその他の情報が含まれます。

SAS カタログ
カタログエントリと呼ばれる小さい単位でさまざまな種類の情報を保存する SAS ファイル。1 つの SAS カタログには、さまざまな種類のカタログエントリが含まれます。

SAS ファイルの変換
SAS ファイルの形式を、SAS のあるバージョンに適した形式から同じオペレーティング環境の別バージョンに適した形式に変更する処理。

SAS ライブラリ
SAS によって認識され、ユニットとして参照および保存される 1 つ以上のファイルの集合。各ファイルはライブラリのメンバになります。

X Window System
マサチューセッツ工科大学で開発されたグラフィカルウィンドウシステム。

X サーバー
X Window System において、1 つ以上のアプリケーションクライアントプログラムからディスプレイ、マウスおよびキーボードへのアクセスを仲介するプログラム。

X リソース
フォントタイプ、フォントサイズ、色、グラビティ、ウィンドウサイズなどのウィンドウインターフェイスの特徴。

X リソースファイル
X Window System において、色、グラビティ、フォントタイプ、サイズ、ウィンドウサイズなどのウィンドウ環境用の属性指定を保存するファイル。

アクティブウィンドウ
開いて表示されており、キーボード入力が直接反映されるウィンドウ。一度にアクティブにできるウィンドウ数は 1 つのみです。

インデックス
SAS データセットの構成要素。インデックスを使用すると、SAS から SAS データセット内のオブザベーションに素早く効率的にアクセスできます。SAS インデックスの目的は、WHERE 句処理を最適化し、BY グループ処理を促進することです。

エラーメッセージ
SAS がプログラムの処理を継続できなかったことを示すメッセージ。SAS ログやメッセージウィンドウに表示されます。

エンコーディング
コード化した文字セットをコード値にマッピングすること。

外部ファイル
ホストオペレーティングシステムまたは別のベンダのソフトウェアアプリケーションによって作成および管理されるファイル。SAS は外部ファイルからデータを読み取り、外部ファイルに出力できます。外部ファイルには生データ、SAS プログラミングステートメント、プロシジャ出力、PUT ステートメントによって作成された出力のいずれかが含まれます。SAS データセットは外部ファイルではありません。

カタログ
SAS カタログを参照してください。

カレントディレクトリ
所定の時間にユーザーが作業している場所のディレクトリ。ログオン時のカレントディレクトリは、相対パス名の開始地点です。

環境変数
UNIX 環境では、シェル変数の値には、そのシェルから実行される任意のプログラムがアクセスできます。シェルによってデフォルト値が一部の環境変数に割り当てられます。たとえば、ターミナルの種類やコマンドプロンプトの種類は、2 つの環境変数のデフォルト値によって指定されます。

クライアント
サーバーから(通常はネットワーク経由で)リソースまたはサービスのいずれかを要求するアプリケーション。

クラス名
個別の X リソースをまとめてグループ化する方法を提供する名前。たとえば、DMSboldFont および DMSFont は個別の 2 つの X リソースですが、両方とも Font クラスに含まれます。

コマンドインタープリタ
コンピュータが理解できる言語へコマンドを変換するプログラム(シェルなど)。

コマンド行
コマンド===>で指定される任意の SAS ウィンドウ環境のウィンドウ内の場所。

コマンドプロンプト
オペレーティングシステムのコマンドを入力した後の記号。UNIX 環境では、シェルが異なるとコマンドプロンプトの種類も異なります。Bourne シェルおよび Korn シェルのデフォルトコマンドプロンプトは$で、C シェルのデフォルトプロンプトは%です。

コンテナウィンドウ
内部ウィンドウを含む SAS ウィンドウ。

サーバー
要求側のクライアントにリソースまたはサービスのいずれかを(通常はネットワーク経由で)提供するソフトウェア。

作業ディレクトリ
ソフトウェアアプリケーションが起動される場所のディレクトリ。

シーケンシャルアクセス
ファイルの開始から終了まで順番にレコードが読み取りまたは書き込みされるファイルアクセス方法。

シェル
UNIX コマンドインタープリタ。サンプルシェルとして sh、csh および ksh があります。

シェルスクリプト
シェルによって読み取られ、実行されるコマンドを含むファイル。

集計構文
1 つのディレクトリやフォルダ内にある個別のファイルを参照する簡易方法。一意のファイル参照名を各ファイルに割り当てるのではなく、ファイル参照名をディレクトリやフォルダに割り当てることになります。続いて、当該フォルダ内にある特定ファイルを参照するには、ファイル名を丸かっこで囲み、後ろにファイル参照名を続

けます。集計構文は、FILE、INFILE および%INCLUDE ステートメント使用されます。

スワップ
データまたはプログラムコードをコンピュータシステムのメインメモリから、ハードディスクなどのストレージデバイスに移動すること。あるいはその逆。

セッション
ソフトウェアアプリケーションが起動してから実行を終了するまでの 1 つの期間。

セッショングラビティ
SAS に対する X Window インターフェイスにおいて、SAS によってそのウィンドウの配置が試行される場所のワークステーション表示の領域を制御するリソース。

対話型ラインモード
SAS セッションプロンプト時にユーザーが SAS プログラムのラインを入力する場合に、SAS プログラムが実行される方法。Enter キーを押した直後に SAS によって各ラインが処理されます。プロシジャ出力と参照メッセージが表示デバイスに直接返されます。

ダウンロード
リモートホストからファイルをローカルホストにコピーすること。

ツールセット
アプリケーションに関連付けられた、事前に定義されたツールの集合。ツールセットを使用すると、各ユーザーはそれぞれのアプリケーションツールボックスを簡単にカスタマイズできるようになります。

ツールボックス
SAS コマンドやマクロに関連付けることができるアイコンを配置可能な SAS ウィンドウ環境の一部。アイコンを選択すると、その関連コマンドやコマンド文字列が実行されます。

ディレクトリ
ファイルのグループや他のディレクトリを含む UNIX オペレーティング環境における特殊な種類のファイル。

デバイスドライバ
コンピュータと外部デバイス(プリンタやディスクドライブなど)間の対話を制御するプログラム。

特殊ファイル
UNIX オペレーティングシステムにおける入力または出力デバイスへのインターフェイス。ファイルから読み取るか、またはファイルに書き込むと、デバイスが有効になります。

トグル
処理機能のオン/オフを切り替えることができるオプション、パラメータまたはその他のメカニズム。

ネットワーク
相互に接続されたコンピュータのグループ。

パイプ
UNIX オペレーティングシステムおよびその派生型において、あるコマンドの標準出力が他のコマンドの標準入力になるように、1 つのコマンドを別のコマンドにリンクする機能。

パス名
UNIX オペレーティングシステムでは、ファイル名によってファイル階層内の特定のファイルに至るすべてのディレクトリが指定されます。

バックグラウンド処理
UNIX 環境では、プロセスはシェルとは独立して実行されます。バックグラウンド処理でコマンドを実行する場合、最初のコマンド実行を待たなくても他のコマンドを入力したり、他のバックグラウンド処理を開始したりすることができます。

バッチモード
SAS ステートメントが含まれるファイル内の SAS プログラムを実行し、必要なオペレーティング環境コマンドがコンピュータのバッチキューに送信される方法。プログラムを送信してコンピュータにコントロールが返されたら、他のタスクを実行できます。バッチモードは、バックグラウンドでの実行と言われる場合もあります。プログラム出力はファイルに書き込んだり、出力デバイスに出力したりすることができます。

バッファ
入力/出力(I/O)処理の実行時に使用するために確保されるコンピュータメモリ領域。

非対話型モード
ユーザーが SAS ステートメントのファイルを準備し、プログラムをオペレーティングシステムに送信する場合に、SAS プログラムが実行される方法。プログラムはただちに実行され、ユーザーの現在のセッションを構成します。

標準エラー
UNIX オペレーティングシステムにおけるプログラムのエラーメッセージの出力先。標準エラーの別名は stderr です。

標準出力
コマンドから送信されるデータの主要送信先。標準出力は、ファイルからリダイレクトされないか、または別のコマンドからパイプ処理されない限り、ディスプレイに送信されます。標準出力の別名は stdout です。

標準入力
コマンドに送信されるデータの主要ソース。標準入力は、ファイルからリダイレクトされないか、または別のコマンドからパイプ処理されない限り、キーボードから送信されます。標準入力の別名は stdin です。

ファイル拡張子
ディレクトリ内のファイルの分類のことで、ファイル内にどの種類の情報を保存するかが識別されます。たとえば、.sas7bcat は UNIX の拡張子で、.pdf は Adobe Acrobat のファイル拡張子です。

ファイルディスクリプタ
UNIX オペレーティングシステムにおいて、読み取りまたは書き込み(あるいは両方)用に開いたファイルへの参照に使用される負ではない整数識別子。

フォアグラウンド処理
UNIX 環境において、コマンドプロンプトが再表示されるまで待機している間に実行されるプロセス。最初のコマンドがフォアグラウンド処理で実行されている間は、追加コマンドを実行できません。

フォント
特定の文字の形、スペース、ウェイト、サイズを有する書体。1 つのフォントには、文字、記号または英数字が含まれます。

プロセス ID
オペレーティングシステムによって各プロセスに割り当てられる一意の番号。短縮形: PID。

プロトコル
コンピュータと周辺機器間のデータ通信を制御するルールの集合。

ホームディレクトリ
ログイン後にユーザーが配置されるディレクトリ。ホームディレクトリの別名はログインディレクトリです。

メモリ
中央処理装置(CPU)がプログラム内で処理に専念する必要がある作業領域のサイズ。

メンバ
SAS ライブラリ内の SAS ファイルの種類。SAS ファイルの種類には、データセット、ビュー、カタログ、プログラム、アクセスディスクリプタが含まれます。

ユニバーサル印刷
SAS ソフトウェアの機能。これを使用すると、SAS 出力を PDF、Postscript、GIF、PNG、SVG、PCL ファイルはもちろん、プリンタに直接送信できます。また、ユニバーサル印刷システムには出力をカスタマイズできるようになる数多くのオプションがあります。このシステムは、SAS がサポートしているすべてのオペレーティングシステムで使用できます。

ライブラリ参照名
SAS ライブラリの場所に関連付けられた SAS 名。たとえば、MYLIB.MYFILE という名前では MYLIB がライブラリ参照名で、MYFILE が SAS ライブラリ内のファイルです。

リダイレクト
出力を標準出力以外の出力先に転送するか、または標準入力以外のソースから入力を読み取ること。

リモートブラウザサーバー
ユーザーのデスクトップで実行され、表示のために URL をブラウザに送信するソフトウェアエージェント。

リモートブラウジング
ユーザーのデスクトップのブラウザを使用して、HTML 情報(ヘルプテキストや ODS HTML 出力など)を表示するために SAS によって使用されるメカニズム。

ローカル SAS セッション
ローカルホストで実行される SAS セッション。ローカルセッションは SAS ステートメントを受け付け、リモートでサブミットされたステートメントをリモートホストに渡して

処理させます。ローカルセッションでは、ローカルセッションとリモートセッション両方の出力およびメッセージが管理されます。

ログインディレクトリ
ホームディレクトリを参照してください。

キーワード

G

H

I

J

K

L

R

S

か

さ

な

は

わ

大